SIEMENS

SIMATIC

# Industrial Software
# SIMATIC Safety - Configuring and Programming

Programming and Operating Manual

11/2014
A5E02714440-AD

# Legal information

## Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

**⚠ DANGER**

indicates that death or severe personal injury **will** result if proper precautions are not taken.

**⚠ WARNING**

indicates that death or severe personal injury **may** result if proper precautions are not taken.

**⚠ CAUTION**

indicates that minor personal injury can result if proper precautions are not taken.

**NOTICE**

indicates that property damage can result if proper precautions are not taken.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

## Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

## Proper use of Siemens products

Note the following:

**⚠ WARNING**

Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

## Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

## Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Division Digital Factory
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E02714440-AD
℗ 11/2014 Subject to change

# Important notes

## Purpose of this documentation

The information in this documentation enables you to configure and program SIMATIC Safety fail-safe systems. In addition, you will obtain information on acceptance of a SIMATIC Safety F-system.

## Basic knowledge requirements

General basic knowledge of automation engineering is needed to understand this documentation. Basic knowledge of the following is also necessary:

- Fail-safe automation systems
- S7-300/400/1200/1500/WinAC RTX F automation systems
- Distributed I/O systems on PROFIBUS DP/PROFINET IO
- Totally Integrated Automation Portal, including:
  - Hardware configuration with the *hardware and network editor*
  - Programming in the LAD and FBD programming languages using the *program editor*.
  - Communication between CPUs

## Scope of this documentation

This documentation is valid for the optional packages *STEP 7 Safety Advanced V13 SP1* and *STEP 7 Safety Basic V13 SP1.*

The optional packages *STEP 7 Safety Advanced V13 SP1* and *STEP 7 Safety Basic V13 SP1* are used for configuration and programming of the fail-safe SIMATIC Safety system.

In this context, integration of the fail-safe I/O listed below in SIMATIC Safety is also addressed:

- ET 200MP fail-safe modules
- ET 200SP fail-safe modules
- ET 200S fail-safe modules
- ET 200eco fail-safe I/O modules
- ET 200pro fail-safe modules
- ET 200iSP fail-safe modules
- S7-300 fail-safe signal modules
- S7-1200 fail-safe modules
- Fail-safe GSD based DP slaves
- Fail-safe GSD based I/O devices

## Approvals

The SIMATIC Safety F-system is certified for use in safety mode up to:

- Safety integrity level SIL3 in accordance with IEC 61508:2010
- Performance Level (PL) e and category 4 in accordance with ISO 13849-1:2006 or EN ISO 13849-1:2008

## Incorporation in the information landscape

Depending on your application, you will need the following supplementary documentation when working with *STEP 7 Safety*.

This documentation includes references to the supplementary documentation where appropriate.

| Documentation | Brief description of relevant content |
|---|---|
| For the SIMATIC Safety system | Depending on which F-CPU you are using, you will need the following documentation:<br>• The S7-1500 automation system (http://support.automation.siemens.com/WW/view/en/59191792) system manual describes the installation and wiring of S7-1500 systems.<br>• The Device manuals (http://support.automation.siemens.com/WW/view/en/67295862/133300) describe S7-1500 F-CPUs.<br>• The "S7-300, CPU 31xC and CPU 31x: Installation" (http://support.automation.siemens.com/WW/view/en/13008499) operating instructions describe the installation and wiring of S7-300 systems.<br>• The "CPU 31xC and CPU 31x, Technical Data" (http://support.automation.siemens.com/WW/view/en/12996906) device manual describes the CPUs 315-2 DP and PN/DP, the CPU 317-2 DP and PN/DP, and the CPU 319-3 PN/DP.<br>• The "S7-400 Automation System, Installation (http://support.automation.siemens.com/WW/view/en/1117849) installation manual describes the installation and wiring of S7-400 systems.<br>• The "S7-400 Automation System, CPU Data" (http://support.automation.siemens.com/WW/view/en/23904550) reference manual describes the CPUs 414-3 PN/DP, the CPU 416-2, and the CPU 416-3 PN/DP.<br>• The "ET 200S Interface Module IM 151-7 CPU" (http://support.automation.siemens.com/WW/view/en/12714722) manual describes the IM 151-7 CPU.<br>• The "ET 200S, Interface Module IM 151-8 PN/DP CPU" (http://support.automation.siemens.com/WW/view/en/47409312) manual describes the IM 151-8 PN/DP CPU.<br>• The "ET 200S, Interface Module IM 154-8 CPU" (http://support.automation.siemens.com/WW/view/de/24363739/0/en) manual describes the IM 154-8 CPU.<br>• The Manual "Windows Automation Center RTX WinAC RTX (F) 2010 (http://support.automation.siemens.com/WW/view/en/43715176)" describes the WinAC RTX 2010 and the WinAC RTX F 2010.<br>• Each F-CPU that can be used has its own product information. The product information only describes the deviations from the respective standard CPUs. |

| Documentation | Brief description of relevant content |
|---|---|
| System manual "S7-1200 Functional Safety manual" (http://support.automation.siemens.com/WW/view/en/34612486/133300) | Describes the F-CPUs S7-1200 and the fail-safe modules S7-1200 (including installation, wiring, and technical specifications) |
| "S7-1500/ET200MP system manual (http://support.automation.siemens.com/WW/view/en/59191792)" system manual and the product manuals (http://support.automation.siemens.com/WW/view/en/57251728/133300) for the corresponding ET 200MP fail-safe modules | Describes the hardware of the ET 200MP fail-safe modules (including installation, wiring, and technical specifications) |
| "ET 200SP distributed I/O system (http://support.automation.siemens.com/WW/view/en/58649293)" system manual and the product manuals (http://support.automation.siemens.com/WW/view/en/55679227/133300) for the corresponding ET 200SP fail-safe modules | Describes the hardware of the ET 200SP fail-safe modules (including installation, wiring, and technical specifications) |
| "ET 200eco Distributed I/O Station Fail-safe I/O Block (http://support.automation.siemens.com/WW/view/en/19033850)" manual | Describes the hardware of the ET 200eco fail-safe I/O module (including installation, wiring, and technical specifications) |
| "Distributed I/O System ET 200S, Fail-Safe Modules (http://support.automation.siemens.com/WW/view/en/27235629)" operating instructions | Describes the hardware of the ET 200S fail-safe modules (including installation, wiring, and technical specifications) |
| "S7-300 Automation System, ET 200M Distributed I/O System, Fail-safe Signal Modules (http://support.automation.siemens.com/WW/view/en/19026151)" manual | Describes the hardware of the S7-300 fail-safe signal modules (including installation, wiring, and technical specifications) |
| "Distributed I/O system ET 200pro, fail-safe I/O modules (http://support.automation.siemens.com/WW/view/en/22098524)" operating instructions | Describes the hardware of the ET 200pro fail-safe modules (including installation, wiring, and technical specifications) |
| "ET 200iSP distributed I/O device - Fail-safe modules (http://support.automation.siemens.com/WW/view/en/47357221)" operating instructions | Describes the hardware of the ET 200iSP fail-safe modules (including installation, wiring, and technical specifications) |
| Online help for *STEP 7* | • Describes the operation of the standard tools in *STEP 7*<br>• Contains information regarding configuration and parameter assignment of hardware<br>• Contains a description of the FBD and LAD programming languages |

The complete *SIMATIC S7* documentation is available on DVD. You can find more information on the Internet (http://www.automation.siemens.com/mcms/industrial-automation-systems-simatic/en/manual-overview/manual-collection/Pages/Default.aspx).

## Guide

This documentation describes how to work with the optional packages *STEP 7 Safety.* It includes instructions and reference sections (description of the instructions for the safety program).

The following topics are addressed:

- Configuration of SIMATIC Safety
- Access protection for SIMATIC Safety
- Programming of the safety program (safety-related user program)
- Safety-related communication
- Instructions for the safety program
- Support for the system acceptance
- Operation and maintenance of SIMATIC Safety
- Monitoring and response times

## Conventions

In this documentation, the terms "safety engineering" and "fail-safe engineering" are used synonymously. The same applies to the terms "fail-safe" and "F-".

When *"STEP 7 Safety Advanced V13 SP1" or "STEP 7 Safety Basic V13 SP1"* appears in italics, it refers to the optional package for the "SIMATIC Safety" F-system.

*"STEP 7 Safety V13 SP1"* stands for the optional package *"STEP 7 Safety Advanced V13 SP1"* and the optional package *"STEP 7 Safety Basic V13 SP1"*.

"(S7-300)" indicates that the section **only** applies to S7-300 F-CPUs as well as WinAC RTX F. S7-300 F-CPUs also includes the F-CPUs ET 200S and ET 200pro (IM F-CPUs).

"(S7-400)" indicates that the section **only** applies to S7-400 F-CPUs.

"(S7-1200)" indicates that the section **only** applies to S7-1200 F-CPUs.

"(S7-1500)" indicates that the section **only** applies to S7-1500 F-CPUs. S7-1500 F-CPUs also includes ET 200SP F-CPUs.

The scopes can be combined.

The term "safety program" refers to the fail-safe portion of the user program and is used instead of "fail-safe user program," "F-program," etc. For purposes of contrast, the non-safety-related part of the user program is referred to as the "standard user program".

All fail-safe modules and instructions are highlighted in yellow to distinguish them from the modules and instructions of the standard user programs on the software interface (e.g., in the project tree). Similarly, the fail-safe parameters of F-CPUs and F-I/O are highlighted in yellow in the hardware configuration.

Each warning is marked with a unique number at the end of the text. This enables you to easily reference other documents to obtain an overview of the safety requirements for the system.

## Additional support

If you have further questions about the use of products presented in this manual, contact your local Siemens representative.

You can find information on whom to contact on the Web (http://www.siemens.com/automation/partner).

A guide to the technical documentation for the various SIMATIC products and systems is available on the Web (http://www.siemens.com/simatic-tech-doku-portal).

You can find the online catalog and online ordering system on the Web (www.siemens.com/industrymall).

## Training center

We offer courses to help you get started with the S7 automation system. Contact your regional training center or the central training center in Nuremberg (90327), Federal Republic of Germany.

You can find more information on the Internet (http://www.sitrain.com).

## Functional Safety Services

Siemens Functional Safety Services support you with a comprehensive package of services from risk assessment to verification all the way to system commissioning and modernization. We also offer consultation on the use of fail-safe and fault-tolerant SIMATIC S7 automation systems.

You can find more detailed information on the Internet (www.siemens.com/safety-services).

Contact us at safety-services.industry@siemens.com if you have any questions.

## Technical Support

To contact Technical Support for all Industry Automation products, use the Support Request Web form (http://www.siemens.com/automation/support-request).

You can find additional information about our Technical Support on the Web (http://www.siemens.com/automation/service).

## Industry Online Support

In addition to our documentation, we also offer a comprehensive technical knowledge base on the Internet (http://www.siemens.com/automation/service&support).

There, you can find the following information:

- Newsletters providing the latest information on your products
- A search engine in Industry Online Support for locating the documents you need
- A forum where users and experts from all over the world exchange ideas
- Your local contact
- Information about on-site services, repairs, and about spare parts Lots more can be found on our "Services" page.

## Important note for maintaining the operational safety of your system

**Note**

The operators of systems with safety-related characteristics must adhere to operational safety requirements. The supplier is also obliged to comply with special product monitoring measures. Siemens publishes a special newsletter to keep system operators informed about product developments and properties which may form important issues in terms of operational safety. You should subscribe to the corresponding newsletter in order to obtain the latest information and to allow you to modify your system accordingly. Please go to the Internet (https://www.automation.siemens.com/WW/newsletter/guiThemes2Select.aspx?HTTPS=REDIR&subjectID=2) and register for the following newsletters:

- SIMATIC S7-300/S7-300F
- SIMATIC S7-400/S7-400H/S7-400F/FH
- SIMATIC S7-1500/SIMATIC S7-1500F
- SIMATIC S7-1200/SIMATIC S7-1200F
- Distributed I/O
- SIMATIC Industrial Software

To receive these newsletters, select the check box "Update".

## Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, solutions, machines, equipment and/or networks. They are important components in a holistic industrial security concept. With this in mind, Siemens' products and solutions undergo continuous development. Siemens recommends strongly that you regularly check for product updates.

For the secure operation of Siemens products and solutions, it is necessary to take suitable preventive action (e.g. cell protection concept) and integrate each component into a holistic, state-of-the-art industrial security concept. Third-party products that may be in use should also be considered. You can find more information about industrial security on the Internet (http://www.siemens.com/industrialsecurity).

To stay informed about product updates as they occur, sign up for a product-specific newsletter. You can find more information on the Internet (http://support.automation.siemens.com).

# Table of contents

# Product Overview

# 1

## 1.1 Overview

### SIMATIC Safety fail-safe system

The SIMATIC Safety fail-safe system is available to implement safety concepts in the area of machine and personnel protection (for example, for emergency STOP devices for machining and processing equipment) and in the process industry (for example, for implementation of protection functions for safety devices of instrumentation and controls and of burners).

**⚠ WARNING**

The SIMATIC Safety F-system is used to control processes and to force the system into a safe state after shutdown.

SIMATIC Safety can only be used for controlling processes in which an immediate shutdown does not pose a danger to persons or the environment. *(S062)*

### Achievable safety requirements

SIMATIC Safety F-systems can satisfy the following safety requirements:

- Safety integrity level SIL3 in accordance with IEC 61508:2010
- Performance level (PL) e and category 4 in accordance with ISO 13849-1:2006 or EN ISO 13849-1:2008

## Principles of safety functions in SIMATIC Safety

Functional safety is implemented principally through safety functions in the software. Safety functions are executed by the SIMATIC Safety system in order to bring the system to a safe state or maintain it in a safe state in case of a dangerous event. Safety functions are contained mainly in the following components:

- In the safety-related user program (safety program) in the F-CPU
- In the fail-safe inputs and outputs (F-I/O)

The F-I/O ensure the safe processing of field information (sensors: e.g., emergency OFF pushbutton, light barriers, actuators for motor control, for example). They have all of the required hardware and software components for safe processing, in accordance with the required Safety Integrity Level. The user only has to program the user safety function. The safety function for the process can be provided through a user safety function or a fault reaction function. In the event of an error, if the F-system can no longer execute its actual user safety function, it executes the fault reaction function; for example, the associated outputs are shut down, and the F-CPU switches to STOP mode, if necessary.

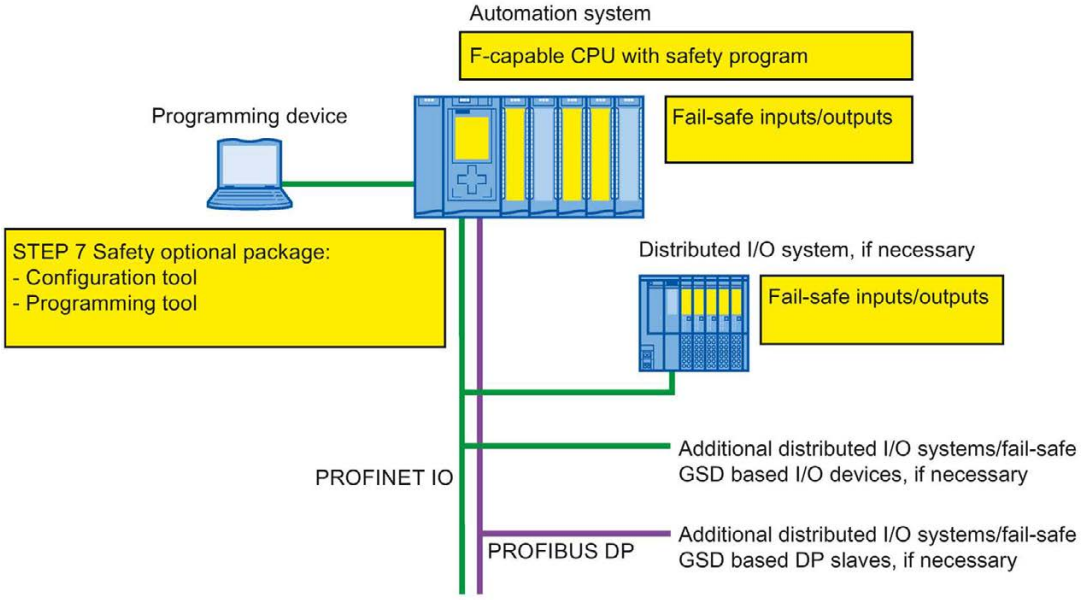
## Example of user safety function and fault reaction function

In the event of overpressure, the F-system will open a valve (user safety function). In the event of a dangerous malfunction of the F-CPU, all outputs are shut down (fault reaction function). This means that the valve opens in the event of overpressure and the other actuators also switch to safe mode. For a non-faulty F-system, only the valve would be opened.

# 1.2 Hardware and Software Components

## Hardware and software components of SIMATIC Safety

The following figure provides an overview of the hardware and software components required to configure and operate a SIMATIC Safety F-system.




## Hardware components for PROFIBUS DP

You can use the following fail-safe components in SIMATIC Safety F-systems on PROFIBUS DP:

- F-CPUs with DP interface, such as CPU 1516F-3 PN/DP
- Fail-safe inputs and outputs (F-I/O), such as:
  - S7-300 fail-safe signal modules in ET 200M
  - ET 200MP fail-safe modules
  - ET 200SP fail-safe modules
  - ET 200S fail-safe modules
  - ET 200pro fail-safe modules
  - ET 200iSP fail-safe modules
  - Fail-safe I/O modules ET 200eco (S7-300, S7-400)
  - Fail-safe GSD based DP slaves (light grid, laser scanner, etc.)

You have the option to expand the configuration with standard I/O.

The following CPs/CMs can be used in a SIMATIC Safety F-system on PROFIBUS DP for connection to distributed F-I/O:

- CP 443-5 Basic
- CP 443-5 Extended
- CM 1542-5
- CP 1542-5

## Hardware components for PROFINET IO

You can use the following fail-safe components in SIMATIC Safety F-systems on PROFINET IO:

- F-CPUs with PN interface, such as CPU 1516F-3 PN/DP
- Fail-safe inputs and outputs (F-I/O), such as:
  - S7-300 fail-safe signal modules in ET 200M
  - ET 200MP fail-safe modules
  - ET 200SP fail-safe modules
  - ET 200S fail-safe modules
  - ET 200pro fail-safe modules
  - ET 200iSP fail-safe modules
  - Fail-safe GSD based I/O devices (light grid, laser scanner, etc.)

You have the option to expand the configuration with standard I/O.

The following CPs/CMs can be used in a SIMATIC Safety F-system on PROFINET IO for connection to distributed F-I/O:

- CP 443-1
- CP 443-1 Advanced-IT
- CM 1542-1

## Hardware components for central configuration

You can use the following fail-safe components in SIMATIC Safety F-systems centrally on an F-CPU:

- S7-300 fail-safe signal modules
- S7-1200 fail-safe modules
- ET 200MP fail-safe modules
- ET 200SP fail-safe modules
- ET 200S fail-safe modules
- ET 200pro fail-safe modules

You have the option to expand the configuration with standard I/O.

## Required software components

You require the following software components:

- *SIMATIC STEP 7 Basic V13 SP1* with the *STEP 7 Safety Basic V13 SP1* optional package

  **or**

- *SIMATIC STEP 7 Professional V13 SP1* with the *STEP 7 Safety Basic V13 SP1* optional package

  **or**

- *SIMATIC STEP 7 Professional V13 SP1* with the *STEP 7 Safety Advanced V13 SP1* optional package

## *STEP 7 Safety* optional packages

This documentation describes the optional packages *STEP 7 Safety Advanced V13 SP1* and *STEP 7 Safety Basic V13 SP1*. The *STEP 7 Safety* optional packages are the configuration and programming software for the SIMATIC Safety F-system. With *STEP 7 Safety*, you receive the following:

- Support for configuring the F-I/O in the *hardware and network editor* of the TIA Portal
- Support for creating the safety program using LAD and FBD and integrating error detection functions into the safety program
- Instructions for programming your safety program in LAD and FBD, which you are familiar with from the standard user programs
- Instructions for programming your safety program in LAD and FBD with special safety functions

Moreover, *STEP 7 Safety* offers functions for comparing safety programs and for assisting you with the system acceptance.

**⚠ WARNING**

The F-CPU and F-I/O must be configured in the *hardware and network editor* of the TIA Portal as described in this documentation. F-blocks must be created with the *program editor* of the TIA Portal as described in this documentation. For system acceptance, you have to use the safety summary created according to this documentation. Any other procedures are not permitted. *(S056)*

## Additional optional packages

In addition to the *STEP 7 Safety* optional packages, you can use additional optional packages with F-I/O and F-CPUs within the SIMATIC Safety F-system. For example, SINUMERIK.

The installation, parameter assignment and programming as well as what is important to note during system acceptance, is described in the documentation for the specific optional packages.

Please also read the notes in Configurations supported by the SIMATIC Safety F-system (Page 50).

## Safety program

You can create a safety program using the *program editor*. You can program fail-safe FBs and FCs in the FBD or LAD programming languages using the instructions from the optional package and create fail-safe DBs.

Safety checks are automatically performed and additional fail-safe blocks for error detection and error reaction are inserted when the safety program is compiled. This ensures that failures and errors are detected and appropriate reactions are triggered to maintain the F-system in the safe state or bring it to a safe state.

In addition to the safety program, a standard user program can be run on the F-CPU. A standard program can coexist with a safety program in an F-CPU because the safety-related data of the safety program are protected from being affected unintentionally by data of the standard user program.

Data can be exchanged between the safety program and the standard user program in the F-CPU by means of bit memory or data of a standard DB or by accessing the process image of the inputs and outputs.

## See also

Data Transfer from the Safety Program to the Standard User Program (Page 160)

# 1.3 Installing/uninstalling the STEP 7 Safety Basic V13 SP1 optional package

## Software requirements for *STEP 7 Safety Basic V13* SP1

At a minimum, one of the following software packages must be installed on the programming device or PC:

- *SIMATIC STEP 7 Basic V13 SP1*
- *SIMATIC STEP 7 Professional V13 SP1*

## Reviewing the Readme file

The readme file contains important up-to-date information about the software (for example, Windows versions supported). You can display the readme file in the setup program or open it later in the *SIMATIC STEP 7* information system.

## Installing *STEP 7 Safety Basic V13 SP1*

1. Start the programming device or PC on which the *"SIMATIC STEP 7 Basic V13 SP1"* or *"SIMATIC STEP 7 Professional V13 SP1*" software package has been installed, and make sure that *SIMATIC STEP 7 Basic V13 SP1 or SIMATIC STEP 7 Professional V13 SP1* is closed.
2. Insert the optional package product DVD.
3. Initiate the *SETUP.EXE* program on the DVD.
4. Follow the instructions of the Setup program, bearing in mind the information in the readme file.

## Displaying integrated Help

The help on *STEP 7 Safety Basic V13 SP1* is completely integrated into the information system of *SIMATIC STEP 7 Basic V13 SP1 or SIMATIC STEP 7 Professional V13 SP1*. You have the following two options to open the integrated help:

- In the "Help" menu, select the "Show help" command or press <F1> to show the appropriate help for the context.
- Click on the link within a tool tip cascade to go directly to a place with more information within the help.

## Uninstalling *STEP 7 Safety Basic V13 SP1*

To uninstall *STEP 7 Safety Basic V13 SP1*, proceed as described in the *STEP 7 Help* under "Uninstall".

### Post-uninstall procedures

After uninstalling the *STEP 7 Safety Basic V13 SP1* optional package, you can no longer edit projects with F-CPUs whose F-capability is enabled.

You may open and continue working with projects with F-CPUs whose F-capability was previously disabled (see also Configuring the F-CPU (Page 42)).

## 1.4 Installing/uninstalling the STEP 7 Safety Advanced V13 SP1 optional package

### Software requirements for *STEP 7 Safety Advanced V13 SP1*

At a minimum, the following software package must be installed on the programming device or PC:

- *SIMATIC STEP 7 Professional V13 SP1*

### Reviewing the Readme file

The readme file contains important up-to-date information about the software (for example, Windows versions supported). You can display the readme file in the setup program or open it later in the *SIMATIC STEP 7* information system.

### Installing *STEP 7 Safety Advanced V13 SP1*

1. Start the programming device or PC on which the *"SIMATIC STEP 7 Professional V13 SP1"* software package has been installed, and make sure that *SIMATIC STEP 7 Professional V13 SP1* is closed.
2. Insert the optional package product DVD.
3. Initiate the *SETUP.EXE* program on the DVD.
4. Follow the instructions of the Setup program, bearing in mind the information in the readme file.

### Displaying integrated Help

The help on *STEP 7 Safety Advanced V13 SP1* is completely integrated into the information system of *SIMATIC STEP 7 Professional V13 SP1*. You have the following two options to open the integrated help:

- In the "Help" menu, select the "Show help" command or press <F1> to show the appropriate help for the context.
- Click on the link within a tool tip cascade to go directly to a place with more information within the help.

### Uninstalling *STEP 7 Safety Advanced V13 SP1*

To uninstall *STEP 7 Safety Advanced V13 SP1*, proceed as described in the *STEP 7 Help* under "Uninstall".

### Post-uninstall procedures

After uninstalling the *STEP 7 Safety Advanced V13 SP1* optional package, you can no longer edit projects with F-CPUs whose F-capability is enabled.

You may open and continue working with projects with F-CPUs whose F-capability was previously disabled (see also Configuring the F-CPU (Page 42)).

## 1.5 Installing the STEP 7 Safety PowerPack

### Software requirements for *STEP 7 Safety PowerPack*

At a minimum, the following software package must be installed on the programming device or PC:

- *SIMATIC STEP 7 Safety Basic V13 SP1*
- *SIMATIC STEP 7 Professional V13 SP1*

### Installing *STEP 7 Safety PowerPack*

1. Start the *Automation License Manager* on the programming device/PC on which the *"SIMATIC STEP 7 Professional V13 SP1"* software package is installed.
2. Install the license included in the *STEP 7 Safety PowerPack* as described in the *Automation License Manager help*.

### Uninstalling *STEP 7 Safety PowerPack*

To uninstall the license included in the *STEP 7 Safety PowerPack*, proceed as described in the *STEP 7 Safety PowerPack Help*.

# 1.6 Migrating projects from S7 Distributed Safety V5.4 SP5 to STEP 7 Safety Advanced

## Introduction

In *STEP 7 Safety Advanced*, you can continue to use projects with safety programs that you created with *S7 Distributed Safety V5.4 SP5*. **To that end, the projects must have been compiled in *S7 Distributed Safety V5.4 SP5* and then migrated.**

## Requirement

*STEP 7 Safety Advanced*, *S7 Distributed Safety V5.4 SP5*, and the *F-Configuration Pack* used to create the project must all be installed on the computer you are using for migration. The *F-Configuration Pack V5.4 SP5* to *V5.5 SP11* is supported.

## Prior to migration

Delete all F-blocks not required by the safety program in your *S7 Distributed Safety V5.4 SP5* project prior to migration.

## Procedure as in *STEP 7 Professional*

Proceed just as you would for standard projects to migrate projects from *S7 Distributed Safety V5.4 SP5* to *STEP 7 Safety Advanced*. Once the migration is complete, you should verify using the collective F-signature whether the project was migrated unchanged.

**Note**

If you use the safety program to migrate F-blocks with know-how protection, remove the know-how protection prior to migration.

You can assign the F-blocks know-how protection again once the migration is completed.

This migration approach is described in the "Migration" section of the *STEP 7 Professional* Help. Special considerations about *STEP 7 Safety Advanced* are described below.

## Older hardware versions

Older versions of F-hardware may not be supported by *STEP 7 Safety Advanced*.

If you have used and configured versions of F-CPUs and F-I/O in S7 Distributed Safety projects that are not approved for *STEP 7 Safety Advanced*, you will need to upgrade this hardware to the new version in *S7 Distributed Safety V5.4 SP5* and the corresponding *F-Configuration Pack*. Once the upgrade is completed, migration to *STEP 7 Safety Advanced* is feasible. A list of approved hardware is available on the Internet. (http://support.automation.siemens.com/WW/view/en/49368678/133100)

## Particularities for safety-related CPU-CPU communication via S7 connections

You can find information about the special considerations for migrated projects with safety-related CPU-CPU communication via S7 connections in Safety-related communication via S7 connections (Page 212). Please also note Communication with S7 Distributed Safety via S7 connections (Page 222).

## Particularities for ESTOP1 or FDBACK instructions

Information on the special considerations when using the ESTOP1 and FDBACK instructions can be found in the "Instruction versions" section in ESTOP1: Emergency STOP up to stop category 1 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 348) and FDBACK: Feedback monitoring (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 393).

## Post-migration procedures

Once migration is complete, you have a complete project with a safety program which has retained the program structure of *S7 Distributed Safety* and the collective F-signature. F-blocks from the S7 Distributed Safety F-library (V1) are converted into instructions that are provided by *STEP 7 Safety Advanced*.

If you receive an error message after migration and subsequent compilation of the hardware that the F-source address does not match the basis for the PROFIsafe addresses of the F-CPU, change the basis for the PROFIsafe addresses of the F-CPU at the corresponding interface. The F-source addresses of all F-I/O connected to the CPU will be reassigned.

The migrated project therefore does not need to be re-accepted; it can be loaded as is to the F-CPU as long as it has not been modified or compiled since migration.

**Note**

**Safety summary**

You cannot create a safety summary in *STEP 7 Safety Advanced* for a migrated project. The printout of the project created with *S7 Distributed Safety V5.4 SP5* and the corresponding acceptance documents are still valid, because the collective F-signature has been retained.

After migration of an SM 326; DI 24 x DC 24V (6ES7 326-1BK01-0AB0 and 6ES7 326-1BK02-0AB0), the following error message may be output during compilation of the hardware configuration: "F_IParam_ID_1: Value outside the permitted range".

**Solution:** Delete the module and reinsert it.

## Using the latest versions used in the safety program

If you want to expand the migrated safety program, we recommend that you update the system library elements used in safety program, specified in the *Safety Administration Editor* in the "Settings" area, to the latest available version before compiling with *STEP 7 Safety Advanced* for the first time.

## Using the latest version of instructions

If you want to expand the migrated safety program, we recommend that you update to the latest version of the instructions used before compiling with *STEP 7 Safety Advanced* for the first time. Read the information on instruction versions for each instruction.

## Compiling the migrated safety programs

As a result of compilation of the migrated project with *STEP 7 Safety Advanced*, the existing program structure (with F-CALL) is transformed into the new program structure of *STEP 7 Safety Advanced* (with main safety block). This changes the collective F-signature and a new acceptance for the safety program may have to be executed.

**Note**

Note that compiling the migrated safety program extends the runtime of the F-runtime group(s) and increases the memory requirements of the safety program (see also Excel file for calculating response time (http://support.automation.siemens.com/WW/view/en/49368678/133100)).

## 1.7 Migrating an S7-300/400 F-CPU

To migrate an F-CPU S7-300/400 onto an F-CPU S7-1500, proceed as with the migration of a standard CPU S7-300/400 onto a CPU S7-1500.

Points to note after migration:

- Non-automatable actions
  - Creating an F-runtime group and assigning it to the main safety block.
  - The hardware configuration of the initial F-CPU is not automatically transferred to an S7-1500 F-CPU. Implement the hardware configuration of the new CPU manually after migration.
- Instructions not supported:
  - OV
  - MUTING
  - TWO_HAND
  - WR_FDB
  - RD_FDB
  - OPN
  - SENDS7
  - RCVS7
- Data types not supported
  - DWORD
- Changes to safety program programming
  - F_GLOBDB.VKE0/1 replaced by FALSE/TRUE (Page 90).
  - Readable values from the F_GLOBDB replaced by the F-runtime group information DB. Additional information is available under F-shared DB (S7-300, S7-400) (Page 114) and F-runtime group information DB (S7-1200, S7-1500) (Page 115).
  - Replacement of the QBAD_I_xx or QBAD_O_xx tag by the value status. Additional information is available under Value status (S7-1200, S7-1500) (Page 124) and F-I/O DB (Page 129).
- F-runtime group communication is not supported.
- New naming convention when naming the F-I/O DBs
- Modified behavior of QBAD and PASS_OUT (Page 135) tags for F-I/O with "RIOforFA-Safety" profile

Compile the safety program and eliminate any compilation errors displayed.

**Note**

A new acceptance must be carried out following F-CPU migration.

### See also

Programming (Page 84)

## 1.8 Upgrading projects from STEP 7 Safety Advanced V11

The safety program signature does not change after upgrading. Acceptance of changes is therefore not required.

Perform the upgrade following the usual procedure for *STEP 7 Professional*. Once upgraded, a project can no longer be opened with *STEP 7 Safety Advanced V11*. We recommend that you back up the project before upgrading.

**Note**

Adjustments are required before you can continue working in the upgraded safety program in *STEP 7 Safety Advanced V13 SP1*:

There was a product warning for *STEP 7 Safety Advanced V11* regarding setting the parameters "Discrepancy behavior" and "Reintegration after discrepancy error" for the fail-safe digital input and output modules 4F-DI/3F-DO DC24V/2A (6ES7138-4FC01-0AB0, 6ES7138-4FC00-0AB0). These parameters could be displayed incorrectly in certain combinations.

Based on the handling instructions in this product warning, you used a conversion table to set the affected parameters so that they were displayed incorrectly in the safety summary and hardware configuration in order for them to have the correct effect in the F-module. You also corrected the safety summary by hand to document the actual behavior of F-modules.

To reverse these changes, follow these steps:

1. Compile the upgraded project with *STEP 7 Safety Advanced V13 SP1*. An error message is displayed for each F-module in *STEP 7 Safety Advanced V11* the parameters of which you have corrected: "The CRC (F_Par_CRC) of the module (xxx) does not match the calculated value (yyy)."
2. Adapt the parameter assignment of each F-module for which this error message is displayed to match your handwritten corrections in the safety summary.
3. Do this for each F-CPU and then compile the safety program.
4. If the collective F-signature following compiling corresponds to the collective F-signature on the safety summary, you have made all the necessary corrections.

### Use of CPs

F-I/Os operated downstream from a CP 443-5 Basic, CP443-5 Extended, CP443-1 or CP 443-1 Advanced-IT have not been assigned a unique F-destination address.

As soon as you compile the hardware in a project with such F-I/Os in *STEP 7 Safety V13 SP1*, you are notified for the affected F-I/Os. You have to assign new, unique F-destination addresses for the reported F-I/Os. Additional information is available under PROFIsafe addresses for F-I/O of PROFIsafe address type 1 (Page 52), PROFIsafe addresses for F-I/O of PROFIsafe address type 2 (Page 54) and Peculiarities when configuring fail-safe GSD based DP slaves and fail-safe GSD based I/O devices (Page 61).

This changes the collective F-signature of the safety program. However, you can verify that only the changed F-destination addresses have caused this change:

- The F-parameter signature (without address) for each changed F-I/O remains the same.
- Only the affected F-I/O DBs are listed in the comparison editor of the safety program with the filter set to "Compare only F-blocks relevant for certification".

## 1.9 Upgrading projects from STEP 7 Safety Advanced V12

The safety program signature does not change after upgrading. Acceptance of changes is therefore not required.

Perform the upgrade following the usual procedure for *STEP 7 Professional*. Once upgraded, a project can no longer be opened with *STEP 7 Safety Advanced V12*. We recommend that you back up the project before upgrading.

## 1.10 Working with projects from STEP 7 Safety Advanced V12

You can open, edit and save projects from *STEP 7 Safety Advanced V12*.

If you want to use an S7-1200 F-CPU or a newer S7-1500 F-CPU, you have to upgrade the project.
Please note that projects can no longer be edited with *STEP 7 Safety Advanced V12* following the upgrade.

## 1.11 Upgrading projects from STEP 7 Safety Advanced V13

The safety program signature does not change after upgrading. Acceptance of changes is therefore not required.

Perform the upgrade following the usual procedure for *STEP 7 Professional*. Once upgraded, a project can no longer be opened with *STEP 7 Safety Advanced V13*. We recommend that you back up the project before upgrading.

## 1.12 Working with projects from STEP 7 Safety Advanced V13

You can open, edit and save projects from *STEP 7 Safety Advanced V13*.

If you want to use an S7-1200 F-CPU or a newer S7-1500 F-CPUs, you have to upgrade the project.
Please note that projects can no longer be edited with *STEP 7 Safety Advanced V13* following the upgrade.

## 1.13 First steps

### Getting Started in SIMATIC Safety

Three Getting Started documents are available to help you begin using SIMATIC Safety.

The Getting Started documentation is an instruction manual that provides a step-by-step description of how to create a project with SIMATIC Safety. It gives you the opportunity to quickly become familiar with the scope of features of SIMATIC Safety.

### Contents

The Getting Started documentation describes the creation of a single, continuous project that is extended with each section. Based on the configuration, you program a fail-safe shutdown, make changes to the programming, and accept the system.

In addition to the step-by-step instructions, the Getting Started documentation also gives you background information for every new topic, which explains the functions used in more detail and how they interrelate.

### Target audience

The Getting Started documentation is intended for beginners but is also suitable for users who are switching from *S7 Distributed Safety*.

### Download

Three Getting Started documents are available as PDF files for free download in the Industry Online Support:

- STEP 7 Safety Advanced with F-CPUs S7-300/400 (http://support.automation.siemens.com/WW/view/en/49972838)
- STEP 7 Safety Basic with S7-1200 F-CPUs (http://support.automation.siemens.com/WW/view/en/34612486/133300) (part of the manual "S7-1200 Functional Safety manual")
- STEP 7 Safety Advanced with F-CPUs S7-1500 (http://support.automation.siemens.com/WW/view/en/101177693)

# Configuring

# 2

## 2.1 Overview of Configuration

### Introduction

You configure a SIMATIC Safety F-system basically in the same way as a standard S7-300, S7-400, S7-1200, S7-1500 or ET200MP, ET 200SP, ET 200S, ET 200iSP, ET 200eco or ET 200pro automation system in *STEP 7.*

This section presents only the essential differences compared to standard configuration you encounter when configuring a SIMATIC Safety F-system.

This documentation distinguishes between two groups of F-I/O:

### F-I/Os of PROFIsafe address type 1

F-I/Os which ensure the uniqueness of the PROFIsafe address solely with the F-destination address, for example, ET 200S F-modules. The PROFIsafe address is usually assigned by DIP switches.

### F-I/Os of PROFIsafe address type 2

F-I/Os which can ensure the uniqueness of the PROFIsafe address with a combination of F-source address and F-destination address, for example, ET 200MP F-modules. The PROFIsafe address is usually assigned with *STEP 7 Safety*.

### Which F-components can you configure with the *STEP 7 Safety Basic V13 SP1* optional package?

The following hardware components are configured for a SIMATIC Safety F-system with the *STEP 7 Safety Basic V13 SP1* optional package:

- S7-1200 F-CPUs
- F-I/Os of PROFIsafe address type 2
  - S7-1200 fail-safe modules (central configuration)

## Which F-components can you configure with the *STEP 7 Safety Advanced V13 SP1* optional package?

The following hardware components are configured for a SIMATIC Safety F-system with the *STEP 7 Safety Advanced V13 SP1* optional package:

- F-CPUs S7-300, S7-400, S7-1200, S7-1500 and WinAC RTX F
- F-I/Os of PROFIsafe address type 1:
  - S7-300 fail-safe signal modules
  - ET 200S fail-safe modules
  - ET 200pro fail-safe modules
  - ET 200iSP fail-safe modules
  - Fail-safe I/O modules ET 200eco (S7-300, S7-400)
  - Fail-safe GSD based DP slaves *
  - Fail-safe GSD based I/O devices *
- F-I/Os of PROFIsafe address type 2
  - S7-1200 fail-safe modules
  - ET 200SP fail-safe modules
  - ET 200MP fail-safe modules
  - Fail-safe GSD based DP slaves *
  - Fail-safe GSD based I/O devices *

* Consult the respective documentation to determine the PROFIsafe address type of a GSD based DP slave/GSD based I/O device. If in doubt, assume that the PROFIsafe address is type 1.

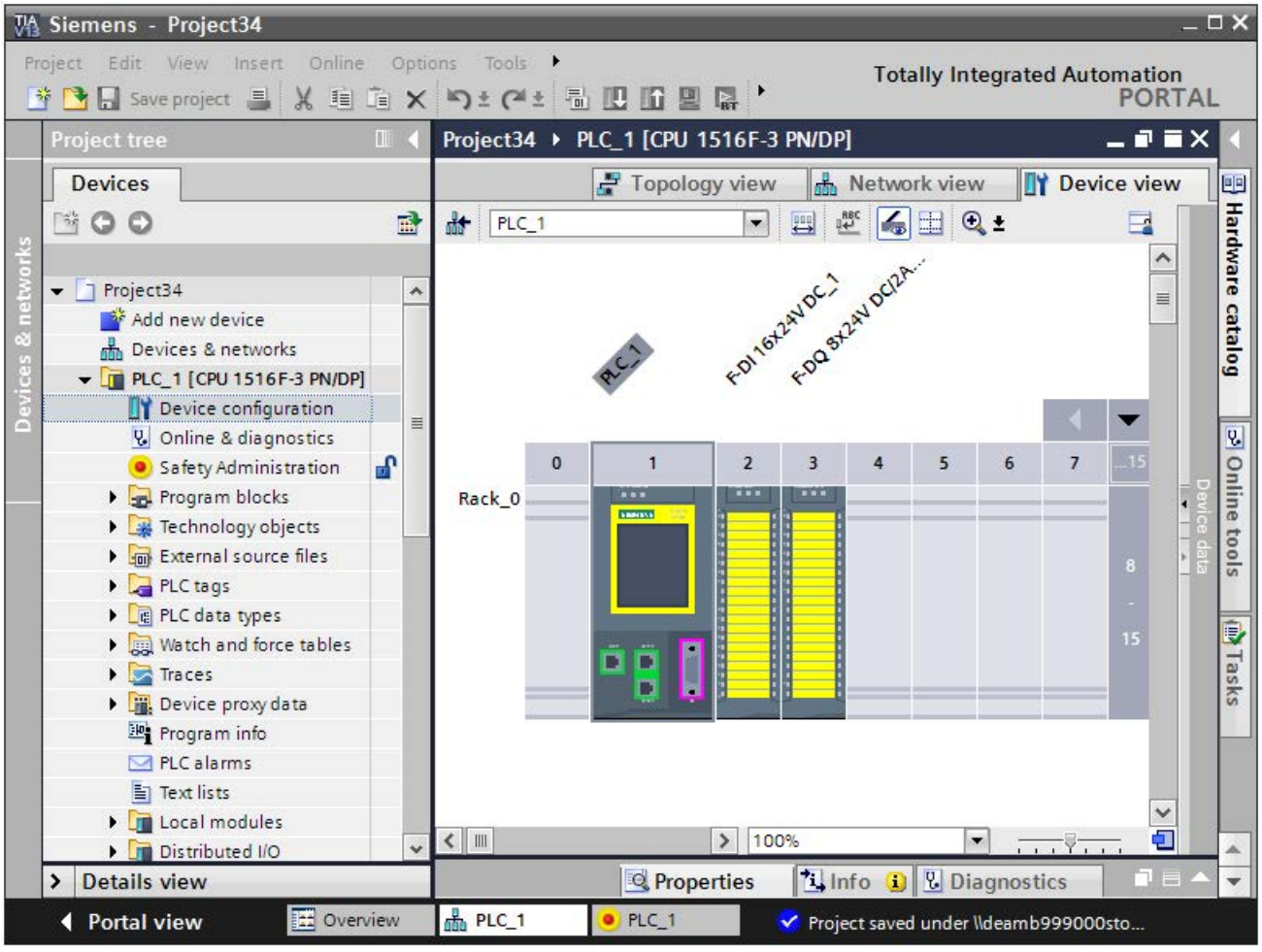
**Note**

"Shared Device" functionality is not permitted for F-I/O.

Safety-related I-slave-slave communication is not permitted for the fail-safe modules ET 200SP and ET 200MP.

## Example: Configured F-system in *STEP 7 Professional*

The following figure presents a configured F-system. You choose the fail-safe components in the "Hardware catalog" task card as you would do with standard components and place them in the work area of the network or device view. F-components are shown in yellow.




## Additional information

For detailed information on F-I/O, refer to the *manuals for the relevant F-I/O*.

## Which safety-related communication options can you configure?

You need to use the *hardware and network editor* to configure the following safety-related communication options (see Configuring and programming communication (S7-300, S7-400) (Page 163) or Configuring and programming communication (S7-1500) (Page 226):

- Safety-related master-master communication
- Safety-related master-master communication for S7 Distributed Safety
- Safety-related master-I-slave communication
- Safety-related I-slave-I-slave communication
- Safety-related I-slave-slave communication
- Safety-related IO controller-IO controller communication
- Safety-related IO controller-IO controller communication for S7 Distributed Safety
- Safety-related IO controller-I-device communication
- Safety-related IO controller-I-slave communication
- Safety-related communication via S7 connections
- Safety-related communication via S7 connections for S7 Distributed Safety or S7 F Systems

## 2.2 Particularities for configuring the F-System

### Configuring is the same as for standard components

You configure a SIMATIC Safety F-system in the same way as a standard S7 system. This means that you configure and assign parameters for the hardware in the *hardware and network editor* in a centralized system (F-CPU and F-IO if required e.g. CPU 1516F-3 PN/DP and F-module ET 200MP) and/or as distributed system (F-CPU, F-SMs in ET 200M, F-module ET 200MP, F-module ET 200SP, ET 200S, ET 200pro, ET 200iSP, ET 200eco, fail-safe GSD based DP slaves and fail-safe GSD based I/O devices).

### Special F-parameters

For the F-functionality there are special F-parameters that you can review and set in the "Properties" of the fail-safe components (F-CPU and F-I/O). F-parameters are marked in yellow.

F-parameters are explained in "Configuring an F-CPU (Page 42)" and "Configuring F-I/O (Page 47)".

### Compiling the hardware configuration

You must compile the hardware configuration of the SIMATIC Safety F-system (shortcut menu "Compile > Hardware configuration"). A configured F-CPU with enabled F-capability is the only prerequisite for programming the safety program.

**Note**

Inconsistencies are possible when configuring the hardware and can also be saved. A full consistency check of the hardware configuration and possible connection data is performed only during compilation. Therefore, perform "Edit > Compile" regularly.

### Changing safety-related parameters

**Note**

If you change a safety-related parameter (marked in yellow) for an F-I/O or an F-CPU, you must then compile the modified hardware configuration and the Compiling the safety program (Page 265) (shortcut menu "Compile > Hardware and software (only changes)") and download. This also applies for changes to the F-I/O which are not used in the safety program. F-I/O in standard operation is not affected by this.

# 2.3 Configuring an F-CPU

## Introduction

You configure the F-CPU basically in the same way as a standard automation system.

F-CPUs are always configurable in *STEP 7*, regardless of whether or not the *STEP 7 Safety* optional package is installed. Without an installed optional package, however, the F-CPU can only be used as a standard CPU.

If the *STEP 7 Safety* optional package is installed, you can enable or disable the F-capability for the F-CPU.

If you want to use the F-I/O in safety mode or in safety-related communication, the F-capability of the F-CPU must be enabled.

F-capability is enabled by default after the *STEP 7 Safety* optional package is installed.

## Enabling/disabling F-capability

If you want to modify the F-capability setting, proceed as follows:

1. Select the F-CPU in the device or network view, and select the "Properties" tab in the inspector window.
2. Select "Fail-safe" in the area navigation.
3. Use the appropriate button to enable/disable the F-capability.
4. If you want to disable F-capability, confirm the "Disable F-activation" dialog with "Yes".

## Disabling F-capability for an existing safety program

If you want to disable the F-capability for an F-CPU because you intend to use the F-CPU as a standard CPU although a safety program is installed, you must note the following:

- You need the password for the safety program, if assigned.
- The Safety Administration Editor (Page 64) is deleted from the project tree.
- The F-OB is deleted. (S7-1200, S7-1500)
- All F-blocks in the "Program blocks" folder are marked as no longer supported in the project tree  . They can no longer be opened or compiled.

  All F-blocks within the "System blocks" folder are deleted.

  A warning is issued during the next compilation.
- From now on you cannot use F-I/O in safety mode with this F-CPU.

## Configuring the F-parameters of the F-CPU

In the "Properties" tab of the F-CPU, you can change or apply the default settings for the following parameters:

- The basis for PROFIsafe addresses

  **Note**

  A change in the basis for the PROFIsafe addresses results in modifications to the safety program when it is recompiled if it contains F-I/O of PROFIsafe address type 2. A new acceptance may therefore be required.

- The default F-monitoring time for central or distributed F-I/O at the F-CPU

  **Note**

  A change of the F-monitoring time for central or distributed F-I/O at the F-CPU results in modifications to the safety program when it is recompiled. A new acceptance may therefore be required.

## "Basis for PROFIsafe addresses" parameter for F-I/O with PROFIsafe address type 1

Setting this parameter defines a start value for the automatic assignment of F-destination addresses of all F-I/O of PROFIsafe address type 1 assigned to the F-CPU. The parameter has no effect on the F-source address of F-I/O of PROFIsafe address type 1.

The F-destination addresses for F-I/O of PROFIsafe address type 1 must be unique network-wide and CPU-wide.

By selecting different start values for different F-CPUs, you can define different ranges for the automatic assignment of the F-destination address. This is useful when you are operating multiple F-CPUs in one network. Subsequent address changes are possible.

You can specify the "Basis for PROFIsafe addresses" parameter in increments of 100. During automatic assignment of the F-destination addresses, the next free F-destination address is selected. The maximum possible F-destination address for ET 200S, ET 200eco, ET 200pro, ET 200iSP F-modules and S7-300 F-SMs is 1022.

**Example:** You set "200" as the basis. The F-destination address is then automatically assigned in ascending order from the F-destination address 200.

For F-I/O in which the "Basis for PROFIsafe addresses" is outside the F-destination address value range, a start value of 1 is used for the automatic assignment of the F-destination address.

**Note**

The "Basis for PROFIsafe addresses" parameter has no influence on the following F-I/O:

- SM 326; DI 8 x NAMUR (as of article number 6ES7326-1RF00-0AB0)
- SM 326; DO 10 x DC 24V/2A (article number 6ES7326-2BF01-0AB0)
- SM 336; AI 6 x 13 Bit (article number 6ES7336-1HE00-0AB0)

### "Basis for PROFIsafe addresses" parameter for F-I/O with PROFIsafe address type 2

The basis for PROFIsafe addresses is applied as an F-source address for F-I/O of PROFIsafe address type 2. The F-source address ("Basis for PROFIsafe addresses" parameter) must be unique network-wide. Subsequent address changes are possible.

**Note**

A change to the "Basis for PROFIsafe addresses" parameter results in modifications to the safety program when it is recompiled. A new acceptance may therefore be required because the F-source addresses of all F-I/O are changed centrally by this step.

You can specify the "Basis for PROFIsafe addresses" parameter in increments of 100.

The F-destination address of F-I/O of PROFIsafe address type 2 is assigned automatically for each F-CPU in descending order starting with 65534.

### "Default F-monitoring time" parameter

Configure the "Default F-monitoring time" for monitoring the communication between the F-CPU and F-I/O.

You can adjust the F-monitoring time via the following parameters:

- "Default F-monitoring time for central F-I/O"
- "Default F-monitoring time for F-I/O of this interface"

The **default F-monitoring time for the central F-I/O** acts on the F-I/O that is arranged centrally, i.e. near the F-CPU. You set this parameter in the properties of the F-CPU (select F-CPU, then select "Properties > Fail-safe > F-parameters").

The **default F-monitoring time for the F-I/O of this interface** acts on the F-I/O that is assigned to this interface of the F-CPU (PROFIBUS or PROFINET). You change this parameter in the properties of the relevant interface (selection of the interface in the "Device view" tab, then "F-parameters").

The various settings available allow you to flexibly adapt the F-monitoring time to the conditions of your F-system, for example to take account of different bus cycles.

You can also change the F-monitoring time individually for each F-I/O in the F-I/O properties (see Configuring F-I/O (Page 47) or Peculiarities when configuring fail-safe GSD based DP slaves and fail-safe GSD based I/O devices (Page 61)).

**Note**

A change of the F-monitoring time for central or distributed F-I/O at the F-CPU results in modifications to the safety program when it is recompiled. A new acceptance may therefore be required.

**WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal state is pending for at least as long as the assigned monitoring time. *(S018)*

You can find additional information in Monitoring and response times (Page 661).

## Automatic generation of the safety program

The safety program of an F-CPU consists of one or two F-runtime groups that contain the F-blocks (see also Defining F-Runtime Groups (Page 102)). When the F-CPU (with activated F-capability) is inserted into the work area of the device view or network view, a safety program with an F-runtime group is generated automatically.

You can define in *STEP 7 Safety* that no F-runtime group is generated while inserting the F-CPU (with activated F-capability).

Proceed as follows:

1. Select the "Options > Settings" menu command.
2. Select the "STEP 7 Safety" area.
3. If it is not already disabled , disable automatic generation of an F-runtime group by deselecting the "Generate default fail-safe program" option.

This change has no influence on any existing safety programs; it only defines whether an F-runtime group is automatically generated for each one of the subsequently inserted F-CPUs.

## Configuring the protection level of the F-CPU

**WARNING**

(S7-300, S7-400) In safety mode, access with the CPU password must not be authorized during changes to the standard user program as this would allow changes to the safety program. To rule out this possibility, you must configure the **protection level "Write protection for fail-safe blocks"** and configure a password for the F-CPU. If only **one person** is authorized to change the standard user program **and** the safety program, the protection level "Write protection" or "Read/write protection" should be configured so that other persons have only limited access or no access at all to the entire user program (standard and safety programs). (See also Access protection (Page 76)) *(S001)*

**WARNING**

(S7-1200, S7-1500) In safety mode, the safety program must be password-protected. You therefore need to configure at least the protection level "Full access (no protection)". This protection level only allows full access to the standard user program, not to F-blocks.

If you select a higher protection level, for example to protect the standard user program, you must assign an additional password for "Full access (no protection)". *(S041)*

You configure the protection level following the same procedure as for standard CPUs.

For information on the password for the F-CPU, refer to Access protection (Page 76). Pay special attention to the warnings in Setting up access permission for the F-CPU (Page 82).

# 2.4 Configuring F-I/O

## Introduction

You configure the ET 200 MP, ET 200SP, ET 200S, ET 200eco (S7-300, S7-400), ET 200pro and ET 200iSP F-modules, the S7-300 F-SMs and the S7-1200 F-modules as usual in *STEP 7*.

After you have inserted the F-I/O in the work area of the *device or network view*, you access the configuration dialogs by selecting the relevant F-I/O and the "Properties" tab.

**Note**

Changes to the parameter assignment result in modifications to the safety program when it is recompiled. A new acceptance may therefore be required.

The use of ET 200SP F-modules is possible with IM155-6PN ST as of firmware V1.1, IM155-6PN HF and IM155-6DP HF.

The distributed use of ET 200MP F-modules is possible with IM 155-5 PN ST as of firmware V3.0, IM 155-5 PN HF as of firmware V2.0 and IM 155-5 DP ST as of firmware V3.0.

The central use of ET 200MP F-modules is possible with S7-1500 F-CPUs as of firmware V1.7.

(S7-1500) When using ET 200MP F-modules you must set the version of the F-I/O access to V1.3 in the SAE under "Settings".

## Channel-granular passivation after channel faults

You can configure how the F-I/O will respond to channel faults, such as a short-circuit, overload, discrepancy error, or wire break, provided the F-I/O supports this parameter (e.g. for ET 200S or ET 200pro F-modules). You configure this response in the properties for the relevant F-I/O ("Behavior after channel fault" parameter). This parameter is used to specify whether the entire F-I/O or just the faulty channel(s) are passivated in the event of channel faults.

**Note**

(S7-300, S7-400) Note that channel-granular passivation increases the runtime of the F-runtime group(s) compared to passivation of the entire F-I/O (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).

### "Channel failure acknowledge" parameter

For ET 200MP F-modules and S7-1200 F-modules, the "Channel failure acknowledge" parameter replaces the ACK_NEC tag of the F-I/O DB.

If an F-I/O fault is detected by the F-I/O, passivation of all channels of the relevant F-I/O occurs. If channel faults are detected, the relevant channels are passivated if "Passivate channel" is configured. If "Passivate the entire module" is configured, all channels of the relevant F-I/O are passivated. Once the F-I/O fault or channel fault has been eliminated, reintegration of the relevant F-I/O occurs in line with the "Channel failure acknowledge" parameter.

- Automatically
- Manually

**⚠ WARNING**

The parameter assignment "Channel failure acknowledge = Automatic" is only allowed if automatic reintegration is permitted for the relevant process from a safety standpoint. *(S045)*

**Note**

The default assignment for the "Channel failure acknowledge" parameter when the F-module is created is "Manually".

### Organization block/Process image (S7-1200, S7-1500)

If you use F-I/O in standard mode (only for S7-1500 CPUs), you can select the organization block/process image as you do for standard I/O.

If you use F-I/O in safety mode, no selection is possible. The process image is updated at the beginning or end of the F-OB (see section Program structure of the safety program (S7-1200, S7-1500) (Page 87)).

### Changing the name and number of the F-I/O DB

The name and number of the F-I/O DB (Page 129) are assigned automatically when the F-I/O is configured. You can change the number in the properties of the F-I/O. Alternatively, you can change the name in the project tree via the shortcut menu for the F-I/O DB.

You can find information on the assignment of number ranges in *Safety Administration Editor*, "Settings" area (Page 71).

## Customizing the F-monitoring time for F-I/O

You can customize the F-monitoring time in the properties of the F-I/O under "F-parameters". This may be necessary to prevent a timeout being triggered when no error occurs and the F-I/O requires a longer F-monitoring time or assignment with a default F-monitoring time is not possible. For this purpose, activate the corresponding check box and assign an F-monitoring time.

**⚠ WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal state is pending for at least as long as the assigned monitoring time. *(S018)*

You can find additional information in Monitoring and response times (Page 661).

## Group diagnostics for fail-safe S7-300 signal modules

By disabling a channel of the fail-safe signal module in the module properties, you also disable the group diagnostics for this channel.

**Exception for S7-300/400 F-CPUs:**

For the following S7-300 fail-safe signal modules

- SM 326; DI 8 x NAMUR (as of article number 6ES7326-1RF00-0AB0)
- SM 326; DO 10 x DC 24V/2A (article number 6ES7326-2BF01-0AB0)
- SM 336; AI 6 x 13Bit (article number 6ES7336-1HE00-0AB0)

the "Group diagnostics" parameter enables and disables the monitoring of channel-specific diagnostic messages of F-SMs (such as wire break and short-circuit) to the F-CPU. You should disable group diagnostics for **unused** input or output channels.

**⚠ WARNING**

(S7-300, S7-400) For the following S7-300 fail-safe signal modules (F-SMs) with activated safety mode, "Group diagnostics" must be enabled for all connected channels:

- SM 326; DI 8 x NAMUR (as of article number 6ES7326-1RF00-0AB0)
- SM 326; DO 10 x DC 24V/2A (article number 6ES7326-2BF01-0AB0)
- SM 336; AI 6 x 13 Bit (article number 6ES7336-1HE00-0AB0)

Check to verify that you have only disabled group diagnostics for these F-SMs for input and output channels that are actually unused. *(S003)*

Diagnostic interrupts can be enabled optionally.

## Additional information

For detailed description of the **parameters**, refer to the help on the properties of the respective F-I/O and in the respective *manual for the F-I/O*.

# 2.5 Configurations supported by the SIMATIC Safety F-system

## Supported configurations

F-I/Os (see Overview of Configuration (Page 37)) are supported in the following configurations:

The F-I/O is not addressed by means of I-slave-slave communication and can be located in one of the following configurations:

- Centrally in the F-CPU (including expansion rack)
- At the PROFIBUS DP (integrated DP interface of the F-CPU or via PROFIBUS CP/ PROFIBUS CM)
- At the PROFINET IO (integrated PN interface of the F-CPU or via PROFINET CP/ PROFINET CM)
- Downstream of an IE/PB-Link at the PROFIBUS DP (integrated PN interface of the F-CPU or via PROFINET CP/PROFINET CM)

The F-I/O is addressed by means of I-slave-slave communication and can be located in one of the following configurations:

- At the PROFIBUS DP (integrated DP interface of the (F)-CPU* or via PROFIBUS CP/ PROFIBUS CM)
- Downstream of an IE/PB-Link at the PROFIBUS DP (integrated PN interface of the (F)-CPU* or via PROFINET CP/PROFINET CM)

* The IO controller of the IE/PB-Link or the DP master can be a standard CPU or an F-CPU.

For F-I/O not listed in Overview of Configuration (Page 37), please check the relevant documentation to see whether or not it is supported by the SIMATIC Safety F-system. If in doubt, treat these F-I/Os as part of a configuration that is not supported.

## Checks performed by the SIMATIC Safety F-system

For supported configuration, the F-system checks:

- Correctness of the PROFIsafe operating mode parameter (F_Par_Version)
- CPU-wide unique assignment of the F-destination address
  You yourself must ensure the network-wide uniqueness of the PROFIsafe address.
- Correct assignment of the F-source address according to the "Basis for PROFIsafe addresses" parameter of the F-CPU for F-I/O of PROFIsafe address type 2.

**WARNING**

Please note the following with configurations that are not included in supported configurations:

- Make sure that the F-I/O appears in the safety summary and that an F-I/O DB has been created for it. Otherwise you cannot use the F-I/O in this configuration. (Contact Industry Online Support.)
- For F-I/Os in the PROFINET IO environment**, you must check the PROFIsafe operating mode parameter (F_Par_Version) against the safety summary to make sure that it is correct. V2 mode must be set in the PROFINET IO environment. F-I/O which only support V1 mode may not be used in the PROFINET IO environment.
- You must ensure that PROFIsafe address assignment is unique CPU-wide* and network-wide***:
  - Use the safety summary to check that the F-source address corresponds to the "Basis for PROFIsafe addresses" parameter of the F-CPU for F-I/O of PROFIsafe address type 2.
  - For F-I/O of PROFIsafe address type 1, or if you cannot set the F-source address in accordance with the "Basis for PROFIsafe addresses" parameter of the F-CPU, you will have to ensure the uniqueness of the PROFIsafe address solely by assigning a unique F-destination address.

  You must check the uniqueness of the F-destination address individually for each F-I/O based on the safety summary in a configuration that is not supported. (see Completeness and correctness of the hardware configuration (Page 312)) *(S050)*

* "CPU-wide" means all F-I/Os assigned to an F-CPU: central F-I/O of this F-CPU as well as F-I/Os for which the F-CPU is DP master/IO controller. An F-I/O that is addressed using I-slave-slave communication is assigned to the F-CPU of the I-slave and not to the F-CPU of the DP master / IO controller.

** The F-I/O is located in the "PROFINET IO environment" if at least part of safety-related communication with the F-CPU takes place via PROFINET IO. If the F-I/O is connected via I-slave-slave communication, also keep in mind the communication line to the DP master/ IO controller.

*** A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

# 2.6 PROFIsafe addresses for F-I/O of PROFIsafe address type 1

## F-destination address

The uniqueness of the PROFIsafe address is ensured solely with the F-destination address. The F-source address has no effect on whether or not the PROFIsafe address is unique.

Therefore, the F-destination address must be unique network-wide and CPU-wide (see the following rules for address assignment).

To prevent incorrect parameter assignment, an F-destination address which is unique CPU-wide is automatically assigned during placement of the F-I/O in the work area of the device or network view as long as you only configure supported configurations (Page 50).

To ensure a network-wide unique F-destination address assignment when multiple DP master systems and PROFINET IO systems are operated on one network, you must set the "Basis for PROFIsafe addresses" parameter (in the properties of the F-CPU) in SIMATIC Safety F-systems appropriately, before placing the F-I/O (see section "Recommendations for address assignment").

When you change the F-destination address of an F-I/O, the CPU-wide uniqueness of the F-destination address is checked automatically for supported configurations. You yourself must ensure the network-wide uniqueness of the F-destination address.

For ET 200S, ET 200eco (S7-300, S7-400), ET 200pro, and ET 200iSP F-modules and S7-300 F-SMs:
You must set the F-destination address at the F-I/O with the DIP switch before you install the F-I/O. You can assign up to 1022 different F-destination addresses.

**Note**

(S7-300, S7-400) For the following fail-safe S7-300 signal modules, the F-destination address is the start address of the F-SM divided by 8:

- SM 326; DI 8 x NAMUR (as of article number 6ES7326-1RF00-0AB0)
- SM 326; DO 10 x DC 24V/2A (article number 6ES7326-2BF01-0AB0)
- SM 336; AI 6 x 13 Bit (article number 6ES7336-1HE00-0AB0)

The next free start address and F-destination address is assigned for these F-SMs beginning with F-destination address 1.

You can show the columns "F-source address" and "F-destination address" in the device view of the device overview. The addresses displayed in these columns are for information purposes only. You have to check the F-source and F-destination addresses in the safety summary when you accept the system.

## Rules for address assignment

**⚠ WARNING**

F-I/Os of PROFIsafe address type 1 are uniquely addressed by their F-destination address (e.g. with the switch setting on the address switch).

The following rules ensure the uniqueness of the F-destination addresses.

The F-destination address (and therefore also the switch setting on the address switch) of the F-I/O must be unique network-wide* and CPU-wide** (system-wide) **for the entire** F-I/O. The F-I/O of PROFIsafe address type 2 must also be considered. *(S051)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

** "CPU-wide" means all F-I/Os assigned to an F-CPU: central F-I/O of this F-CPU as well as F-I/Os for which the F-CPU is DP master/IO controller. An F-I/O that is addressed using I-slave-slave communication is assigned to the F-CPU of the I-slave and not to the F-CPU of the DP master / IO controller.

Please also note Recommendation for PROFIsafe address assignment (Page 56).

## See also

Completeness of the safety summary (Page 310)

# 2.7 PROFIsafe addresses for F-I/O of PROFIsafe address type 2

## F-source address and F-destination address

The uniqueness of the PROFIsafe address is ensured by the combination of F-source address and F-destination address.

The PROFIsafe address must be unique network-wide and CPU-wide. This is the case if the following two conditions are met:

- The F-source address ("Basis for PROFIsafe addresses" parameter) of the F-CPU is unique network-wide.
- The F-destination address of the F-module is unique CPU-wide.

You define the F-source address using the "Basis for PROFIsafe addresses" parameter in the F-CPU. Provided you only configure supported configurations (Page 50), this parameter is automatically applied as the F-source address and a CPU-wide unique F-destination address is assigned (usually in descending order starting with 65534).

Please note that the F-source address must be unique network-wide, especially when the "Basis for PROFIsafe addresses" parameter is changed. When you change the F-destination address, the CPU-wide uniqueness of the F-destination address is checked automatically for supported configurations.

You must assign the F-source address and F-destination address to the F-I/O before you commission the F-I/O. You can find additional information in Assigning F-destination addresses to fail-safe modules with SIMATIC Safety (Page 57).

You can show the columns "F-source address" and "F-destination address" in the device view of the device overview. The addresses displayed in these columns are for information purposes only. You have to check the F-source and F-destination addresses in the safety summary when you accept the system.

## Rules for address assignment

**⚠ WARNING**

F-I/O of PROFIsafe address type 2 is uniquely addressed using a combination of F-source address ("Basis for PROFIsafe addresses of the assigned F-CPU" parameter) and F-destination address.

The combination of F-source address and F-destination address for each F-I/O must be unique network-wide* and CPU-wide** (system-wide). In addition, the F-destination address may not be occupied by F-I/O of PROFIsafe address type 1.

To ensure that addresses are unique across F-CPUs for supported configurations (Page 50), you need to ensure that the "Basis for PROFIsafe addresses" parameter of all F-CPUs is unique network-wide*. This is achieved through different settings for the "Basis for PROFIsafe addresses" parameter of the F-CPUs. *(S052)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

** "CPU-wide" means all F-I/Os assigned to an F-CPU: central F-I/O of this F-CPU as well as F-I/Os for which the F-CPU is DP master/IO controller. An F-I/O that is addressed using I-slave-slave communication is assigned to the F-CPU of the I-slave and not to the F-CPU of the DP master / IO controller.

Please also note Recommendation for PROFIsafe address assignment (Page 56).

## See also

Completeness of the safety summary (Page 310)

## 2.8 Recommendation for PROFIsafe address assignment

Before inserting the F-I/O, specify an address range for each F-CPU for the F-destination addresses of the F-I/O of PROFIsafe address type 1 that does not overlap with the address range of any other F-CPU network-wide or CPU-wide (system-wide). You specify the beginning of this range for F-I/O of PROFIsafe address type 1 with the "Basis for PROFIsafe addresses" parameter.

The F-destination addresses of F-I/O of PROFIsafe address type 2 must not overlap with any address range of the F-I/O of PROFIsafe address type 1. The ranges of the F-destination addresses of the F-I/O of PROFIsafe address type 2 may overlap if the F-source addresses are different. This is the case for supported configurations (Page 50) if the "Basis for PROFIsafe addresses" parameter has been set differently for each F-CPU.

Assign relatively low F-destination addresses for F-I/O of PROFIsafe address type 1 and relatively high F-destination addresses for F-I/O of PROFIsafe address type 2.

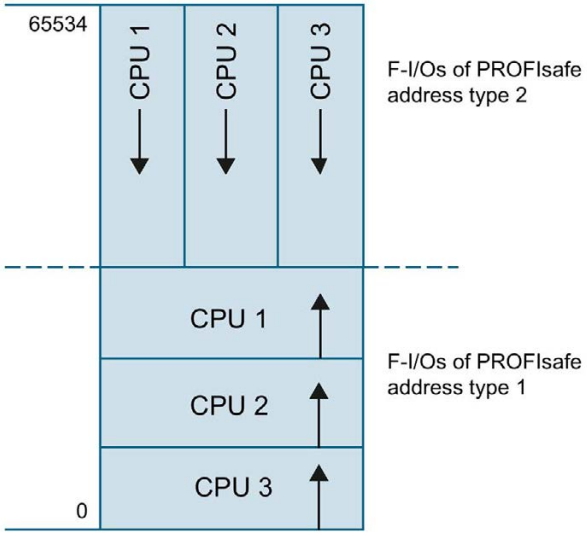



Figure 2-1 Address assignment for F-I/O of PROFIsafe address types 1 and 2

The safety summary (Page 310) lists the following information for each F-CPU:

- "Basis for PROFIsafe addresses" parameter (F-source address for F-I/O of PROFIsafe address type 2)
- Range of the F-destination addresses of the assigned F-I/O of PROFIsafe address type 1
- Range of the F-destination addresses of the assigned F-I/O of PROFIsafe address type 2

Any F-I/O configured using I-slave-slave communication is listed in the safety summary as part of the F-destination address range of the I-slave.

# 2.9 Assigning F-destination addresses to fail-safe modules with SIMATIC Safety

## Introduction

Fail-safe ET 200SP modules, fail-safe ET 200MP modules and fail-safe S7-1200 modules do not have a DIP switch that allows you to assign the unique F-destination address for each module. Instead, you assign the PROFIsafe address (Page 50) directly from *STEP 7 Safety* for fail-safe ET 200SP modules and fail-safe ET 200MP modules. The PROFIsafe addresses for S7-1200 F-modules are automatically assigned during download of the hardware configuration.

You can manually change the F-destination addresses set by the F system in the hardware configuration.

You configure the F-destination address in the hardware configuration for each F-module. The F-source address for supported configurations (Page 50) corresponds to the "Basis for PROFIsafe addresses" parameter of the corresponding F-CPU.

In the following cases it is necessary to reassign the addresses of the fail-safe ET 200SP and fail-safe ET 200MP modules:

- Later placement of a fail-safe module during initial commissioning
- Intentional modification of the F-destination address
- Modification of the "Basis for PROFIsafe addresses" parameter of the associated F-CPU (changes the F-source address).
- Replacement of the coding element
- Commissioning of a mass-produced machine

In the following cases it is not necessary to reassign the addresses of the fail-safe ET 200SP and fail-safe ET 200MP modules:

- Power On/Off
- Replacement of an F-module (repair) without PG/PC
- Replacement of the BaseUnit
- Changes in the design in case a new BaseUnit is inserted in front of a fail-safe module
- Repair/replacement of the interface module

### Basic procedure

**Note**

**Assigning the PROFIsafe address for S7-1200 fail-safe modules**

The procedure described below for identifying and assigning the PROFIsafe addresses is not required for S7-1200 fail-safe modules.

Note that an S7-1200 F-CPU must not include an additional unconfigured F-module.

1. Configure the F-destination address (Page 54) and F-source address in the hardware configuration in *STEP 7 Safety*.
2. Identify the ET 200SP or ET 200MP F-modules to which you want to assign the configured F-destination addresses (together with the F-source address).
3. Assign the F-destination address (together with the F-source address) to the F-modules.

## 2.9.1 Identifying F-modules

### Requirement

The following requirements must be met:

- The F-CPU and fail-safe modules are configured.
- The configuration has been downloaded.
- The F-CPU and fail-safe modules can be reached online.

**⚠ WARNING**

Clicking "Identification" confirms the fail-safe correctness of the PROFIsafe addresses for the fail-safe modules.

Therefore, proceed carefully when confirming the fail-safe modules by LED flashing or the serial number of the F-CPU with fail-safe modules or the interface module with fail-safe modules. *(S046)*

## Procedure

Proceed as follows to identify the F-modules:

1. Establish an online connection to the F-CPU with which the fail-safe modules are operated.
2. In the network view, select the F-CPU with fail-safe modules or the interface module with the fail-safe modules to which you want to assign the F-destination address (together with the F-source address).
3. Select "Assign F-destination address" from the shortcut menu.
4. Under "Assign F-destination address by", select the method to be used for identifying the F-modules.
   - "Identify by LED flashing"

     This is the default setting. The DIAG and STATUS LED of the F-modules to be identified flash upon identification.
   - "By serial number"

     If you cannot see the fail-safe modules directly, you can identify the fail-safe modules by the serial number of the F-CPU or interface module.

     **Note**

     The displayed serial number may be amended with a year number compared to the serial number printed on the interface module. The serial numbers are nevertheless identical.
5. In the "Assign" column, select all the fail-safe modules to which you want to assign the F-destination address (together with the F-source address).

   If you select the F-CPU or the interface module in the "Assign" column, all F-modules of the station are selected.
6. Click the "Identification" button. Check whether the DIAG and status LED for the F-modules whose F-destination address you want to assign are flashing green. If you identify using the serial number, compare the displayed serial number to the serial number of the F-CPU with the fail-safe modules or the interface module with fail-safe modules.
7. If you have configured more ET 200MP fail-safe modules than exist online, a dialog is displayed. Enter the number of ET 200MP fail-safe modules actually existing in this dialog and confirm the dialog.

   If you have configured fewer ET 200MP fail-safe modules than exist online, the online-offline difference is shown and the assignment of the F-destination addresses is not possible.

## 2.9.2 Assigning a fail-safe destination address

### Requirement

The F-modules have been successfully identified.

### Procedure

To assign an F-destination address, proceed as follows:

1. Use the "Assign F-destination address" button to assign the F-destination address (together with the F-source address) to the fail-safe modules. You may have to enter the password of the F-CPU.

   To assign the F-destination address (together with the F-source address), the "Confirm assignment" dialog must be confirmed within 60 seconds.

## 2.9.3 Changing the F-destination address and the F-source address

### Changing the F-destination address or F-source address

1. Change the F-destination address or F-source address in the hardware configuration.
2. Compile the hardware configuration.
3. Download the hardware configuration to the F-CPU.
4. Select "Assign F-destination address" from the shortcut menu.
5. Proceed as described under Identifying F-modules (Page 58) and Assigning a fail-safe destination address (Page 60).

# 2.10 Peculiarities when configuring fail-safe GSD based DP slaves and fail-safe GSD based I/O devices

## Requirement

In order to use fail-safe GSD based DP slaves for SIMATIC Safety, these GSD based slaves must be operated on PROFIBUS DP and support the PROFIsafe bus profile. In order to use an F-CPU S7-1500, the GSD based DP slaves have to support the PROFIsafe bus profile in V2 mode.

Fail-safe GSD based DP slaves used in hybrid configurations on PROFIBUS DP and PROFINET IO downstream from a IE/PB link must support the PROFIsafe bus profile in V2 mode.

In order to use fail-safe GSD based I/O devices for SIMATIC Safety, the GSD based devices must be operated on PROFINET IO and support the PROFIsafe bus profile in V2 mode.

## Configuration with GSD files

As is the case in a standard system, the basis for configuring fail-safe GSD based DP slaves/GSD based I/O devices is the device specification in the GSD file (generic station description file).

A GSD file contains all of the properties of a GSD based DP slave or GSD based I/O device. For fail-safe GSD based DP slaves/GSD based I/O devices, certain parts are protected by a CRC.

The GSD files are supplied by the device manufacturers.

## Protection of the data structure of the device in GSD files

The only GSD files supported are those that satisfy the requirements for protection defined as of *PROFIsafe Specification* V2.0 using a CRC stored in this file ("desired value" for F_IO_StructureDescCRC).

The data structure described in the GSD file is checked when the F-I/O is added to the hardware configuration and when the hardware configuration is compiled. If an error is detected, you should clarify whether the GSD file provided by the device manufacturer contains the desired value for F_IO_StructureDescCRC.

## Assigning the PROFIsafe address

**⚠ WARNING**

Check the documentation for your fail-safe GSD based DP slaves / GSD based I/O devices to find out the valid PROFIsafe address type. If you do not find the necessary information, assume PROFIsafe address type 1. Proceed as described under PROFIsafe addresses for F-I/O of PROFIsafe address type 1 (Page 52) or PROFIsafe addresses for F-I/O of PROFIsafe address type 2 (Page 54).

Set the PROFIsafe source address for fail-safe GSD based DP slaves / GSD based I/O devices according to the manufacturer's specifications. If the F-source address needs to correspond to the "Basis for PROFIsafe addresses" parameter of the F-CPU (PROFIsafe address type 2), you will find the latter in the "Properties" tab of the F-CPU. In this case, also check in the safety summary that the value of the F-CPU for the "Basis for PROFIsafe addresses" parameter matches the value of the F-source address of the fail-safe GSD based DP slave / fail-safe GSD based I/O device. *(S053)*

## Configuration procedure with GSD files

You import the GSD files to your project (see *Help on STEP 7 "GSD files"*).

1. Select the fail-safe GSD based DP slave / GSD based I/O device in the "Hardware catalog" task card and connect it to the relevant subnet in the network view.
2. Select the fail-safe GSD based DP slave/GSD based I/O device and insert the necessary F-modules, if this does not occur automatically.
3. Select the relevant F-module and open the "Properties" tab in the inspector window.

For fail-safe GSD based DP slaves/GSD based I/O devices (contrary to other F-I/O), the "Manual assignment of the F-monitoring time" parameter is enabled. The result is that the value specified for the F-monitoring time is used as default value when the slaves/devices are plugged. You can change both values (time and type of assignment) later manually.

## F-parameter "F_CRC_Seed" and "F_Passivation" for fail-safe GSD based I/O devices

The F-parameters "F_CRC_Seed" and "F_Passivation" influence the behavior of a fail-safe GSD based I/O device. The combination of the F-parameters cannot be set but is specified by selecting a corresponding F-module. Up to three F-module variants can be used, depending on the S7-300/400 or S7-1500 F-CPU used.

| F-module variant | F_CRC_Seed | F_Passivation | Behavior of the fail-safe GSD based I/O device | Can be used with F-CPU |
|---|---|---|---|---|
| 1 | Parameter does not exist | Parameter does not exist | The GSD based I/O device works with the Basic Protocol (BP) from PROFIsafe.<br>The RIOforFA-Safety profile is not supported. | S7-300/400/1500* |
| 2 | CRC-Seed24/32 | Device/module | The GSD based I/O device works with the Expanded Protocol (XP) from PROFIsafe.<br>The RIOforFA-Safety profile is not supported. | S7-1500 |
| 3 | CRC-Seed24/32 | Channel | The GSD based I/O device works with the Expanded Protocol (XP) from PROFIsafe.<br>The RIOforFA-Safety profile is supported. | S7-1500 |

* Only use the F-module variant 1 with S7-1500 F-CPUs if neither F-module variant 2 nor 3 exists.

## Additional information

You can find the description of the parameters in the Help on fail-safe GSD based DP slaves and GSD based I/O devices.

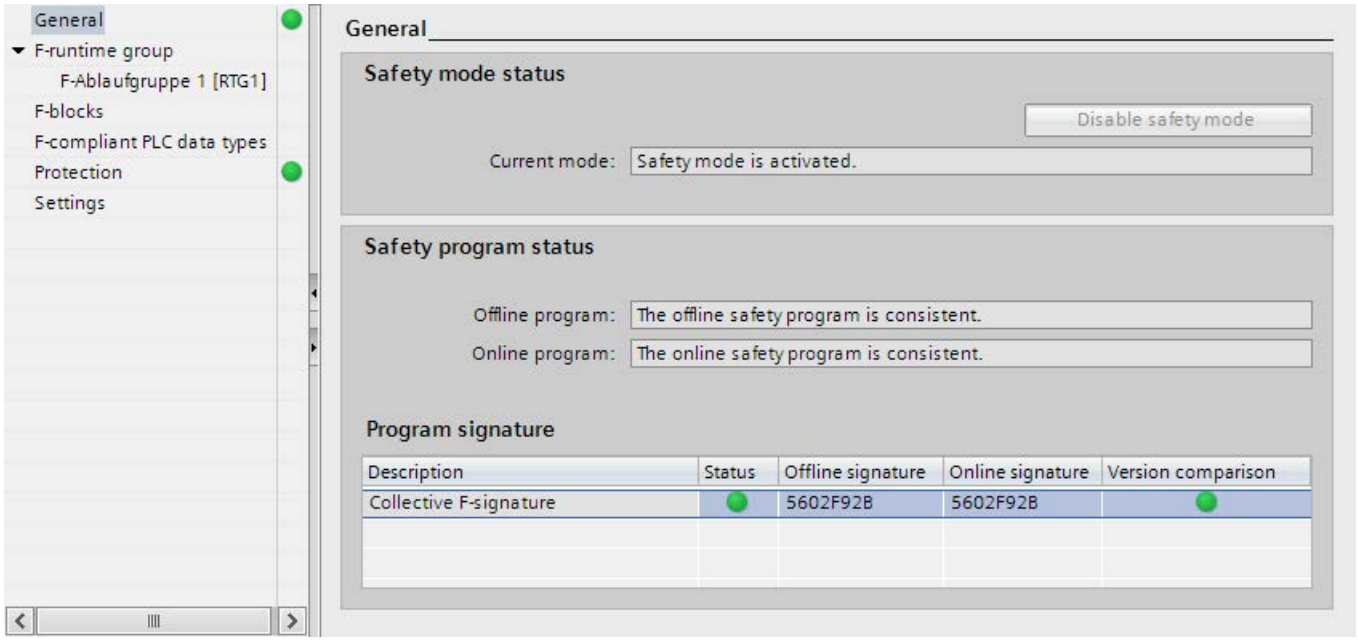
# Safety Administration Editor

# 3

## Overview

The *Safety Administration Editor* supports you in the following tasks:

- Displaying of status of the safety program
- Displaying of collective F-signature
- Displaying of status of safety mode
- Creating and organizing of F-runtime groups
- Displaying information on the F-blocks
- Displaying information about F-compliant PLC data types
- Specifying/changing access protection
- Specifying/changing general settings for the safety program
- Set/modify advanced settings for the safety program, e.g. Enable F-change history



The *Safety Administration Editor* is divided into the following areas:

- General

  Under "General", the status of safety mode, the safety program, and the collective F-signature are displayed to you. Additional information on the "General" area can be obtained in ""General" area (Page 67)".

- F-runtime groups

  A safety program consists of one or two F-runtime groups. Under "F-runtime groups", you determine the blocks and properties of an F-runtime group.

  General information on F-runtime groups can be found in "Program structure of the safety program (S7-1200, S7-1500) (Page 87)" and "Program structure of the safety program (S7-300, S7-400) (Page 85)".

  Information on the creation of F-runtime groups can be obtained under "Defining F-Runtime Groups (Page 102)".

- F-blocks

  Under "F-blocks", you can find information on the F-blocks used in your safety program and their properties . Additional information on the "F-blocks" area can be obtained in " "F-blocks" area (Page 69)".

- F-compliant PLC data types

  Under "F-compliant PLC data types", you obtain information on the created F-compliant PLC data types. There you also obtain information whether or not an F-compliant PLC data type is used in the safety program. Additional information on "F-compliant PLC data types" can be found in ""F-compliant PLC data types" area (S7-1200, S7-1500) (Page 70)".

- Access protection

  Under "Access protection", you can set up, change, or revoke the password for the safety program. Access protection is mandatory for productive operation. Additional information on access protection can be found in "Setting up, changing and revoking access permission for the safety program (Page 79)"

- Settings

  Under "Settings", you set the parameters for the safety program. Information on the settings for your safety program can be found in ""Settings" area (Page 71)".

# 3.1 Opening the Safety Administration Editor

## Requirement

The *Safety Administration Editor* is visible as an element in the project tree, if you have configured a CPU as an F-CPU in the project, which means the "F-capability activated" option must be selected (in the properties of the F-CPU).

## Procedure

To open the *Safety Administration Editor*, follow these steps:

1. Open the folder for your F-CPU in the project tree.
2. Double-click on "Safety administration" or right-click and select the corresponding shortcut menu for the *Safety Administration Editor*.

## Result

The *Safety Administration Editor* for your F-CPU opens in the work area.

# 3.2 "General" area

## "Safety mode status"

The "Safety mode status" shows the current status of safety mode. The prerequisite is an existing online connection to the selected F-CPU.

The following statuses are possible:

- "Safety mode is activated"

  Safety mode for the selected F-CPU is activated.
- "The safety mode is not activated"

  The safety mode for the selected F-CPU is not activated.
- "F-CPU is in STOP"

  The selected F-CPU is in STOP mode.
- "F-CPU is in RUN but no safety program exists"
- "No active F-CPU available"
- "Unknown"

  The safety mode status for the selected F-CPU could not be determined.
- "F-runtime group was not called"
- "The safety program is not called"
- "Safety program status changing"
- "(No online connection)"

  An online connection to the selected F-CPU does not exist.

## "Disable safety mode"

For existing online connection and active safety mode operation, you have the option of using the "Disable safety mode" button to disable safety mode for the selected F-CPU. Safety mode can be deactivated only for the entire safety program and not for individual F-runtime groups.

**Requirement:** "Safety mode can be disabled" is selected in the "Settings" area.

Proceed as follows:

1. Click the "Disable safety mode" button.
2. Enter the password for the safety program in the dialog window and confirm with "OK".
3. A corresponding dialog will display the collective F-signature.
4. Using the displayed collective F-signature, verify that you have selected the desired F-CPU.
5. If you have selected the correct F-CPU, confirm the dialog with "Yes".

**Result:**Safety mode for the selected F-CPU is deactivated.

## "Safety program status"

"Safety program status" displays the current status of your online and offline program.

The following statuses are possible:

- Consistent (with information if no password has been assigned.)
- Inconsistent
- Modified

If no connection to the online program could been established, the message "(no online connection) " will be shown.

## "Program signature"

### For a non-existing online connection

"Program signature" displays the collective F-signature offline and the "Time stamp" displays the time of the last compilation process.

### For an existing online connection

For an existing online connection, the "Program signature" shows the following:

- The status of safety program
- The online and offline collective F-signatures
- Information on whether the block versions and the versions from "System library elements used in safety program " match online and offline.

The meaning of the symbols in the "Status" column are described in the following table.

| Status | Meaning |
|---|---|
|  | The online and offline collective F-signatures match, and a password was assigned for the online and offline safety programs. |
|  | The online and offline collective F-signatures do *not* match or no password was assigned for a safety program. |
| — | The safety program status could not be determined. |

## See also

Disabling safety mode (Page 293)

# 3.3 "F-blocks" area

## Overview

The "F-Blocks" area helps you in the following tasks:

- Displaying the F-blocks used in your safety programs.
- Displaying the F-blocks used in the F-runtime groups.
- Displaying additional information about the F-blocks.

A description of the F-blocks is available in "Creating F-blocks in FBD / LAD (Page 118)".

## Displayed information

The following information is displayed for F-blocks in offline mode:

- Has the F-block been compiled and used?
- Function of F-block in the safety program
- Offline signature
- Time stamp of the last change

The following information is displayed for F-blocks in online mode:

- Status (whether block has the same time stamp online and offline)
- Function of F-block in the safety program
- Offline signature
- Online signature

The F-blocks are hierarchically displayed as in the "Program blocks" folder.

The description of the symbols in the "Status" column can be found in "Comparing Safety Programs (Page 286)".

**Note**

During the offline-online comparison, the comparison statuses may occasionally differ between the *comparison editor* and status display in the *Safety Administration Editor*. The decisive status is the result of the comparison in the *comparison editor*, since this is the only comparison that takes into account the contents of the F-blocks.

### Filter function


Using the filter function, you can select whether you want to view all F-blocks of a certain F-runtime group or the entire safety program.

- Select "All F-blocks " from the drop-down list to view all F-blocks.
- Select an F-runtime group from the drop-down list to see all F-blocks of this F-runtime group.

## 3.4 "F-compliant PLC data types" area (S7-1200, S7-1500)

### Overview

Under "F-compliant PLC Data Types" you obtain information on the F-compliant PLC data types you have defined.

You can delete F-compliant PLC data types from the shortcut menu.

A description of F-compliant PLC data types is available in "F-compliant PLC data types (S7-1200, S7-1500) (Page 97)".

### Displayed information

The following information is displayed for F-compliant PLC data types in offline mode:

- Is the F-compliant PLC data type used in the safety program?
- Time stamp of the last change.

The following information is displayed for F-compliant PLC data types in online mode:

- Status (whether the F-compliant PLC data types have the same time stamp online and offline)

The F-compliant PLC data types are displayed hierarchically as in the folder "PLC Data Types".

Double-click the F-compliant PLC data type to open it for editing.

The description of the symbols in the "Status" column can be found in "Comparing Safety Programs (Page 286)".

**Note**

During offline-online comparison, the comparison statuses between the *comparison editor* and status display in the *Safety Administration Editor* can be different under certain circumstances. The comparison result in the *comparison editor* is decisive, since this is the only comparison that takes into account the contents of the F-compliant PLC data types.

# 3.5 "Settings" area

### "Number ranges of the F-system blocks"

The number ranges assigned here are used by the F-System for new, automatically generated F-blocks.

At this point, you can select whether the number ranges are managed by the F-system or if a fixed range specified by you is used.

- "System managed"

  The number ranges are managed automatically by the F-system, depending on the F-CPU used. The F-system selects an available number range. The start and end ranges of the number ranges are displayed.

- "Fixed range"

  You can select the start and end ranges of the number ranges from the available range. The available range depends on the F-CPU used.

  An invalid number range selection is indicated by an error message.

  The only check performed during configuration is whether the configured low limit is less than or equal to the high limit and within the available range of the F-CPU. The check as to whether the configured range is sufficient is first made during compiling. You need to ensure a sufficiently large range. Where the available range is insufficient, a compiling error occurs. Not all blocks are generated and the safety program is not executable.

Changes will become valid only during the next compilation. The automatically created F-blocks may be moved into the new area during compilation. The F-I/O DBs are an exception. They always retain their original number that you may change in the properties of the F-I/O.

## "System library elements used in safety program"

Usually, you do not need to make any settings for these parameters.

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

- Instructions (without their own version)

  Specifies the functionality of instructions (without their own version).

  A number of versions are available:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | These versions have identical functions. |
| 1.2 | x | — | x | |
| 1.3 | x | x | x | |

- F-system blocks

  Specifies the version of the F-system blocks to be used.

  A number of versions are available:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | These versions have identical functions. |
| 1.2 | x | — | x | |
| 1.3 | x | x | x | This version has identical functions to version 1.2. In addition, the current and the longest runtime of the F-runtime group are available in the F-runtime group information DB for S7-1200/1500 F-CPUs. |

- F-I/O access

  Specifies the version of the access procedure for the F-I/O.

  A number of versions are available:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | These versions have identical functions<br>For S7-1200/1500 F-CPUs, you evaluate the value status using the process image of the inputs. For S7-300/400 F-CPUs, you evaluate the QBAD_I_xx and QBAD_O_xx tags in the F-I/O DB. |
| 1.2 | x | — | x | |
| 1.3 | x | x | x | This version also supports the PROFIsafe Expanded Protocol (XP) and the RIOforFA profile for S7-1200/1500 F-CPUs. |

## "Local data used in safety program" (S7-300, S7-400)

You use this parameter to specify the amount of temporary local data (in bytes) that is available for the call hierarchy below the main safety block.

The setting applies to each F-runtime group of a safety program. Additional information on F-runtime groups can be found in Program structure of the safety program (S7-1200, S7-1500) (Page 87) and "Program structure of the safety program (S7-300, S7-400) (Page 85)".

The **minimum possible amount** is determined by the local data requirement of the F-blocks generated automatically when the safety program is compiled.

For this reason, you must provide at least 440 bytes. However, the local data requirement for the automatically added F-blocks may be higher depending on the local data requirement of the F-blocks you created with FBD or LAD.

Therefore, provide as much local data as possible. If there is not enough local data available for the automatically added F-blocks (440 bytes or more), the safety program will be compiled nevertheless.

Data in automatically added F-DBs are then used instead of local data. However, this increases the runtime of the F-runtime group(s). You will receive a notice when the automatically added F-blocks require more local data than configured.

**⚠ WARNING**

The calculated maximum runtime of the F-runtime group using the Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100) is no longer correct in this case because the calculation assumes sufficient availability of F-local data.

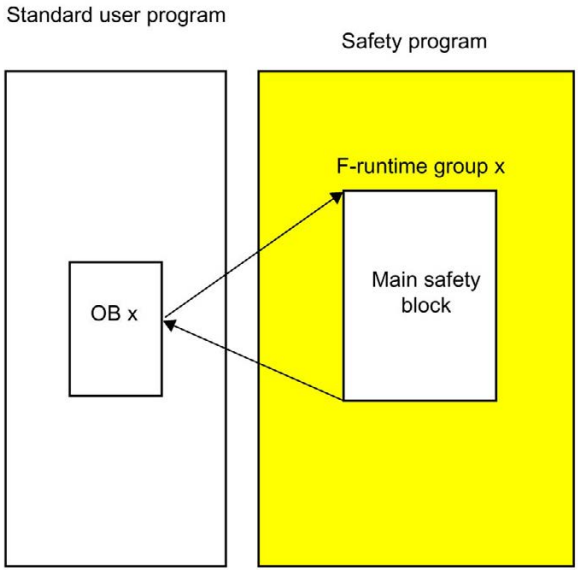
In this case, use the value you configured for the maximum cycle time of the F-runtime group (F-monitoring time) as the maximum runtime of the F-runtime group when calculating the maximum response times in the event of an error and for any runtimes of the standard system using the above-mentioned Excel file. *(S004)*

The maximum possible amount depends on:

- Local data requirement of the main safety block and the higher-level standard user program. For this reason, you should call the main safety blocks directly in OBs (cyclic interrupt OBs, whenever possible), and additional local data should not be declared in these cyclic interrupt OBs.
- Maximum volume of local data of the utilized F-CPU (see Technical Specifications in the product information for the utilized F-CPU). For S7-400 F-CPUs, you can configure the local data for each priority class. Therefore, assign the largest possible local data volume for the priority classes in which the safety program (the main safety blocks) will be called (e.g., OB 35).

### Maximum possible amount of local data as a function of local data requirement of main safety block and higher-level standard user program (S7-300, S7-400):

#### Case 1: Main safety block called directly from OBs

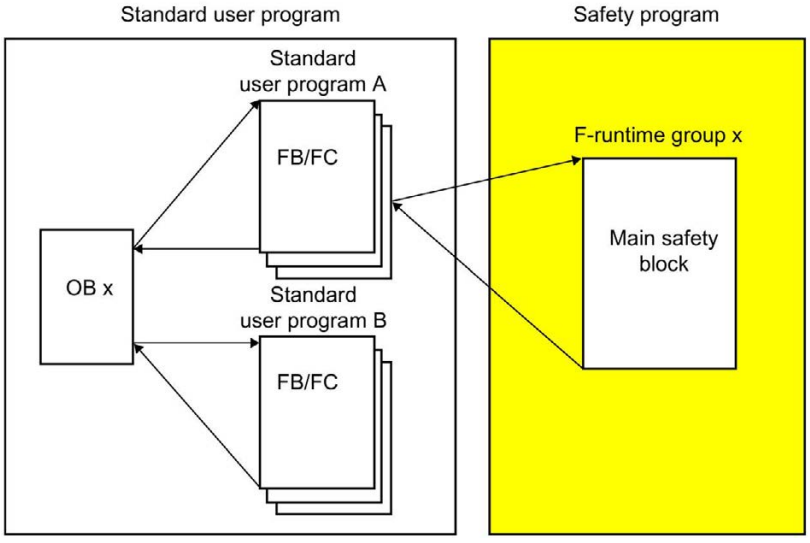



Set the "Local data used in safety program" parameter to the maximum amount of local data of the utilized F-CPU minus the local data requirement of the main safety block (if the main safety block has 2 F-runtime groups, use the largest local data requirement ) and minus the local data requirement of the calling OBx (if there are 2 F-runtime groups, use the OB with the largest local data requirement).

**Note:** If you have not declared any temporary local data in the main safety blocks and calling OBx, the local data requirement of the main safety blocks is 6 bytes and the local data requirement of the calling OBx is 26 bytes. You can derive the local data requirement of the main safety blocks and calling OBx from the program structure.

Select the utilized F-CPU in the project tree and then "Tools > Call structure". The table gives the local data requirement in the path or for the individual blocks (see also the help on *STEP 7*).

**Case 2: Main safety block not called directly from OBs**




Set the "Local data settings" parameter to the value calculated for Case 1, minus the local data requirement of standard user program A (if standard user program A has 2 F-runtime groups, use the largest local data requirement).

**Note:** You can derive the local data requirement of the standard user program A from the program structure.

Select the utilized F-CPU in the project tree and then "Tools > Call structure". The table gives the local data requirement in the path or for the individual blocks (see also the help on *STEP 7*).

## "Advanced settings"

If you deselect the "Safety mode can be disabled" option, you can prevent the disabling of safety mode for a safety program.

When you change the setting for this option, you need to recompile the safety program and download it to the F-CPU for the change to become effective. This changes the collective F-signature of your safety program.

We recommend that you disable this option before you start production and before acceptance of the safety program to prevent an unintentional disabling of the safety mode.

Use the option "Deactivate the online-offline comparison status to accelerate the online connection.

If you select this option, the results of the online-offline comparison are no longer displayed in the project tree next to "Safety Administration". The online-offline comparison in the SAE under "Safety program status" is not affected by this option.

Enable the logging of changes to the safety program by using the "Enable F-change history" option. For more information, refer to the section "F-change history (Page 307)".

# Access protection

# 4

## Access protection is necessary for productive operation

Access protection to the SIMATIC Safety F-system is mandatory for productive operation.

For test purposes, commissioning, etc. no access protection is necessary at first. This means that you can execute all offline and online actions without access protection, i.e., without password prompt.

**⚠ WARNING**

Access to the SIMATIC Safety F-system without access protection is intended for test purposes, commissioning, etc., when the system is not in productive operation mode. You must guarantee the safety of the system through other organizational measures, for example, restricted access to certain areas.

**Before you transition into productive operation mode, you must have set up and activated access protection.** *(S005)*

**Note**

(S7-300/400) If you also want to download an F-CPU with *STEP 7 Distributed Safety*, you must specify the password for the F-CPU because assignment of a password is mandatory for the F-CPU in *STEP 7 Distributed Safety* .

# 4.1 Overview of Access Protection

## Introduction

You can protect access to the SIMATIC Safety F-system by two password prompts: one for the safety program and another for the F-CPU.

## Password for the safety program

The password for the safety program is available in two forms:

- The offline password is part of the safety program in the offline project on the programming device or PC.
- The online password is part of the safety program in the F-CPU.

## Password for the F-CPU

The access protection is set at the F-CPU level. This password is also used to identify the F-CPU and must therefore be unique network-wide.

## Overview of password assignment and prompt

The following table provides an overview of the access permissions for the F-CPU and the safety program.

The sections below show you how to assign the passwords and how to set up, change, and cancel access permissions for the F-CPU and the safety program.

| | Password for F-CPU | Password for safety program |
|---|---|---|
| **Assignment** | In the *hardware and network editor*, during configuration of the F-CPU, inspector window, in "Settings" tab under "Protection", corresponding safety level, e.g., "Write protection for fail-safe blocks" (S7-300, S7-400).<br>Select at least the access level "Full access (no protection)" for S7-1200/1500 F-CPUs and assign a password for "Full access incl. fail-safe (no protection)".<br>If you select a higher protection level, for example to protect the standard user program, you must assign an additional password for "Full access (no protection)". | In the *Safety Administration Editor* under "Access Protection" |
| **Prompt** | If you do not have access permission for the safety program (Page 79):<br>e.g.<br>• When the safety program is downloaded in its entirety<br>• (S7-300, 400) When the hardware configuration is downloaded<br>• When F-destination addresses are assigned<br>• When F-blocks that are used in the safety program are downloaded and deleted<br>• When disabling safety mode<br>• When restoring a backup of the F-CPU.<br>Exception with S7-1200/1500 F-CPUs: If neither the safety program nor the F-CPU password is changed by the restore process, you are not prompted for the F-CPU password. | If you have assigned a password and this has not yet been entered since the project was opened, or you do not have access permission for the safety program (Page 79):<br>**Offline password**<br>e.g.:<br>• When the password is changed<br>• When modifying the safety program<br>• When changing and deleting F-runtime groups<br>• When changing safety-related parameters of F-I/O<br>**Online password**<br>e.g., when disabling safety mode (the password must always be entered, even if access permission for the safety program is still valid) |

Safety program recompilation is required after changes to standard DBs to which the safety program has read or write access (Page 159). These standard DBs are not governed by the safety program access protection.

Please note that you also require the online password to download the safety-relevant changes to the hardware configuration. This is also true for changes to F-I/O not used in the safety program.
You have to also recompile and download the safety program for the download to be consistent.

## 4.2 Setting up, changing and revoking access permission for the safety program

### Procedure for assigning the password for the safety program

Use the following procedure to assign the password for the safety program:

1. Open the folder for your F-CPU in the project tree.
2. Select "Safety Administration" and select "Go to protection" in the shortcut menu.

   Alternatively, double-click on "Safety Administration". The *Safety Administration Editor* of the F-CPU will open. Select "Protection" in the area navigation.
3. Under "Offline safety program protection", click "Setup" and enter the password for the safety program in the following dialog in the "New password" and "Confirm password" fields.
4. Confirm the assigned password with "OK".

**Note**

You cannot define the online password separately; the offline password assigned during the next download is applied. After a change to the offline password, the online and offline passwords may differ until the next time the offline safety program is downloaded to the F-CPU.

**Note**

Use different passwords for the F-CPU and the safety program to optimize access protection.

**⚠ WARNING**

If access protection is not used to limit access to the programming device or PC to only those persons who are authorized to modify the safety program, the following organizational measures must be taken to ensure the effectiveness of the password protection for the F-CPU at the programming device or PC:

- Only authorized personnel may have access to the password.
- Authorized personnel must explicitly cancel the access permission for the F-CPU before leaving the programming device or PC by closing *STEP 7* or via the "Online > Delete access rights" menu. If this is not strictly implemented, a screen saver equipped with a password accessible only to authorized personnel must also be used. *(S006)*

### Changing the password for the safety program

You may change the safety program password as long as you have the necessary access permissions. It takes place likewise in the "Protection" area (via "Change" button) and is carried out as usual under Windows through entry of the old and double entry of the new password.

## Revoking the password for the safety program

You can revoke the password for a safety program, i.e., the safety program is then no longer protected by a password.

Proceed as follows:

1. Open the folder for your F-CPU in the project tree.
2. Select "Safety Administration" and select "Go to protection" in the shortcut menu.

   Alternatively, double-click on "Safety Administration". The *Safety Administration Editor* of the F-CPU will open.
3. Select "Protection" in the area navigation.
4. Click the "Change" button.
5. Under "Old password", enter the password for the safety program.
6. Click "Revoke" and then on "OK".

## Logging in to the safety program

Log in to the safety program as follows:

1. Open the folder for your F-CPU in the project tree.
2. Select "Safety Administration" and select "Go to protection" in the shortcut menu.

   Alternatively, double-click on "Safety Administration". The *Safety Administration Editor* of the F-CPU will open.
3. Select "Protection" in the area navigation.
4. Enter the password for the safety program in the "Password" input field.
5. Select the "Login" button.

## Validity of access permission for a safety program

If access permission for a safety program was granted through the entry of the password, this remains until the project is closed. If *STEP 7* is closed, any project that is still open is automatically closed and any access permission granted is reset.

## Canceling access permission for the safety program

The access permission for the safety program can be canceled as follows:

- By clicking the "Log off" button in the "Access protection" area in the "*Safety Administration Editor*".
- In the shortcut menu for the *Safety Administration Editor* shortcut menu (access by right-clicking).
- By using the lock symbol in the line of the *Safety Administration Editor*.

The user will then be prompted to enter the password for the safety program again the next time an action requiring a password is performed. To "cancel" access permission when using the Modify function, the connection to the F-CPU must first be terminated (e.g. with the "Online > Go offline" menu command or the corresponding icon in the toolbar).

Access permission for the safety program is reset automatically, if the project or *STEP 7* has been closed.

## Displaying the validity of access permission

The validity of the access permission is displayed in the project tree as follows:

- The access permission is valid, if the lock symbol in the line of the *Safety Administration Editor* is shown unlocked.
- The access permission is not available, if the lock symbol shows a closed lock.
- If no lock symbol is shown, no password was assigned.

# 4.3 Setting up access permission for the F-CPU

## Obtaining access permission for the F-CPU

You obtain access permission for the F-CPU - depending on the configured protection level - by entering the password for the F-CPU prior to performing an action requiring a password.

## Procedure for assigning a password for the F-CPU

You assign the password for the F-CPU when configuring the F-CPU.

You arrive there directly, if you click the link "Go to the "Protection" area of the F-CPU" in the "Access protection" area in the *Safety Administration Editor*.

Assign a password for "Write protection for fail-safe blocks" for F-CPUs S7-300/400.

Assign at least the level "Full access (no protection)" for S7-1200/1500 F-CPUs and assign a password under "Full access incl. fail-safe (no protection)". Proceed as described in the *STEP 7* help under "Configuring access levels".

(S7-1200, S7-1500) If you select a higher protection level, for example to protect the standard user program, you must assign an additional password for "Full access (no protection)".

**⚠ WARNING**

If **multiple F-CPUs** can be accessed over a network (e.g. Industrial Ethernet) by **a programming device or PC**, you must take the following additional measures to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, such as a unique password for the F-CPU having the respective Ethernet address "PW_8".

Note the following:

- A point-to-point connection must be used when assigning a password to an F-CPU for the first time (as when assigning an MPI address to an F-CPU for the first time).
- Before downloading a safety program to an F-CPU, you must first revoke an existing access permission for any other F-CPU.
- After activation of the access protection and before the transition to productive mode, you must download the safety program again to the F-CPU. *(S021)*

## Changing the password for the F-CPU

For the new password to become valid after a password change for the F-CPU, you must download the changed configuration into the F-CPU. If necessary, you must enter the "old" password for the F-CPU for this load operation. The F-CPU must be in STOP mode.

## Validity/cancelation of the access permission for the F-CPU

Access permission for the F-CPU remains valid until the project is closed in *STEP 7* or access permission is canceled using the "Online > Delete access rights" menu command.

**⚠ WARNING**

If access protection is not used to limit access to the programming device or PC to only those persons who are authorized to modify the safety program, the following organizational measures must be taken to ensure the effectiveness of the password protection for the F-CPU at the programming device or PC:

- Only authorized personnel may have access to the password.
- Authorized personnel must explicitly cancel the access permission for the F-CPU before leaving the programming device or PC by closing *STEP 7* or via the "Online > Delete access rights" menu. If this is not strictly implemented, a screen saver equipped with a password accessible only to authorized personnel must also be used. *(S006)*

# Programming

# 5

## 5.1 Overview of Programming

### Introduction

A safety program consists of F-blocks that you create using the FBD or LAD programming language and F-blocks that are automatically added. Fault detection and reaction measures are automatically added to the safety program you create, and additional safety-related tests are performed. Moreover, you have the option to incorporate special ready-made safety functions in the form of instructions into your safety program.

An overview of the following is given below:

- The structure of the safety program
- The fail-safe blocks
- Differences in the programming of the safety program with FBD/LAD compared to programming of standard user programs

## 5.1.1 Program structure of the safety program (S7-300, S7-400)

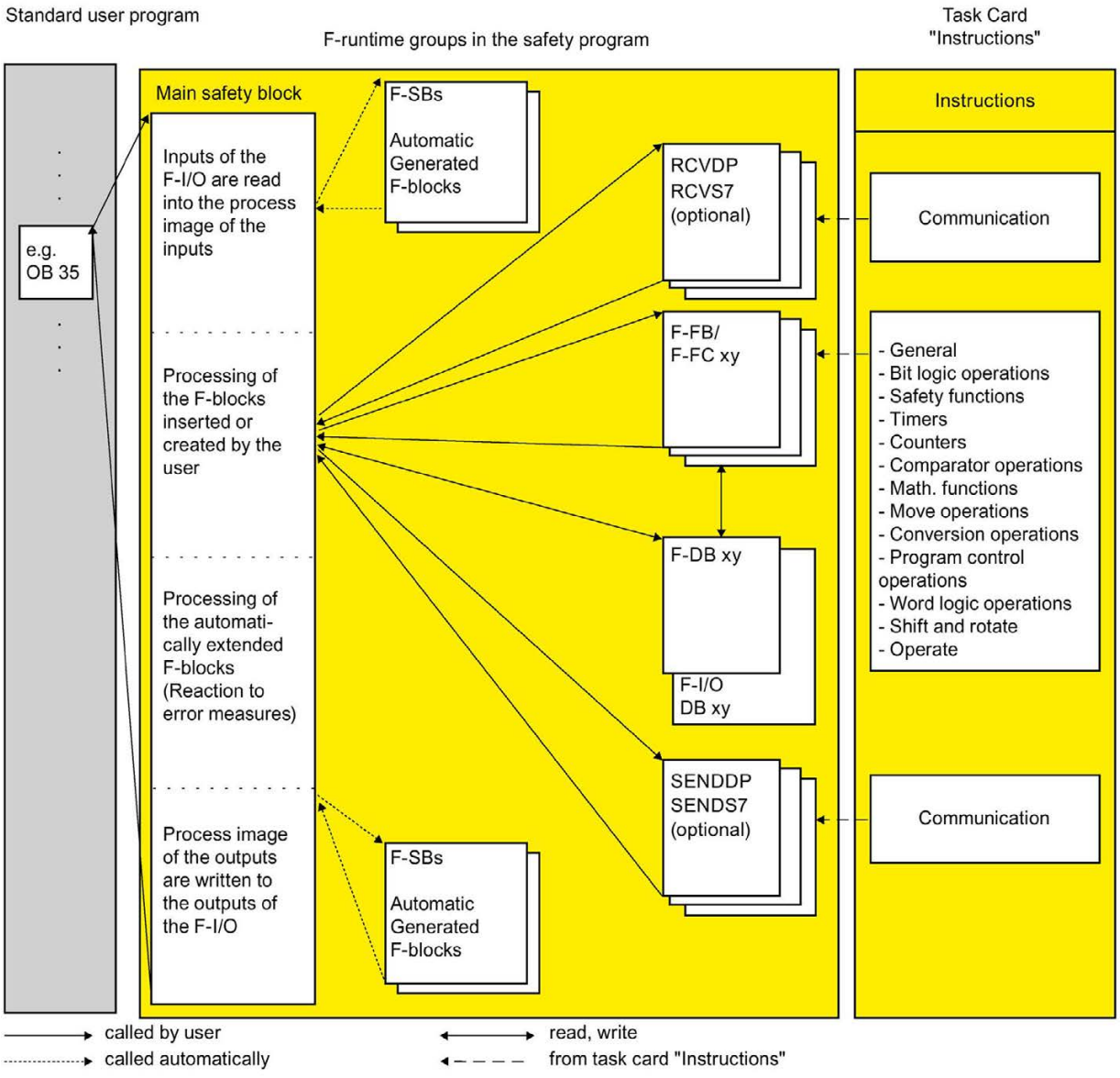
### Representation of program structure

For structuring purposes, a safety program consists of one or two F-runtime groups.

Each F-runtime group contains:

- F-blocks that you create using FBD or LAD or that are inserted from the project library or global libraries
- F-blocks that are added automatically (F-system blocks, automatically generated F-blocks, and F-I/O DBs)

Below is a schematic diagram of a safety program or an F-runtime group for an S7-300/400 F-CPU.

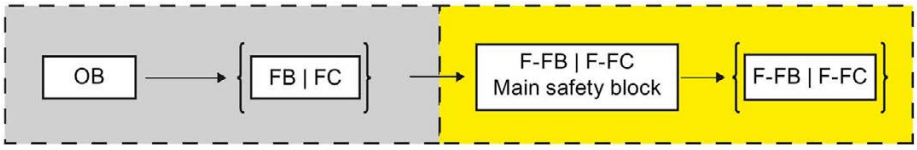


## Main safety block

The main safety block is the first F-block of the safety program that you program yourself. During compiling, it is supplemented by additional invisible calls of F-system blocks.

You must assign the main safety block to an F-runtime group (Page 102).

The main safety block in an S7-300/400 F-CPU is called from any block in the standard user program. We recommend a call from an OB 3x.




## F-runtime groups

To improve handling, a safety program consists of one or two "F-runtime groups". An F-runtime group is a logical construct of several related F-blocks that is formed internally by the F-system.

An F-runtime group consists of the following:

- a main safety block (an F-FB/F-FC that you assign to the OB)
- Any additional F-FBs or F-FCs that you program using FBD or LAD and call from the main safety block
- One or more F-DBs, as needed
- F-I/O DBs
- F-blocks from the project library or global libraries
- F-system blocks F-SBs
- Automatically generated F-blocks

## Structuring the safety program in two F-runtime groups

You can divide your safety program into two F-runtime groups. By having parts of the safety program (one F-runtime group) run in a faster priority class, you achieve faster safety circuits with shorter response times.

## 5.1.2 Program structure of the safety program (S7-1200, S7-1500)

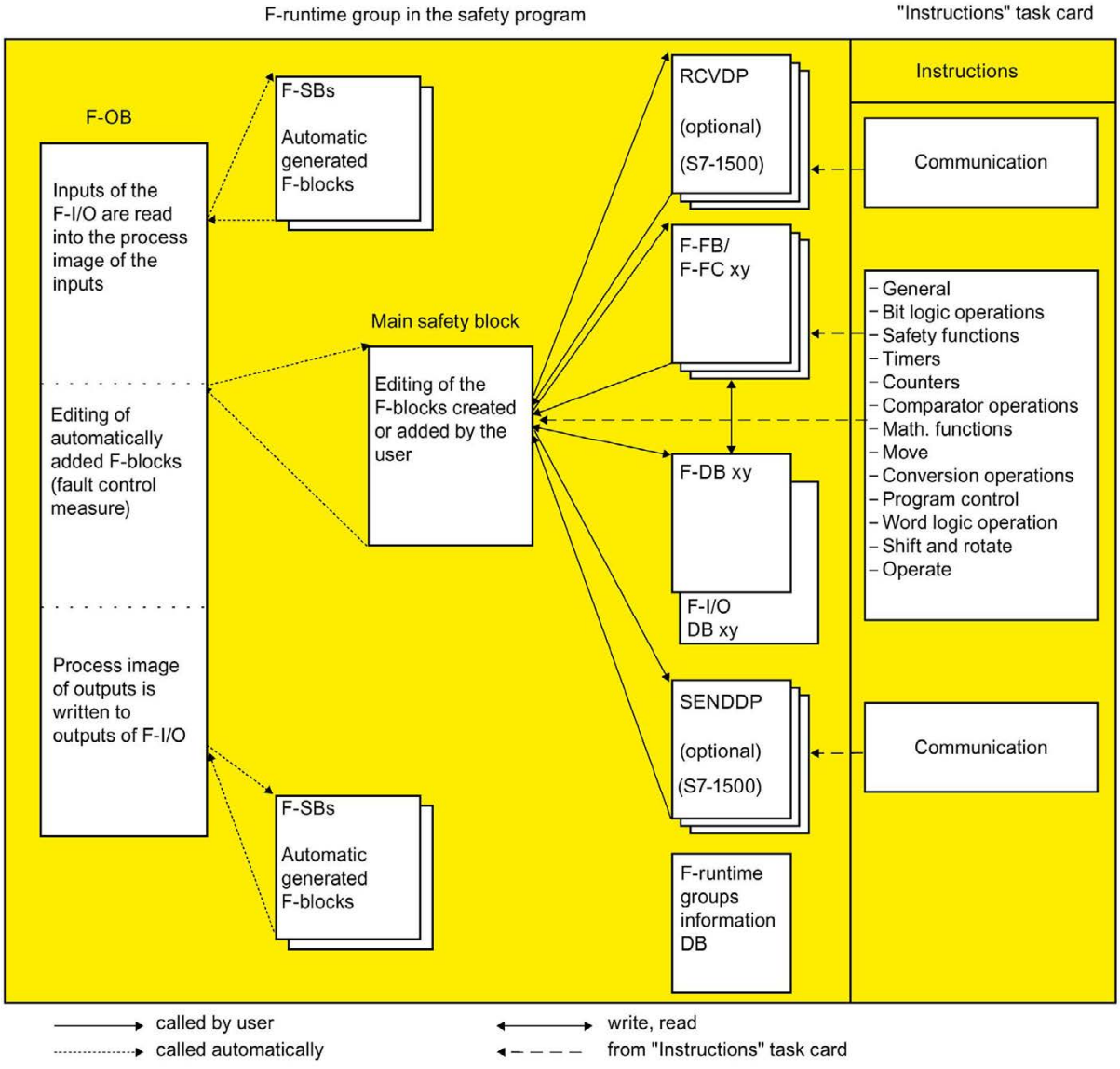
### Representation of program structure

For structuring purposes, a safety program consists of one or two F-runtime groups.

Each F-runtime group contains:

- F-blocks that you create using FBD or LAD or that are inserted from the project library or global libraries
- F-blocks that are added automatically (F-system blocks F-SBs, automatically generated F-blocks, F-runtime DB, and F-I/O DBs)

Below is a schematic diagram of a safety program or an F-runtime group for an S7-1200/1500 F-CPU.

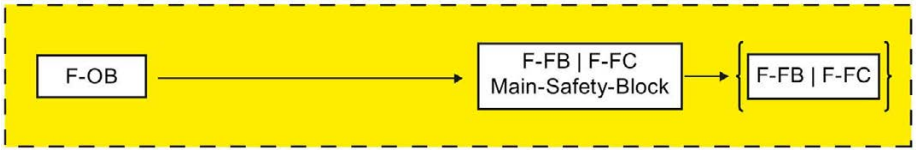


## Main safety block

The main safety block is the first F-block of the safety program that you program yourself.

You must assign the main safety block to an F-runtime group (Page 102).

The main safety block in an S7-1200/1500 F-CPU is called by the F-OB assigned to the F-runtime group.




## F-runtime groups

To improve handling, a safety program consists of one or two "F-runtime groups". An F-runtime group is a logical construct of several related F-blocks that is formed internally by the F-system.

An F-runtime group consists of the following:

- An F-OB which calls the main safety block
- A main safety block (an F-FB/F-FC that you assign to the F-OB)
- Any additional F-FBs or F-FCs that you program using FBD or LAD and call from the main safety block
- One or more F-DBs, as needed
- F-I/O DBs
- F-runtime group information DB
- F-blocks from the project library or global libraries
- F-system blocks F-SBs
- Automatically generated F-blocks

## Structuring the safety program in two F-runtime groups

You can divide your safety program into two F-runtime groups. By having parts of the safety program (one F-runtime group) run in a faster priority class, you achieve faster safety circuits with shorter response times.

## See also

F-runtime group information DB (S7-1200, S7-1500) (Page 115)

## 5.1.3 Fail-Safe Blocks

### F-blocks of an F-runtime group

The following table shows the F-blocks that you use in an F-runtime group:

| F-block | Function | S7-300/400 F-CPUs | S7-1200/ 1500 F-CPUs |
|---|---|---|---|
| Main safety block | The first step in programming of the safety program is the main safety block.<br>The main safety block in S7-300/400 F-CPUs is an F-FC or F-FB (with instance DB), which is called by a standard block (recommendation: OB 35) from the standard user program.<br>The main safety block in S7-1200/1500 F-CPUs is an F-FC or F-FB (with instance DB), which is called by the F-OB. | X | X |
| F-FB/F-FC | Both in the main safety block as well as additional F-FBs and F-FCs, you can perform the following:<br>• Program the safety program with the instructions available for F-blocks in FBD or LAD<br>• Call other created F-FBs/F-FCs for structuring the safety program<br>• Insert F-blocks from the project library or global libraries | X | X |
| F-DB | Optional fail-safe data blocks that can be read- and write-accessed within the entire safety program. | X | X |
| F-I/O DB | An F-I/O DB is automatically generated for each F-I/O when it is configured. You can or you must access the tags of the F-I/O DB in conjunction with F-I/O accesses. | X | X |
| F-shared DB | The F-shared DB is a fail-safe data block that contains all of the shared data of the safety program and additional information needed by the F-system. | X | — |
| F-runtime group information DB | An F-runtime group information DB is created when you create an F-runtime group.<br>The F-runtime group information DB provides information on the F-runtime group and on the safety program as a whole. | — | X |

**Note**

Main safety blocks, F-FBs/F-FCs and F-DBs of the safety program of an S7-1200/1500 F-CPU cannot be know-how-protected. If they are, the safety program cannot be compiled.

**Note**

You may not add, edit or delete blocks in the "System blocks" folder. Failure to do so can lead to compilation errors or cause an F-CPU stop.

**Note**

You are not permitted to insert F-system blocks from the "System blocks" folder in a main safety block/F-FB/F-FC.

### Instructions for the safety program

In the "Instructions" task card, you can find instructions for the F-CPU used and which you can to use program the safety program.

You can find instructions that you know from the standard user program, such as bit logic operations, mathematical functions, functions for program control, and word logic operations.

Moreover, there are instructions with safety functions, e.g., for two-hand monitoring, discrepancy analysis, muting, emergency STOP, safety door monitoring, and feedback loop monitoring.

### Additional information

For a detailed description of the instructions for the safety program, refer to Overview of instructions (Page 328).

## 5.1.4 Restrictions in the programming languages FBD/LAD

### LAD and FBD programming languages

The user program in the F-CPU typically consists of a standard user program and a safety program.

The standard user program is created using standard programming languages such as SCL, STL, LAD, or FBD.

For the safety program, LAD or FBD may be used with certain restrictions in the instructions and the applicable data types and operand areas. Please also note the restrictions for the individual instructions in FBD (Page 494) and LAD (Page 329).

### Supported instructions

The instructions available depend on the F-CPU used. You can find the supported instructions in the description of the instructions (starting from Overview of instructions (Page 328)).

**Note**

Preconnection of enable input EN or evaluation of enable output ENO is not possible.

## Supported data types and parameter types

**Only** the following data types are supported:

- BOOL
- INT
- WORD
- DINT
- DWORD (S7-300, S7-400)
- TIME
- F-compliant PLC data type (S7-1200, S7-1500)

**Note**

If the result of an instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostics event is entered in the diagnostics buffer of the F-CPU.
You can find additional information on the cause in the online help for diagnostic messages.
You must therefore ensure that the permitted range for the data type is observed when creating the program or select a matching data type.

**Note**

**DINT data type for S7-1200 F-CPUs**

Note that the runtimes of the instructions "Comparator operations" and "Math functions" with data type DINT are very time consuming on S7-1200 F-CPUs.

## Non-permitted data and parameter types

The following types are **not** permitted:

- All types not listed in the section "Supported data types and parameters types" (e.g. BYTE, REAL)
- Complex data types (e.g., STRING, ARRAY, STRUCT, PLC data type (S7300/400))
- Parameter types (e.g., BLOCK_FB, BLOCK_DB, ANY)

## Supported operand areas

The system memory of an F-CPU is divided into the same operand areas as the system memory of a standard CPU. You can access the operand areas listed in the table below from within the safety program.

Table 5- 1 Supported operand areas

| Operand area | Access via units of the following size: | S7 Notation | Description |
|---|---|---|---|
| **Process image of the inputs** | | | |
| • Of F-I/O | Input (bit)<br>Input word<br>Input double-word | I<br>IW<br>ID | Only read-only access to input channels of F-I/O is possible.<br>Transfer to IN_OUT parameters of an F-FB or F-FC is therefore not valid either.<br>The process image of the inputs of F-I/O is updated before the start of the main safety block. |
| • Of standard I/O | Input (bit)<br>Input word<br>Input double-word | I<br>IW<br>ID | Input channels of standard I/O can only be accessed read-only.<br>Transfer to IN_OUT parameters of an F-FB or F-FC is therefore not valid either.<br>In addition, a process-specific validity check is required.<br>See the *STEP 7 help* for the update times of the process image of the inputs of standard I/O. |
| **Process image of the outputs** | | | |
| • Of F-I/O | Output (bit)<br>Output word<br>Output double-word | Q<br>QW<br>QD | Only write-only access to output channels of F-I/O is possible.<br>Transfer to IN_OUT parameters of an F-FB or F-FC is therefore not valid either.<br>In the safety program, the values for the outputs of the F-I/O are calculated and stored in the process image of the outputs.<br>The process image of the outputs for F-I/O is updated after the end of the main safety block. |
| • Of standard I/O | Output (bit)<br>Output word<br>Output double-word | Q<br>QW<br>QD | Output channels of standard I/O are write-only channels.<br>Transfer to IN_OUT parameters of an F-FB or F-FC is therefore not valid either.<br>In the safety program, the values for the outputs of the standard I/O are also calculated and stored in the process image of the outputs, if needed.<br>See the *STEP 7 help* for the update times of the process image of the outputs of standard I/O. |

| Operand area | Access via units of the following size: | S7 Notation | Description |
|---|---|---|---|
| **Bit memory** | Bit memory (bit)<br>Memory word<br>Memory double word | M<br>MW<br>MD | This area is used for data exchange with the standard user program.<br>In addition, read access requires a process-specific validity check.<br>A particular element of the bit memory can be either read- or write-accessed in the safety program.<br>Transfer to IN_OUT parameters of an F-FB or F-FC is therefore not valid either.<br>Note that it is only permitted to use bit memory for connecting the standard user program and the safety program; it must not be used as a buffer for F-data. |
| **Data blocks** | | | |
| • F-DB | Data bit<br>Data word<br>Data double word | DBX<br>DBW<br>DBD | Data blocks store information for the program. They can either be defined as global data blocks such that all F-FBs, F-FCs, or main safety blocks can access them or assigned to a particular F-FB or main safety block (instance DB). A tag of a shared DB can only be accessed from one F-runtime group, and an instance DB only from the F-runtime group in which the corresponding F-FB/instruction is called. |
| • DB | Data bit<br>Data word<br>Data double word | DBX<br>DBW<br>DBD | This area is used for data exchange with the standard user program.<br>In addition, read access requires a process-specific validity check.<br>For a tag of a DB, either read access or write access is possible in the safety program.<br>Transfer to IN_OUT parameters of an F-FB or F-FC is therefore not valid either.<br>Please note that the tags of a DB can only be used for transferring data between the standard user program and the safety program; DBs must not be used as a buffer for F-data. |
| **Temporary local data** | Local data bit<br>Local data word<br>Local data double word | L<br>LW<br>LD | This memory area holds the temporary tags of a block (or F-block) while the (F-) block is being executed. The local data stack also provides memory for transferring block parameters and for saving intermediate results. |

### File type conversion

Just as with the standard user program, there are two possibilities for file type conversion in the safety program.

- Implicit conversion

  The implicit conversion is executed as in the standard user program with the following restrictions: The bit length of the source data type has to match the bit length of the destination data type.

- Explicit conversion

  You use the explicit conversion instruction in FBD (Page 617) and LAD (Page 448), before the actual instruction is executed.

### Slice access

Slice access is not possible in the safety program.

### Non-permitted operand areas

Access via units other than those listed in the table above is **not** permitted. The same applies to access to operand areas not listed, in particular:

- Data blocks that were automatically added

  Exception: Certain tags in the F-I/O DB (Page 129) and in the F-shared DB (S7-300, S7-400) (Page 114) or F-runtime group information DB (S7-1200, S7-1500) (Page 115)

- I/O area: Inputs
- I/O area: Outputs

### Boolean constants "0" or "FALSE" and "1" or "TRUE" (S7-300, S7-400)

If you require Boolean constants "0" or "FALSE" and "1" or "TRUE" in your safety program to assign parameters during block calls, you can access the tags "VKE0" and "VKE1" in the F-shared DB using fully qualified DB access ("F_GLOBDB".VKE0 or "F_GLOBDB".VKE1).

### Boolean constants "0" or "FALSE" and "1" or "TRUE" (S7-1200, S7-1500)

The Boolean constants "0" or "FALSE" and "1" or "TRUE" are available for S7-1200/1500 F-CPUs to assign parameters during block calls.

If you require the Boolean constant 1 for bit logic operations, you can interconnect the corresponding input of the bit logic operation with the "Assignment" instruction. Do not interconnect the box input of the "Assignment" instruction in FBD.

If you require the Boolean constant 0 for bit logic operations, you can interconnect the corresponding input of the bit logic operation with the "Assignment" instruction. Negate the output of the "Assignment" instruction. Do not interconnect the box input of the "Assignment" instruction in FBD.

## Operand area of temporary local data: Particularities

**Note**

Note when using the operand area of temporary local data that the first access of a local data element in a main safety block/F-FB/F-FC must always be a write access. This initializes the local data element.

Make sure that the initialization of the local data element is **not** skipped over by JMP, JMPN. or RET instructions (branching).

The "local data bit" should be initialized with the Assign ("=") (FBD) or ("--( )") (LAD) instruction. Assign the local data bit a signal state of "0" or "1" as a Boolean constant.

Local data bits cannot be initialized with the Flip Flop (SR, RS), Set Output (S) or Reset Output (R) instructions.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

## "Fully qualified DB access"

Access to tags of a data block in an F-FB/F-FC is "fully qualified DB access". This also applies to initial access to tags of a data block after a jump label.

For S7-300/400 F-CPUs, only initial access needs to be "fully qualified DB access". Alternatively, you can use the instruction "OPN".

## Example of "fully qualified DB access":

Assign a name for the F-DB, e.g. "F_Data_1". Use the names assigned in the declaration of the F-DB instead of the absolute addresses.

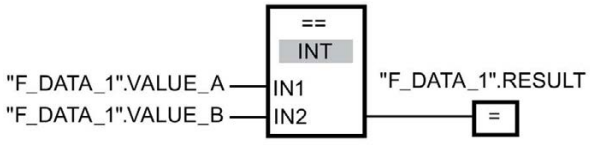



Figure 5-1 Example with fully-qualified access and symbolic names

### Example of "non-fully qualified DB access" (S7-300, S7-400):

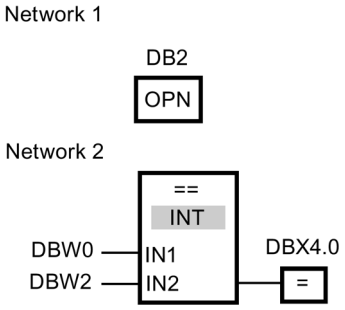



Figure 5-2 Example without fully-qualified access and symbolic names

### Access to instance DBs

You can also access instance DBs of F-FBs with fully qualified access, e.g., for transfer of block parameters. It is not possible to access static local data in single/multi-instances of other F-FBs.

Note that accessing instance DBs of F-FBs that are not called in the safety program can cause the F-CPU to go to STOP mode.

## 5.1.5 F-compliant PLC data types (S7-1200, S7-1500)

### Introduction

You declare and use F-compliant PLC data types as you would standard PLC data types. You can use F-compliant PLC data types in the safety program as well as in the standard user program.

Differences to standard PLC data types are described in this chapter.

Information on the use and declaration of standard PLC data types is available in the *STEP 7 online help* under "Using PLC data types (UDT)" and "Programming PLC data types".

### Declaring F-compliant PLC data types

You declare F-compliant PLC data types as you would PLC data types.

In F-compliant PLC data types, you can use all data types (Page 90) that you can also use in safety programs.

Nesting of F-compliant PLC data types within F-compliant PLC data types is not supported.

Proceed as follows for declaration:

1. Click on "Add new PLC data type" in the "PLC Data Types" folder in the project tree.
2. To create an F-compliant PLC data type, enable the option "Create F-compliant PLC data type" in the "Add new PLC data type" dialog.
3. Proceed as described in the *STEP 7 online help* under "Programming structure of PLC data types".

You specify default values for F-compliant PLC data types during the declaration.

When you access the F-I/O using an F-compliant PLC data type, the structure of the F-compliant PLC data type tag must match the channel structure of the F-I/O.

An F-compliant PLC data type for an F-I/O with 8 digital inputs, for example, may consist of up to 8 tags of the BOOL data type. For F-I/O with value status, there are up to 8 more tags of the BOOL data type for the value status. For an F-I/O with 6 analog inputs, it can consist of up to 6 tags of the INT data type. For F-I/O with value status, there are up to 6 more tags of the BOOL data type for the value status. Access to F-I/O is only permitted for existing and enabled channels. Please note the restrictions detailed in F-I/O access (Page 122).

### Using F-compliant PLC data types

You use F-compliant PLC data types as you would standard PLC data types.

### Changes to F-compliant PLC data types

You need the password for the safety program to change F-compliant PLC data types. Regardless if you are using the F-compliant PLC data type in an F-block, in a standard block or not at all.

### See also

"F-compliant PLC data types" area (S7-1200, S7-1500) (Page 70)

## 5.1.5.1 Example of the use of F-compliant PLC data types for access to F-I/O (S7-1200, S7-1500)

### Introduction

This example uses the F-module 4 F-DI/3 F-DO DC24V/2A with 1oo2 evaluation to demonstrate how you use F-compliant PLC data types for access to F-I/O.

### Channel structure of the 4 F-DI/3 F-DO DC24V/2A F-module

The table below sets out the channel structure and address assignment of the 4 F-DI/3 F-DO DC24V/2A F-module with 1oo2 evaluation. You may only access existing and enabled channels (addresses I11.0 to I11.3 and I12.0 to I12.3). These channels provide the result of 1oo2 evaluation generated internally in the F-module.

Table 5- 2 Channel structure and addresses of the channel values of inputs with 1oo2 evaluation

| Channel | Address |
|---|---|
| DI channel 0 channel value | I11.0 |
| DI channel 1 channel value | I11.1 |
| DI channel 2 channel value | I11.2 |
| DI channel 3 channel value | I11.3 |
| — | I11.4 |
| — | I11.5 |
| — | I11.6 |
| — | I11.7 |

Table 5- 3 Channel structure and addresses of the value status of inputs with 1oo2 evaluation

| Channel | Address |
|---|---|
| DI channel 0 value status | I12.0 |
| DI channel 1 value status | I12.1 |
| DI channel 2 value status | I12.2 |
| DI channel 3 value status | I12.3 |
| — | I12.4 |
| — | I12.5 |
| — | I12.6 |
| — | I12.7 |

Table 5- 4 Channel structure and addresses of the value status of outputs

| Channel | Address |
|---|---|
| DO channel 0 value status | I13.0 |
| DO channel 1 value status | I13.1 |
| DO channel 2 value status | I13.2 |
| DO channel 3 value status | I13.3 |

Table 5- 5 Channel structure and addresses of the channel values of outputs

| Channel | Address |
|---|---|
| DO channel 0 channel value | Q11.0 |
| DO channel 1 channel value | Q11.1 |
| DO channel 2 channel value | Q11.2 |
| DO channel 3 channel value | Q11.3 |

## Creating F-compliant PLC data types

Create two F-compliant PLC data types, for example, for access to all channels.

The figure below shows an F-compliant PLC data type for access to the channel values and value status of the inputs with 1oo2 evaluation:

**4 F-DI/3 F-DO DC24V/2A_DI**

| | Name | Data type | Comment |
|---|---|---|---|
| 1 | CH_DI_0 | Bool | Channel 0 (DI) |
| 2 | CH_DI_1 | Bool | Channel 1 (DI) |
| 3 | CH_DI_2 | Bool | Channel 2 (DI) |
| 4 | CH_DI_3 | Bool | Channel 3 (DI) |

Figure 5-3 F-compliant "4 F-DI/3 F-DO DC24V/2A_DI" PLC data type

The figure below shows the F-compliant PLC data type for access to the channel values and value status of the outputs:

**4 F-DI/3 F-DO DC24V/2A_DO**

| | Name | Data type | Comment |
|---|---|---|---|
| 1 | CH_DO_0 | Bool | Channel 0 (DQ) |
| 2 | CH_DO_1 | Bool | Channel 1 (DQ) |
| 3 | CH_DO_2 | Bool | Channel 2 (DQ) |

Figure 5-4 F-compliant "4 F-DI/3 F-DO DC24V/2A_DO" PLC data type

## Using F-compliant PLC data types

As demonstrated in the figure below, you can use the two F-compliant PLC data types that you have created in an F-FC (e.g. "Motor"):

**Motor**

| | Name | Data type | Default value | Comment |
|---|---|---|---|---|
| 1 | Input | | | |
| 2 | Motor SS DI | "4 F-DI/3 F-DO DC24V/2A_DI" | | Motor interface channel values DI |
| 3 | CH_DI_0 | Bool | | Channel 0 (DI) |
| 4 | CH_DI_1 | Bool | | Channel 1 (DI) |
| 5 | CH_DI_2 | Bool | | Channel 2 (DI) |
| 6 | CH_DI_3 | Bool | | Channel 3 (DI) |
| 7 | Motor SS DI VS | "4 F-DI/3 F-DO DC24V/2A_DI" | | Motor interface value status DI |
| 8 | CH_DI_0 | Bool | | Channel 0 (DI) |
| 9 | CH_DI_1 | Bool | | Channel 1 (DI) |
| 10 | CH_DI_2 | Bool | | Channel 2 (DI) |
| 11 | CH_DI_3 | Bool | | Channel 3 (DI) |
| 12 | Motor SS DO VS | "4 F-DI/3 F-DO DC24V/2A_DO" | | Motor interface value status DQ |
| 13 | CH_DO_0 | Bool | | Channel 0 (DQ) |
| 14 | CH_DO_1 | Bool | | Channel 1 (DQ) |
| 15 | CH_DO_2 | Bool | | Channel 2 (DQ) |
| 16 | Output | | | |
| 17 | Motor SS DO | "4 F-DI/3 F-DO DC24V/2A_DO" | | Motor interface channel values DQ |
| 18 | CH_DO_0 | Bool | | Channel 0 (DQ) |
| 19 | CH_DO_1 | Bool | | Channel 1 (DQ) |
| 20 | CH_DO_2 | Bool | | Channel 2 (DQ) |
| 21 | InOut | | | |

## Access to the channels with PLC tags

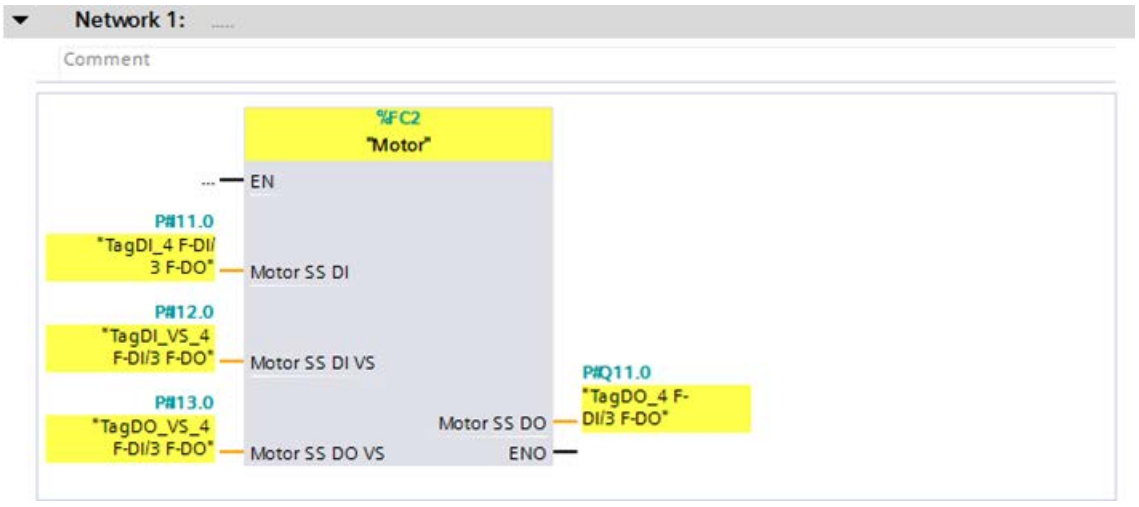
Create PLC tags for access to the channels (incl. value status):

**Tag table_1**

| | Name | Data type | Address | Comment |
|---|---|---|---|---|
| 1 | TagDI_4 F-DI/3 F-DO(1) | "4 F-DI/3 F-DO DC24V/2A_DI" | %I11.0 | Motor 1 DI |
| 2 | TagDI_VS_4 F-DI/3 F-DO(1) | "4 F-DI/3 F-DO DC24V/2A_DI" | %I12.0 | Motor 1 DI VS |
| 3 | TagDO_VS_4 F-DI/3 F-DO(1) | "4 F-DI/3 F-DO DC24V/2A_DO" | %I13.0 | Motor 1 DQ VS |
| 4 | TagDO_4 F-DI/3 F-DO(1) | "4 F-DI/3 F-DO DC24V/2A_DO" | %Q11.0 | Motor 1 DQ |

## Accessing the F-FC

Transfer the PLC tags you have created when you call the F-FC (e.g. "Motor"):




## See also

F-I/O access (Page 122)

Value status (S7-1200, S7-1500) (Page 124)

# 5.2 Defining F-Runtime Groups

## 5.2.1 Rules for F-Runtime Groups of the Safety Program

### Rules

Note the following:

- The channels (channel values and value status) of an F-I/O can only be accessed from a single F-runtime group.
- Tags of the F-I/O DB of an F-I/O can only be accessed from one F-runtime group and only from that F-runtime group from which the channels or value status of this F-I/O are also accessed (if access is made).
- F-FBs can be used in more than one F-runtime group but they must be called with different instance DBs.
- Instance DBs can only be accessed from the F-runtime group in which the associated F-FB is called.
- Individual tags of F-DBs (except the F-shared DB) can only be used in one F-runtime group (however, an F-DB can be used in more than one F-runtime group).
- (S7-300, S7-400) A DB for F-runtime group communication can be read and write accessed by the F-runtime group to which it was assigned as "DB for runtime group communication", but only read-accessed by the "receiver" F-runtime group.
- (S7-300, S7-400) Individual tags of the F-runtime group DB can only be used in that F-runtime group.
- (S7-300, S7-400) An F-communication DB can only be accessed from one F-runtime group.
- (S7-1200, S7-1500) You must not call the main safety block yourself. It is automatically called by the assigned F-OB.
- (S7-1200, S7-1500). The F-OB should be created with the highest priority of all OBs.
- (S7-300, S7-400) The main safety block may only be called once from a standard block. Multiple calls can cause the F-CPU to go to STOP mode.
- (S7-300, S7-400) For optimal use of temporary local data, you must call the F-runtime group (the main safety block) directly in an OB (cyclic interrupt OB, if possible); you should not declare any additional temporary local data in this cyclic interrupt OB.
- (S7-300, S7-400) Within a cyclic interrupt OB, the F-runtime group should be executed **before** the standard user program; i.e. it should be called at the very beginning of the OB, so that the F-runtime group is always called at fixed time intervals, regardless of how long it takes to process the standard user program.

  For this reason, the cyclic interrupt OB should also not be interrupted by higher priority interrupts.

- The process image inputs and outputs of standard I/O, bit memory, and tags of DBs in the standard user program may be accessed either as read-only or read/write from more than one F-runtime group. (See also Data exchange between standard user program and safety program (Page 159))
- F-FCs can generally be called in more than one F-runtime group.

**Note**

You can improve performance by writing sections of the program that are not required for the safety function in the standard user program.

When determining which elements to include in the standard user program and which to include in the safety program, you should keep in mind that the standard user program can be modified and downloaded to the F-CPU more easily. In general, changes in the standard user program do not require an acceptance.

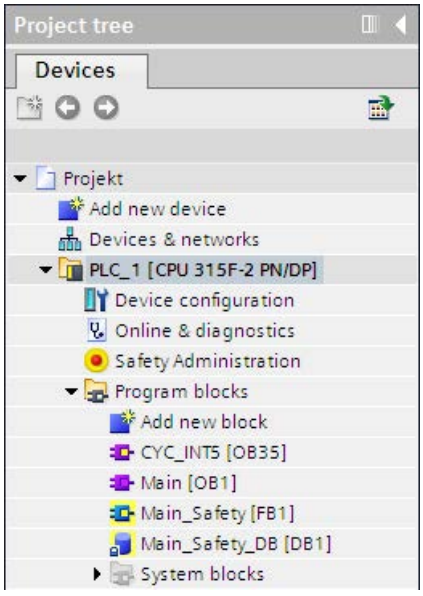
## 5.2.2 Procedure for defining an F-runtime group (S7-300, S7-400)

### Requirements

In your project, you have selected an S7-300 or S7-400 F-CPU from the "Hardware catalog" task card and inserted it into the *hardware and network editor*. In the "Properties" tab of the F-CPU, the "F-capability activated" check box is selected (default setting).

### F-runtime group created by default

*STEP 7 Safety* inserts F-blocks for an F-runtime group in the project tree by default. You see the (F-)blocks of the F-runtime group (CYC_INT5 [OB 35], Main_Safety [FB 1], and Main_Safety_DB [DB1] in the project tree when you open the "Program blocks" folder.

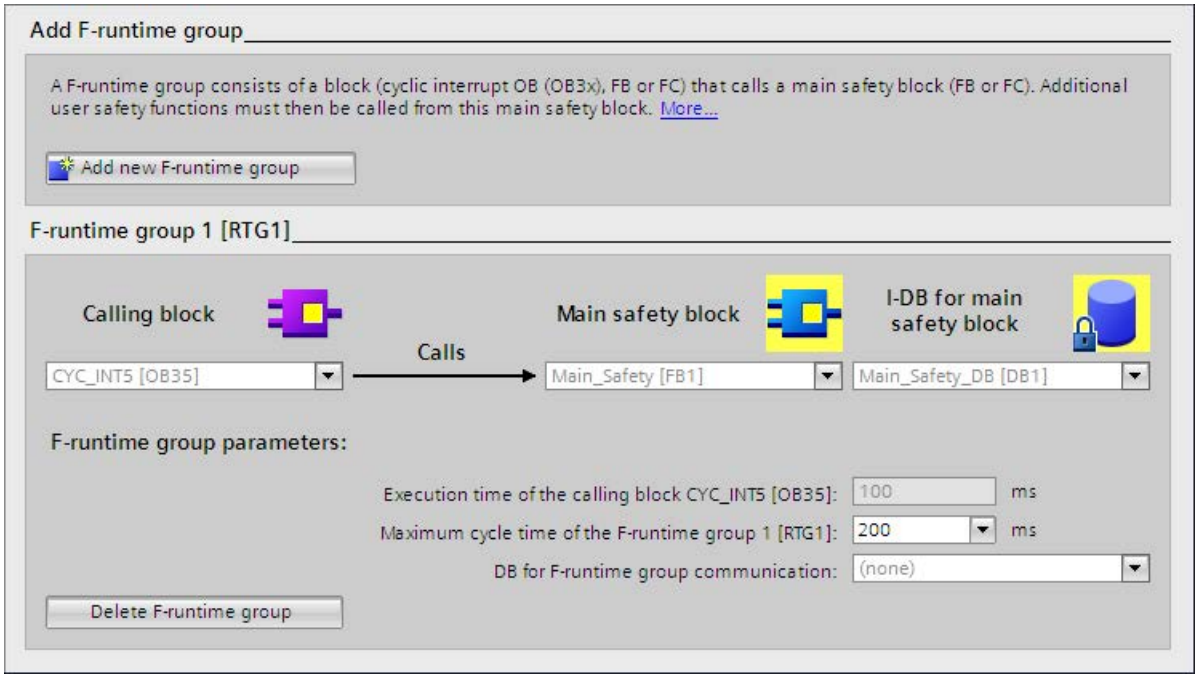


## Procedure for defining an F-runtime group

Proceed as follows to define an F-runtime group:

1. Open the *Safety Administration Editor* by double-clicking in the project tree.
2. Select "F-runtime group" in the area navigation.

   **Result:** The work area for defining an F-runtime group with the (default) settings for F-runtime group 1 opens.




3. Specify the block in which the main safety block is to be called.

   Cyclic interrupt OB 35 is suggested here by default. The advantage of using cyclic interrupt OBs is that they interrupt the cyclic program execution in OB 1 of the standard user program at fixed time intervals; that is, the safety program is called and executed at fixed time intervals in a cyclic interrupt OB.

   In this input field, you can select only those blocks that were created in the LAD, FBD, or STL programming language. If you select a block here, the call is inserted automatically into the selected block and, if necessary, removed from a previously selected block.

   If you want to call the main safety block in a block that was created in another programming language, you must program this call itself. The input field is then not editable (grayed out), and you can change the call only in the calling block and not the Safety Administration Editor.

4. Assign the desired main safety block to the F-runtime group. If the main safety block is an FB, you must also assign an instance DB.

   Main_Safety [FB1] and Main_Safety_DB [DB1]) are suggested by default.

5. The F-CPU monitors the F-cycle time in the F-runtime group. For "Maximum cycle time of F-runtime group", enter the maximum permitted time between two calls of this F-runtime group.

   **WARNING**

   The F-runtime group call interval is monitored for the maximum value; i.e. monitoring is performed to determine whether the call is executed often enough, but not whether it is executed too often. Fail-safe timers must therefore be implemented using the TP, TON, or TOF instructions (Page 409) from the "Instructions" task card and not using counters (OB calls). *(S007)*

6. If one F-runtime group is to provide tags for evaluation to another F-runtime group of the safety program, assign a DB for F-runtime group communication. Select an F-DB for "DB for F-runtime group communication". (See also Safety-related communication between the F-runtime groups of a safety program (S7-300, S7-400) (Page 110))
7. **Create an additional F-runtime group** by clicking the "Add new F-runtime group" button.
8. Assign an F-FB or F-FC as the main safety block to a calling block. This F-FB or F-FC is automatically generated in the project tree, if not already present.
9. If the main safety block is an F-FB, assign an instance DB to the main safety block. The instance DB is generated automatically in the project tree.
10. Follow steps 3 and 4 above to complete generation of the second F-runtime group.

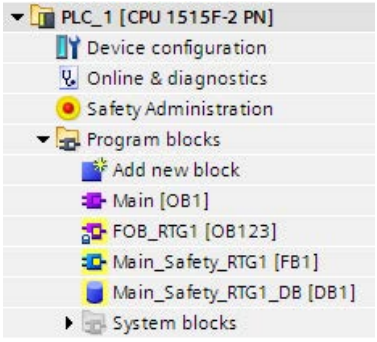
## 5.2.3 Procedure for defining an F-runtime group (S7-1200, S7-1500)

### Requirements

In your project, you have selected an S7-1200/1500 F-CPU from the "Hardware catalog" task card and inserted it into the *hardware and network editor*. In the "Properties" tab of the F-CPU, the "F-capability activated" check box is selected (default setting).

### F-runtime group created by default

*STEP 7 Safety* inserts F-blocks for an F-runtime group in the project tree by default. The (F-)blocks of the F-runtime group (FOB_RTG1 [OB123], Main_Safety_RTG1 [FB1] and Main_Safety_RTG1_DB [DB1]) are shown in the project tree when you open the "Program blocks" folder.

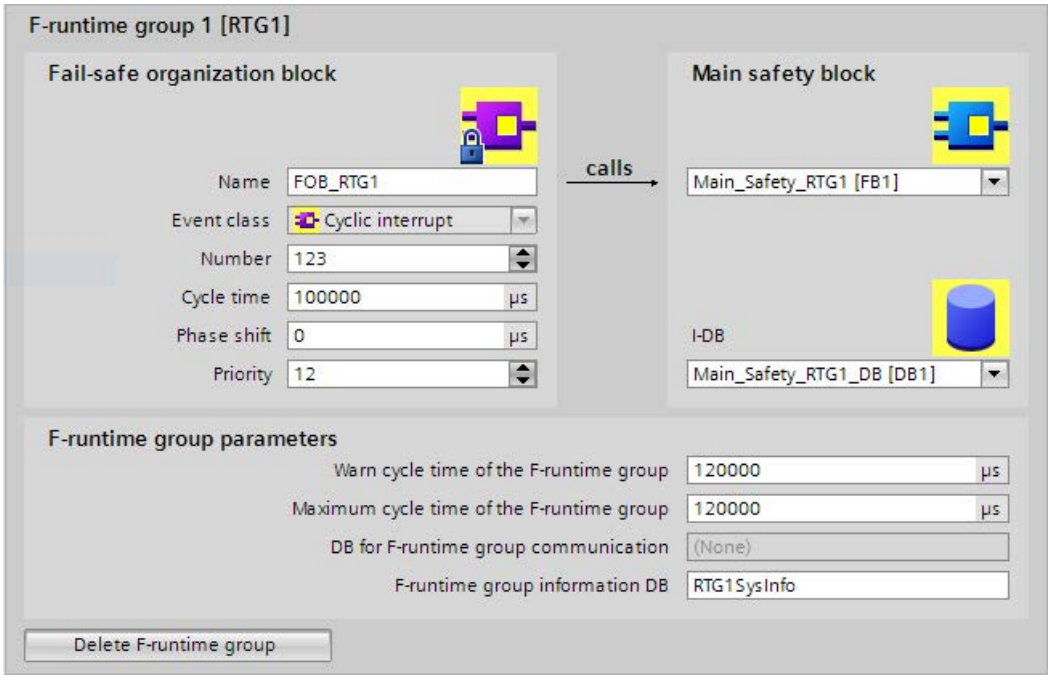


## Procedure for defining an F-runtime group

Proceed as follows to define an F-runtime group:

1. Open the *Safety Administration Editor* by double-clicking in the project tree.
2. Select "F-runtime group" in the area navigation.

   **Result:** The work area for defining an F-runtime group with the (default) settings for F-runtime group 1 opens.




3. Assign a name for the F-OB under "F-OB". The F-OB calls the main safety block.

   By default, the F-OB is created with the event class "cyclic interrupt OB". The advantage of the event class "cyclic interrupt OB" is that the F-OB calls the main safety block, and therefore the safety program, at set intervals.

4. Assign the F-OB a number.

5. Assign parameters for the cycle time, phase shifting and priority of the F-OB.

   Select a cycle time which is less than the "Maximum cycle time of F-runtime group" and less than the "Cycle time warning limit of F-runtime group".

   Select a phase shift which is less than the cycle time.

   **Note**

   You can also specify the event class "cyclic" for the F-OB. Keep in mind that the F-OB has the lowest priority in this case and that the runtime as well as the reaction time of the safety program are also influenced by the standard user program.

   **Note**

   We recommend creating the F-OB with the highest priority of all OBs. This will ensure:
   - Defined response times
   - Guaranteed data consistency during data communication between the standard and safety program. See also Data exchange between standard user program and safety program (Page 159).

6. Assign the calling main safety block to the F-OB. If the main safety block is an FB, you must also assign an instance DB.

   Main_Safety_RTG1 [FB1] and Main_Safety_RTG1_DB [DB1] are suggested by default.

7. The F-CPU monitors the F-cycle time in the F-runtime group. Two parameters are available:
   - If "Cycle time warning limit of F-runtime group" is exceeded, an entry is written to the diagnostics buffer of the F-CPU. This parameter can, for example, be used to determine whether the cycle time exceeds a required value without the F-CPU switching to STOP mode.
   - If "Maximum cycle time of F-runtime group" is exceeded, the F-CPU switches to STOP mode. For "Maximum cycle time of F-runtime group", select the maximum permitted time between two calls of this F-runtime group (maximum of 20000000 µs).

     **⚠ WARNING**

     The F-runtime group call interval is monitored for the maximum value; i.e. monitoring is performed to determine whether the call is executed often enough, but not whether it is executed too often. Fail-safe timers must therefore be implemented using the TP, TON, or TOF instructions (Page 409) from the "Instructions" task card and not using counters (OB calls). *(S007)*

   "Cycle time warning limit of F-runtime group" must be configured as less than or equal to the "Maximum cycle time of F-runtime group".

8. Assign a symbolic name for the F-runtime group DB (Page 115) under "F-runtime group DB".
9. **Create an additional F-runtime group** by clicking the "Add new F-runtime group" button.

10. If the main safety block is an F-FB, assign an instance DB to the main safety block. The instance DB is generated automatically in the project tree.
11. Follow steps 3 to 6 above to complete generation of the second F-runtime group.

## 5.2.4 Safety-related communication between the F-runtime groups of a safety program (S7-300, S7-400)

### Safety-related communication between F-runtime groups

Safety-related communication can take place between the two F-runtime groups of a safety program. This means fail-safe tags can be provided by one F-runtime group in an F-DB and read in another F-runtime group.

**Note**

A DB for F-runtime group communication can be read and write accessed by the F-runtime group to which it was assigned as "DB for runtime group communication", while it can only be read-accessed by the "receiver" F-runtime group.

**Tip:** You can improve performance by structuring your safety program in such a way that as few tags as possible are exchanged between the F-runtime groups.

### Procedure for defining a DB for F-runtime group communication

You define the DB for F-runtime group communication in the work area "F-runtime groups". Proceed as follows:

1. Click "F-runtime groups" in "*Safety Administration Editor*".
2. Select an existing F-DB in the "DB for F-runtime group communication" field or assign a new one.
3. Assign a symbolic name for the F-DB.

## Up-to-dateness of tags read from another F-runtime group

**Note**

Tags read are up-to-date as at the time of the last completed processing cycle of the F-runtime group providing the tags prior to start-up of the F-runtime group reading the tags.

If the tags supplied undergo multiple changes during runtime of the F-runtime group supplying the tags, the F-runtime group reading the tags only receives the last change (see figure below).

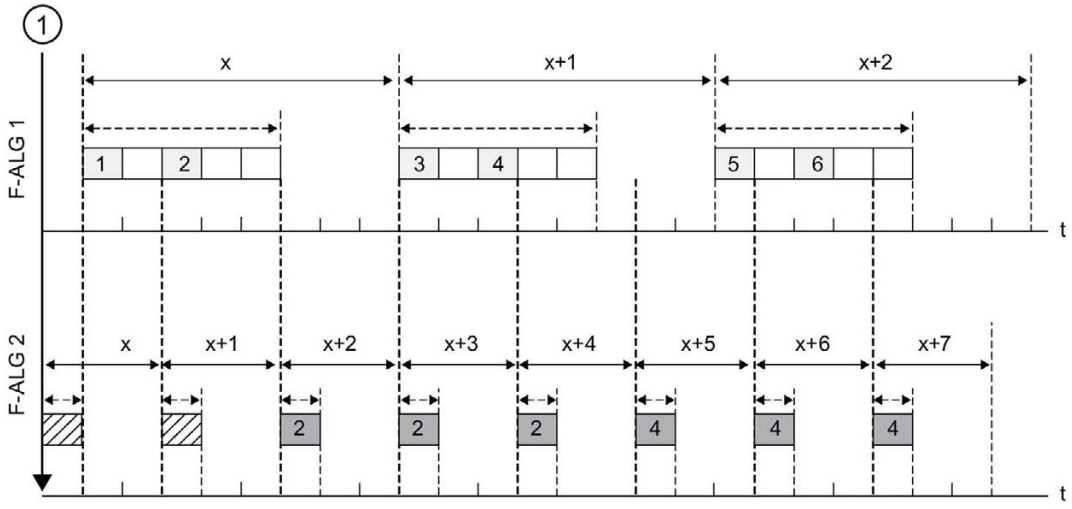
### Assignment of fail-safe values

After F-system start-up, fail-safe values are supplied to the F-runtime group which has read access to tags in the DB for F-runtime group communication of another F-runtime group (for example, F-runtime group 2). These are the values you specified as initial values in the DB for F-runtime group communication of F-runtime group 1.

F-runtime group 2 reads the fail-safe values the first time it is called. The second time the F-runtime group 2 is called, it reads the latest tags if F-runtime group 1 has been processed completely between the two calls of F-runtime group 2. If F-runtime group 1 has not been processed completely, F-runtime group 2 continues to read the fail-safe values until F-runtime group 1 is completely processed.

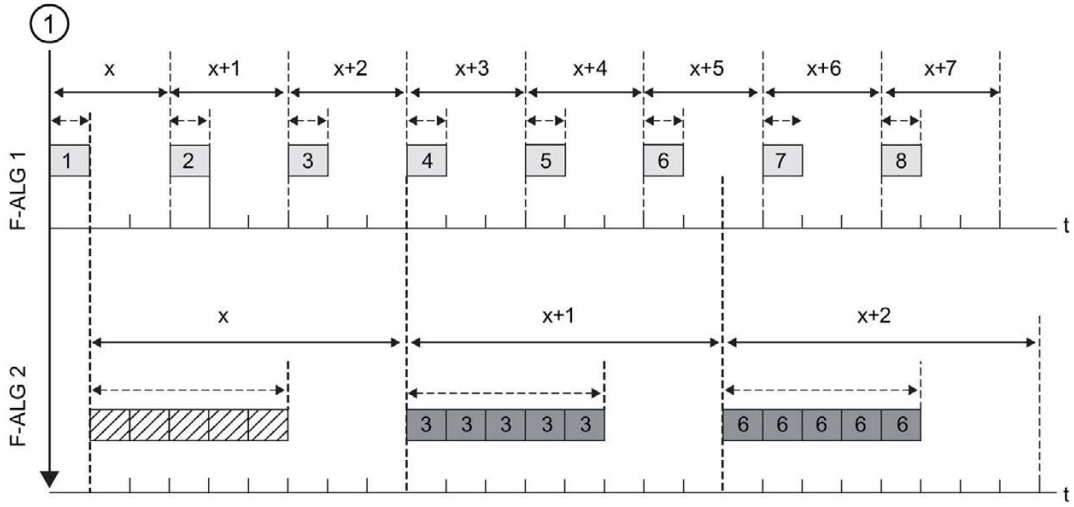
The behavior is illustrated in the two figures below.

**Reading tags from F-runtime group 1, which has a longer OB cycle and lower priority than F-runtime group 2**




| | |
|---|---|
| ① | Startup of F-system |
| ⟷ | Cycle time of the (F-)OB in which the F-runtime group is called. |
| ⟷ (dashed) | Runtime of the F-runtime group |
| 1 | ... Tag of F-runtime group 1, written to DB for F-runtime group communication of F-runtime group 1 |
| 2 | ...Tag of F-runtime group 2, read in DB for F-runtime group communication of F-runtime group 1 |
| ▨ | Initial value in the DB for F-runtime group communication |

**Reading tags from F-runtime group 1, which has a shorter OB cycle and higher priority than F-runtime group 2**




① Startup of F-system

Cycle time of the (F-)OB in which the F-runtime group is called.

Runtime of the F-runtime group

1 ... Tag of F-runtime group 1, written to DB for F-runtime group communication of F-runtime group 1

2 ... Tag of F-runtime group 2, read in DB for F-runtime group communication of F-runtime group 1

Initial value in the DB for F-runtime group communication

## F-runtime group supplying tags is not processed

**Note**

If the F-runtime group whose DB for F-runtime group communication is to supply the tags is not processed (the main safety block of the F-runtime group is not called), the F-CPU goes to STOP mode. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- Error in safety program: cycle time exceeded
- Number of the relevant main safety block (of F-runtime group that is not processed)
- Current cycle time in ms: "0"

## 5.2.5 F-shared DB (S7-300, S7-400)

### Function

The F-shared DB is a fail-safe data block that contains all of the shared data of the safety program and additional information needed by the F-system. The F-shared DB is automatically inserted when the hardware configuration is compiled.

Using its name F_GLOBDB, you can evaluate certain data elements of the safety program in the standard user program.

### Reading an F-shared DB in the standard user program

You can read out the following information in the F-shared DB in the standard user program or on an operator control and monitoring system:

- Operating mode: safety mode or disabled safety mode ("MODE" tag)
- Error information "Error occurred when executing safety program" ("ERROR" tag)
- Collective F-signature ("F_PROG_SIG" tag)
- Compilation date of the safety program ("F_PROG_DAT" tag, DATE_AND_TIME data type)

You use fully qualified access to access these tags (e.g. ""F_GLOBDB".MODE").

## 5.2.6 F-runtime group information DB (S7-1200, S7-1500)

### Introduction

The F-runtime group information DB provides key information on the corresponding F-runtime group and on the safety program as a whole.

The F-runtime group information DB is generated automatically when an F-runtime group is created. A symbol, for example, "RTG1SysInfo", is assigned for the F-runtime group information DB. You have the option of changing the symbol in SAE.

### Information in the F-runtime group information DB

The F-runtime group information DB provides the following information:

| Symbolic name | | Data type | For processing in the safety program | For processing in the standard user program | Description |
|---|---|---|---|---|---|
| MODE | | BOOL | x | x | 1 = Disabled safety mode |
| F_SYSINFO | | | | | |
| | MODE | BOOL | — | x | 1 = Disabled safety mode |
| | TCYC_CURR | DINT | — | x | Current cycle time of the F-runtime group, in ms |
| | TCYC_LONG | DINT | — | x | Longest cycle time of the F-runtime group, in ms |
| | TRTG_CURR | DINT | — | x | Current runtime of the F-runtime group, in ms |
| | TRTG_LONG | DINT | — | x | Longest runtime of the F-runtime group, in ms |
| | T1RTG_CURR | DINT | — | x | Not supported by *STEP 7 Safety V13 SP1*. |
| | T1RTG_LONG | DINT | — | x | Not supported by *STEP 7 Safety V13 SP1*. |
| | F_PROG_SIG | DWORD | — | x | Collective F-signature of the safety program |
| | F_PROG_DAT | DTL | — | x | Compilation date of the safety program |
| | F_RTG_SIG | DWORD | — | x | Collective F-signature of the F-runtime group |
| | F_RTG_DAT | DTL | — | x | Compilation date of the F-runtime group |
| | VERS_S7SAF | DWORD | — | x | Version identifier for *STEP 7 Safety* |

You access the content of the F-runtime group information DB with fully qualified addressing. Either collectively with the F_SYSINFO PLC data type, for example, "RTG1SysInfo.F_SYSINFO", provided by the F-system or individual information, for example, "RTG1SysInfo.F_SYSINFO.MODE".

## 5.2.7 Deleting an F-runtime group

### Deleting an F-runtime group

To delete an F-runtime group, proceed as follows:

1. In the area navigation of the *Safety Administration Editor*, click on the F-runtime group to be deleted.
2. Select the "Delete F-runtime group" button in the work area.
3. Confirm the dialog with "Yes".
4. Compile the safety program (Page 265) (menu command "Edit > Compile") to put your changes into effect.

The assignment of the F-blocks to an F-runtime group (to the calling block of the main safety block) is deleted. However, the F-blocks continue to exist.

**Note**

If you want to delete your safety program, delete all F-blocks outside the System blocks folder in the project tree.

F-blocks that do not allow deletion are deleted by recompiling the safety program or deactivating F-capability for the F-CPU (see Configuring an F-CPU (Page 42)).

## 5.2.8 Changing the F-runtime group (S7-300, S7-400)

### Changing an F-runtime group

You can make the following changes for each F-runtime group in your safety program in the corresponding "F-runtime group" work area:

- Specify another block as the calling block of the main safety block.
- Specify another F-FB or F-FC as main safety block.
- Enter a different or new I-DB for the main safety block.
- Change the value for the maximum cycle time of the F-runtime group
- Specify another DB as a data block for F-runtime group communication.

## 5.2.9 Changing the F-runtime group (S7-1200, S7-1500)

### Changing an F-runtime group

You can make the following changes for each F-runtime group in your safety program in the corresponding "F-runtime group" work area:

- Change the name, number, cycle time, phase shift and priority of the F-OB.
- Specify another F-FB or F-FC as main safety block.
- Enter a different or new I-DB for the main safety block.
- Change the value for the maximum cycle time and the cycle time warning limit of the F-runtime group.
- Assign another symbolic name for the F-runtime group information DB.

### Copying F-OB

The F-OB in the "Program blocks" folder reacts like a standard OB.

You should not copy an F-OB to another F-CPU. Copying an F-OB to another F-CPU terminates the F-OB connection to the main safety block. The F-OB is stored in the destination F-CPU as a block. It is not connected to an F-runtime group. You can UNDO these actions.

# 5.3 Creating F-blocks in FBD / LAD

## 5.3.1 Creating F-blocks

### Introduction

In order to create F-FBs, F-FCs, and F-DBs for the safety program, you should follow the same basic procedure as for standard blocks. In the following, only the deviations from the procedure for standard blocks are presented.

### Creating F-FBs, F-FCs, and F-DBs

You create F-blocks in the same way as for standard blocks. Proceed as follows:

1. Double-click on "Add new block" under "Program blocks" in the project tree.
2. In the dialog that appears, specify the type, name, and language and select the "Create F-block" check box. (If you do not select the check box, a standard block is created.)
3. After the dialog is confirmed, the F-block is opened in the *Program editor*.

### Note the following

Note the following important instructions:

**Note**

The main safety block must not contain any parameters, as they cannot be provided with actual values when the block is called.

**Note**

Editing the start values in instance DBs of F-FBs is not permitted online or offline and can cause the F-CPU to switch to STOP mode.

**Note**

It is not permitted to access static local data in single instances or multi-instances of other F-FBs.

**Note**

Outputs of F-FCs must always be initialized.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

If you wish to assign an operand from the data area (data block) to a formal parameter of an F-FC as an actual parameter, you must use fully qualified DB access. (S7-300, S7-400)

**Note**

Note that access to the inputs in an F-FB/F-FC is read-only, while access to the outputs is write-only.

Use an in/out if you wish to have both read and write access.

**Note**

For greater clarity, assign unique symbolic names to the F-blocks you have created. These symbolic names appear in the project tree and in the block interface. Symbolic names are assigned in the same way as for standard blocks.

## Copying/pasting F-blocks

You can copy F-FBs, F-FCs, and F-DBs in exactly the same was as blocks of the standard user program.

(S7-1200, S7-1500) Information on copying an F-OB is provided in Changing the F-runtime group (S7-1200, S7-1500) (Page 117).

**Exception:**

You must not copy blocks from the folder "Program blocks > System blocks".

## Nesting depth for F-FBs and F-FCs

We recommend that you do not exceed a nesting depth of 8 levels.

If you exceed this, a warning is reported during compilation for S7-1200/1500 F-CPUs.

## 5.3.2 Using libraries

### Introduction

As with standard blocks, you have the option of storing F-blocks which you wish to reuse as master copy or types in global libraries or in the project library.

Additional information can be found in the *STEP 7* help in "Using libraries".

### Reusing F-blocks which have already been tested and accepted

Please note the following when reusing F-blocks which have already been tested and if applicable also accepted:

- Document the signature and, if applicable, also the initial value signature of the F-block for S7-300/400 F-CPUs after you have finished compiling, testing, and accepting the F-block, if necessary.
- Document the version set for "Instructions (without their own version)" in the Safety Administration Editor under "Settings"/"System library elements used in safety program".

  For documentation purposes, you can create a safety summary of the safety program you used to create the F-block.
- When using the F-block again, make sure that the signature and, if applicable, the initial value signature of the F-block for S7-300/400 F-CPUs are unchanged.

For F-blocks with instructions for which a version is indicated in the "Version" column of the "Instructions" task card, you must also note the following:

- Document the versions of these instructions that were set at the time the F-block was compiled and on the basis of which the F-block was tested and, if applicable, approved.

  You can use the safety summary for documentation purposes.
- When using the F-block again make sure that the documented versions are set for these instructions in the "Instructions" task card.
- If versions other than the documented versions are set for these instructions, you must check yourself whether the F-block still works with these versions as intended.

  To do so, identify any differences between the versions based on the online help on the instructions.

  If there are differences you have to retest the F-block and have it accepted if necessary. Additional information on instruction versions can be found in the help on *STEP 7* in "Basics for instruction versions".

To help you manage F-blocks that you wish to reuse and that have already been tested and if applicable accepted, we recommend that you save these blocks as types in a global library.

# 5.4 Programming startup protection

## Introduction

**⚠ WARNING**

When an F-CPU is switched from STOP to RUN mode, the standard user program starts up in the usual way. When the safety program is started up, all F-DBs are initialized with the values from the load memory - as is the case with a cold restart. This means that saved error information is lost.

The F-system automatically reintegrates the F-I/O.

An operating error or an internal error can also trigger a startup of the safety program with the values from the load memory. If your process does not allow such a startup, you must program a restart/startup protection in the safety program: The output of process data must be blocked until manually enabled. This enable must not occur until it is safe to output process data and faults have been corrected. *(S008)*

## Example of restart/startup prevention

In order to implement a restart/startup prevention, it must be possible to detect a startup. To detect a startup, you declare a tag of data type BOOL with initial value "TRUE" in an F-DB.

Block the output of process data when this tag has the value "1," for example, by passivating F-I/O using the PASS_ON tag in the F-I/O DB.

To manually enable the output of process data, you reset this tag by means of a user acknowledgment.

## See also

Implementing User Acknowledgment in the Safety Program of the F-CPU of a DP Master or IO controller (Page 151)

Implementing user acknowledgment in the safety program of the F-CPU of a I-slave or I-device (S7-300, S7-400, S7-1500) (Page 156)

F-I/O DB (Page 129)

# F-I/O access

# 6

## 6.1 F-I/O access

### Overview

The following describes how you can access the F-I/O and the special characteristics you must consider when programming this access.

### Access via the process image

As with standard I/O, you access F-I/O (e.g., ET 200MP fail-safe modules) via the **process image** (PII and PIQ).

Access here is only permitted via the corresponding channel for the specified data type.

Example: To access input channels of data type BOOL in the process input image of F-I/O, you must use the "input (bit)" (I x.y) unit. Access to the 16 consecutive input channels of the data type BOOL via the unit "input word" (IW x) is not possible.

Direct I/O access is not permitted. Inputs or outputs of the F-I/O to which you have access must be enabled in the hardware configuration. You must not access the channel value/value status of an F-I/O from various F-runtime groups. The first programmed access defines the assignment for the F-runtime group.

The process image of the inputs is updated at the start of the F-runtime group. The process image of the outputs is updated at the end of the F-runtime group (see Program structure of the safety program (S7-300, S7-400) (Page 85) or Program structure of the safety program (S7-1200, S7-1500) (Page 87)). For additional information on updating the process image, refer to the note in Data Transfer from the Safety Program to the Standard User Program (Page 160).

The actual communication between the F-CPU (process image) and the F-I/O for the purpose of updating the process image takes place in the background using a special safety protocol in accordance with PROFIsafe.

**⚠ WARNING**

If you use an additional component between the F-CPU (S7-300/400) and the F-I/O that copies the safety frame in accordance with PROFIsafe between the F-CPU (S7-300/400) and F-I/O per user program, you must test all safety functions affected by the copy function whenever you change the user-programmed copy function. *(S049)*

**Note**

Due to the special safety protocol, the F-I/O occupies a larger area of the process image than is required for the existing and enabled channels on the F-I/O (channel values and value status). To find the area of the process image where the channels are stored, refer to the relevant *manuals for the F-I/O*. When the process image is accessed in the safety program, only the channels that are actually present may be accessed.

Note that for certain F-I/O (such as ET 200SP fail-safe modules or ET 200MP fail-safe modules), a "1oo2 evaluation of the sensors can be specified. To find out which of the channels combined by the "1oo2 evaluation of the sensors" you can access in the safety program, refer to the relevant manuals for the F-I/O.

## See also

Safety-Related I-Slave-Slave Communication - F-I/O Access (Page 210)

Value status (S7-1200, S7-1500) (Page 124)

# 6.2 Value status (S7-1200, S7-1500)

## Properties

The value status is additional binary information for the channel value of an F-I/O. The value status is entered in the process image input (PII).

The value status is supported by ET 200MP, ET 200SP, ET 200S, ET 200iSP, ET 200pro, S7-1200 fail-safe modules or S7-300 F-SMs and fail-safe GSD based I/O devices which support the "RIOforFA-Safety" profile.

We recommend you assign the name for the channel value amended with "_VS" as a symbolic name for the value status, for example, "TagIn_1_VS".

The value status provides information on the validity of the corresponding channel value:

- 1: A valid process value is output for the channel.
- 0: A fail-safe value is output for the channel.

The channel value and value status of an F-I/O can only be accessed from the same F-runtime group.

## Location of value status bits in the PII for F-I/O with digital inputs

The value status bits come straight after the channel values in the PII.

Table 6- 1 Example: Address assignment in PII for F-I/O with 16 digital input channels

| Byte in the F-CPU | Assigned bits in F-CPU per F-I/O: | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| x + 0 | $DI_7$ | $DI_6$ | $DI_5$ | $DI_4$ | $DI_3$ | $DI_2$ | $DI_1$ | $DI_0$ |
| x + 1 | $DI_{15}$ | $DI_{14}$ | $DI_{13}$ | $DI_{12}$ | $DI_{11}$ | $DI_{10}$ | $DI_9$ | $DI_8$ |
| x + 2 | Value status $DI_7$ | Value status $DI_6$ | Value status $DI_5$ | Value status $DI_4$ | Value status $DI_3$ | Value status $DI_2$ | Value status $DI_1$ | Value status $DI_0$ |
| x + 3 | Value status $DI_{15}$ | Value status $DI_{14}$ | Value status $DI_{13}$ | Value status $DI_{12}$ | Value status $DI_{11}$ | Value status $DI_{10}$ | Value status $DI_9$ | Value status $DI_8$ |

x = Module start address

The location of the channel values in the PII can be found in the device manual for the F-I/O.

## Location of value status bits in the PII for F-I/O with digital outputs

The value status bits in the PII are mapped with the same structure as the channel values in the PIQ.

Table 6- 2 Example: Address assignment in PIQ for F-I/O with 4 digital output channels

| Byte in the F-CPU | Assigned bits in F-CPU per F-I/O: | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| x + 0 | — | — | — | — | $DQ_3$ | $DQ_2$ | $DQ_1$ | $DQ_0$ |

x = Module start address

Table 6- 3 Example: Address assignment in PII for F-I/O with 4 digital output channels

| Byte in the F-CPU | Assigned bits in F-CPU per F-I/O: | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| x + 0 | — | — | — | — | Value status $DQ_3$ | Value status $DQ_2$ | Value status $DQ_1$ | Value status $DQ_0$ |

x = Module start address

The location of the channel values in the PIQ can be found in the device manual for the F-I/O.

## Location of value status bits in the PII for F-I/O with digital outputs and digital inputs

The value status bits come directly after the channel values in the PII in the following order:

- Value status bits for the digital inputs
- Value status bits for the digital outputs

Table 6- 4 Example: Address assignment in PIQ for F-I/O with 2 digital input channels and 1 digital output channel

| Byte in the F-CPU | Assigned bits in F-CPU per F-I/O: | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| x + 0 | — | — | — | — | — | — | — | $DQ_0$ |

x = Module start address

Table 6- 5 Example: Address assignment in PII for F-I/O with 2 digital input channels and 1 digital output channel

| Byte in the F-CPU | Assigned bits in F-CPU per F-I/O: | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| x + 0 | — | — | — | — | — | — | $DI_1$ | $DI_0$ |
| x + 1 | — | — | — | — | — | — | Value status $DI_1$ | Value status $DI_0$ |
| x + 2 | — | — | — | — | — | — | — | Value status $DQ_0$ |

x = Module start address

The location of the channel values in the PII and PIQ can be found in the device manual for the F-I/O.

## Location of value status bits in the PII for F-I/O with analog inputs

The value status bits come directly after the channel values in the PII.

Table 6- 6 Example: Address assignment in PII for F-I/O with 6 analog input channels (data type INT)

| Byte in the F-CPU | Assigned bytes/bits in the F-CPU per F-I/O: | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| x + 0 | Channel value $AI_0$ | | | | | | | |
| ... | ... | | | | | | | |
| x + 10 | Channel value $AI_5$ | | | | | | | |
| x + 12 | — | — | Value status $AI_5$ | Value status $AI_4$ | Value status $AI_3$ | Value status $AI_2$ | Value status $AI_1$ | Value status $AI_0$ |

x = Module start address

The location of the channel values in the PII can be found in the device manual for the F-I/O.

## Location of value status bits in the PII for F-I/O with analog outputs

The value status bits are mapped in the PII.

Table 6- 7 Example: Address assignment in PIQ for F-I/O with 6 analog output channels (data type INT)

| Byte in the F-CPU | Assigned bytes in the F-CPU per F-I/O: | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| x + 0 | Channel value $AO_0$ | | | | | | | |
| ... | ... | | | | | | | |
| x + 10 | Channel value $AO_5$ | | | | | | | |

x = Module start address

Table 6- 8 Example: Address assignment in PII for F-I/O with 6 analog output channels (data type INT)

| Byte in the F-CPU | Assigned bits in F-CPU per F-I/O: | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| x + 0 | — | — | Value status $AO_5$ | Value status $AO_4$ | Value status $AO_3$ | Value status $AO_2$ | Value status $AO_1$ | Value status $AO_0$ |

x = Module start address

The location of the channel values in the PIQ can be found in the device manual for the F-I/O.

# 6.3 Process Data or Fail-Safe Values

## When are fail-safe values used?

The safety function requires that fail-safe values (0) be used instead of process data for passivation of the entire F-I/O or individual channels of an F-I/O in the following cases. This applies both to digital channels (data type BOOL) as well as for analog channels (data type INT or DINT):

- When the F-system starts up
- When errors occur during safety-related communication (communication errors) between the F-CPU and F-I/O using the safety protocol in accordance with PROFIsafe
- When F-I/O faults and channel faults occur (such as wire break, short circuit, and discrepancy errors)
- As long as you enable passivation of the F-I/O with PASS_ON = 1 in the F-I/O DB (see below)

## Fail-safe value output for F-I/O/channels of an F-I/O

When **passivation** occurs for an **F-I/O with inputs**, the F-system provides the safety program with fail-safe values (0) in the PII instead of the process data pending at the fail-safe inputs of the F-I/O.

The overflow or underflow of a channel of the **SM 336; AI 6 x 13Bit** or **SM 336; F-AI 6 x 0/4 ... 20 mA HART** is recognized by the F-system as an F-I/O/channel fault. The fail-safe value 0 is provided in place of $7FFF_H$ (for overflow) or $8000_H$ (for underflow) in the PII for the safety program.

If you want to process fail-safe values other than "0" in the safety program for an F-I/O with inputs **for analog channels of data type INT or DINT**, you can assign individual fail-safe values for QBAD = 1 and value status = 0 or QBAD_I_xx/QBAD_O_xx = 1 (instructions JMP/JMPN, LABEL and MOVE).

**⚠ WARNING**

For an F-I/O with digital input channels (data type BOOL), the value provided in the PII must always be processed in the safety program, regardless of the value status or QBAD/QBAD_I_xx. *(S009)*

When passivation occurs in an F-I/O with outputs, the F-system transfers fail-safe values (0) to the fail-safe outputs instead of the output values in the PIQ provided by the safety program. The F-system overwrites the associated PIQ with fail-safe values (0).

### Reintegration of F-I/O/channels of an F-I/O

The switchover from fail-safe values (0) to process data (**reintegration of an F-I/O**) takes place **automatically** or following **user acknowledgment** in the F-I/O DB. The reintegration method depends on the following:

- The reason for passivation of the F-I/O or channels of the F-I/O
- The parameter assignment to be implemented by you in the F-I/O DB (Page 129) or the parameter assignment for the ET 200MP F-modules (S7-1500)/S7-1200 F-modules and possibly GSD based I/O devices according to RIOforFA-Safety profile

**Note**

Note that for channel faults in the F-I/O, channel-granular passivation takes place if configured accordingly in the *hardware and network editor*. For the concerned channels, fail-safe values (0) are output.

Reintegration after channel faults reintegrates all channels whose faults were eliminated; faulty channels remain passivated.

### See also

Configuring F-I/O (Page 47)

## 6.4 F-I/O DB

### Introduction

An F-I/O DB is automatically generated for each F-I/O (in safety mode) when the F-I/O is configured in the *hardware and network editor*. The F-I/O DB contains tags that you can evaluate or can/must write to in the safety program. It is not permitted to change the initial values of the tags directly in the F-I/O DB. When an F-I/O is deleted, the associated F-I/O DB is also deleted.

### Access to an F-I/O DB

You access tags of the F-I/O DB for the following reasons:

- For reintegration of F-I/O after communication errors, F-I/O faults, or channel faults
- If you want to passivate the F-I/O depending on particular states of your safety program (for example, group passivation)
- For changing parameters for fail-safe GSD based DP slaves/GSD based I/O devices
- If you want to evaluate whether fail-safe values or process data are output

### 6.4.1 Tags of the F-I/O DB

The following table shows the tags of an F-I/O DB:

| | Tag | Data type | Function | Initial value |
|---|---|---|---|---|
| **Tags that you can or must write** | PASS_ON | BOOL | 1=enable passivation | 0 |
| | ACK_NEC | BOOL | 1=acknowledgment for reintegration required in the event of F-I/O or channel faults | 1 |
| | ACK_REI | BOOL | 1=acknowledgment for reintegration | 0 |
| | IPAR_EN | BOOL | Tag for parameter reassignment of fail-safe GSD based DP slaves/GSD based I/O devices or, in the case of SM 336; F-AI 6 x 0/4 ... 20 mA HART, for enabling HART communication | 0 |
| **Tags that you can evaluate** | PASS_OUT | BOOL | Passivation output | 1 |
| | QBAD | BOOL | 1=Fail-safe values are output | 1 |
| | ACK_REQ | BOOL | 1=Acknowledgment request for reintegration | 0 |
| | IPAR_OK | BOOL | Tag for parameter reassignment of fail-safe GSD based DP slaves/GSD based I/O devices or, in the case of SM 336; F-AI 6 x 0/4 ... 20 mA HART, for enabling HART communication | 0 |
| | DIAG | BYTE | Service information | |
| | QBAD_I_xx | BOOL | 1=fail-safe values are output to input channel xx (S7-300/400) | 1 |
| | QBAD_O_xx | BOOL | 1=fail-safe values are output to output channel xx (S7-300/400) | 1 |

### Differences in evaluation in S7-1200/1500 F-CPUs and S7-300/400

The following table describes the differences in the evaluation of tags of the F-I/O DB and the value status depending on the F-I/O and F-CPU used.

| Tag in the F-I/O DB or value status | F-I/O with "RIOforFA-Safety" profile with S7-1200/1500 F-CPU | F-I/O without profile "RIOforFA-Safety" with S7-1500 F-CPU | F-I/O with S7-300/400 F-CPU |
|---|---|---|---|
| ACK_NEC | —[2] | x | x |
| QBAD | x[3] | x | x |
| PASS_OUT | x[3] | x | x |
| QBAD_I_xx[1] | — | — | x |
| QBAD_O_xx[1] | — | — | x |
| Wertstatus[1] | x | x | — |

1 QBAD_I_xx and QBAD_O_xx indicate the validity of the channel value on a channel-granular basis and are therefore equivalent to the value status in S7-1200. Value status or QBAD_I_xx and QBAD_O_xx are not available for fail-safe GSD based DP slaves or fail-safe GSD based I/O devices without the "RIOforFA-Safety" profile.

2 Via F-I/O configuration; for ET 200MP/S7-1200 fail-safe modules using the "Channel failure acknowledge" parameter

3 For details about the characteristics, see "PASS_OUT/QBAD/QBAD_I_xx/QBAD_O_xx and value status" below

#### 6.4.1.1 PASS_ON

The PASS_ON tag allows you to enable passivation of an F-I/O, for example, depending on particular states in your safety program.

Using the PASS_ON tag in the F-I/O DB, you can passivate F-I/O; channel-granular passivation is not possible.

As long as PASS_ON = 1, **passivation** of the associated F-I/O occurs.

#### 6.4.1.2 ACK_NEC

If an F-I/O fault is detected by the F-I/O, **passivation** of the relevant F-I/O occurs. If channel faults are detected and channel granular passivation is configured, the relevant channels are passivated. If passivation of the entire F-I/O is configured, all channels of the relevant F-I/O are passivated. Once the F-I/O fault or channel fault has been eliminated, **reintegration** of the relevant F-I/O occurs, depending on ACK_NEC:

- With ACK_NEC = 0, you can assign an **automatic reintegration**.
- With ACK_NEC = 1, you can assign a **reintegration** by **user acknowledgment**.

**WARNING**

Assignment of the ACK_NEC = 0 tag is only allowed if automatic reintegration is permitted for the relevant process from a safety standpoint. *(S010)*

**Note**

The initial value for ACK_NEC is 1 after creation of the F-I/O DB. If you do not require automatic reintegration, you do not have to write ACK_NEC.

#### 6.4.1.3 ACK_REI

When the F-system detects a communication error or an F-I/O fault for an F-I/O, the relevant F-I/O is passivated. If channel faults are detected and channel granular passivation is configured, the relevant channels are passivated. If passivation of the entire F-I/O is configured, all channels of the relevant F-I/O are passivated. **Reintegration** of the F-I/O/channels of the F-I/O after elimination of faults requires a **user acknowledgment** with a positive edge at the ACK_REI tag of the F-I/O DB:

- After every communication error
- After F-I/O or channel faults only with parameter assignment "Channel failure acknowledge =

  Manually" or ACK_NEC = 1

Reintegration after channel faults reintegrates all channels whose faults were eliminated.

Acknowledgment is not possible until tag ACK_REQ = 1.

In your safety program, you must provide a user acknowledgment by means of the ACK_REI tag for each F-I/O.

**WARNING**

For the user acknowledgment, you must interconnect the ACK_REI tag of the F-I/O DB with a signal generated by an operator input. An interconnection with an automatically generated signal is not permitted. *(S011)*

**Note**

Alternatively, you can use the "ACK_GL" instruction to carry out reintegration of the F-I/O following communication errors or F-I/O/channel faults (ACK_GL: Global acknowledgment of all F-I/O in an F-runtime group (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 407)).

#### 6.4.1.4 IPAR_EN

The IPAR_EN tag corresponds to the iPar_EN_C tag in the PROFIsafe bus profile as of PROFIsafe Specification V1.20 and higher.

**Fail-safe GSD based DP slaves/GSD based I/O devices**

To find out when this tag must be set or reset when parameters of fail-safe GSD based DP slaves/GSD based I/O devices are reassigned, consult the PROFIsafe Specification V1.20 or higher or the documentation for the fail-safe GSD based DP slave/GSD based I/O device.

Note that IPAR_EN = 1 does not trigger passivation of the relevant F-I/O.

If passivation is to occur when IPAR_EN = 1, you must also set tag PASS_ON = 1.

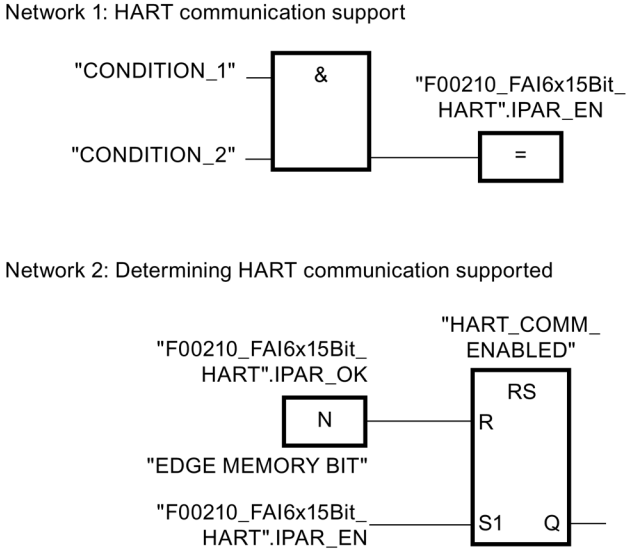
**HART communication with SM 336; F-AI 6 x 0/4 ... 20 mA HART**

If you set the IPAR_EN tag to "1" while parameter "HART_Tor" = "switchable" is assigned, the HART communication for the SM 336; F-AI 6 x 0/4 ... 20 mA HART is enabled. Setting this tag to "0" disables the HART communication. The F-SM acknowledges the enabled or disabled HART communication with tag IPAR_OK = 1 or 0.

Enable HART communication only when your system is in a status, in which any reassignment of parameters for an associated HART device can be done without any risk.

If you want to evaluate the "HART communication enabled" status in your safety program, e.g., for the purpose of programming interlocks, you must build up the information as shown in the following example. This is necessary to ensure that the information is properly available even if communication errors occur while the HART communication is enabled with IPAR_EN = 1. Only change the status of the IPAR_EN tag during this evaluation if there is no passivation due to a communication error or F-I/O or channel fault.

### Example of enabling HART communication




Additional information on HART communication with SM 336; F-AI 6 x 0/4 ... 20 mA HART can be found in the Automation System S7-300, ET 200M Distributed I/O System manual, Fail-Safe Signal Modules (http://support.automation.siemens.com/WW/view/en/19026151) manual and in the online help on the module.

### 6.4.1.5 PASS_OUT/QBAD/QBAD_I_xx/QBAD_O_xx and value status

The following table set outs differences in the reaction of the channel values and PASS_OUT, QBAD, QBAD_I_xx/QBAD_O_xx tags and of the value status depending on the F-I/O and F-CPU used.

| **Fail-safe value output after...** | **F-I/O with "RIOforFA-Safety" profile with S7-1200/1500 F-CPU** | **F-I/O without profile "RIOforFA-Safety" with S7-1500 F-CPUs** | **F-I/O with S7-300/400 F-CPU** |
|---|---|---|---|
| Startup of F-system<br>Communication errors<br>F-I/O faults<br>Channel faults in configuration of passivation for complete F-I/O | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0* | | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>QBAD_I_xx and QBAD_O_xx = 1* |
| Channel faults during configuration of channel-granular passivation | QBAD and PASS_OUT **unchanged**<br>For the **affected** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0 | QBAD and PASS_OUT = 1<br>For the **affected** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0* | QBAD and PASS_OUT = 1<br>For the **affected** channels:<br>Channel value = fail-safe value (0)<br>QBAD_I_xx and QBAD_O_xx = 1* |
| As long as passivation of the F-I/O is activated in the F-I/O DB with PASS_ON = 1 | QBAD = 1, PASS_OUT **unchanged**<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0* | | QBAD = 1, PASS_OUT **unchanged**<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>QBAD_I_xx and QBAD_O_xx = 1* |

* Value status or QBAD_I_xx and QBAD_O_xx are not available for fail-safe GSD based DP slaves and fail-safe GSD based I/O devices without the "RIOforFA-Safety" profile.

#### 6.4.1.6 ACK_REQ

When the F-system detects a communication error or an F-I/O fault or channel fault for an F-I/O, the relevant F-I/O or individual channels of the F-I/O are passivated. ACK_REQ = 1 signals that **user acknowledgment** is required for reintegration of the relevant F-I/O or channels of the F-I/O.

The F-system sets ACK_REQ = 1 as soon as the fault has been eliminated and user acknowledgment is possible. For channel-granular passivation, the F-system sets ACK_REQ = 1 as soon as the channel fault is corrected. User acknowledgment is possible for this fault. Once acknowledgment has occurred, the F-system resets ACK_REQ to 0.

**Note**

For F-I/O with outputs, acknowledgment after F-I/O or channel faults may only be possible some minutes after the fault has been eliminated, until the necessary test signal is applied (see *F-I/O manuals*).

#### 6.4.1.7 IPAR_OK

The IPAR_OK tag corresponds to the iPar_OK_S tag in the PROFIsafe bus profile, PROFIsafe Specification V1.20 and higher.

**Fail-safe GSD based DP slaves/GSD based I/O devices**

To find out how to evaluate this tag when parameters of fail-safe GSD based DP slaves or GSD based I/O devices are reassigned, consult the PROFIsafe Specification V1.20 or higher or the documentation for the fail-safe GSD based DP slave/GSD based I/O device.

**For HART communication with SM 336; F-AI 6 x 0/4 ... 20 mA HART see Chapter** IPAR_EN (Page 133).

### 6.4.1.8 DIAG

The DIAG tag provides non-fail-safe information (1 byte) about errors or faults that have occurred for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until you perform an acknowledgment with the ACK_REI tag or until automatic reintegration takes place.

#### Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Timeout detected by F-I/O | The PROFIBUS/PROFINET connection between F-CPU and F-I/O is faulty.<br>The value of the F-monitoring time for the F-I/O is set too low.<br>The F-I/O are receiving invalid parameter assignment data<br>or | • Check the PROFIBUS/PROFINET connection and ensure that there are no external sources of interference.<br>• Check the parameter assignment of the F-I/O. If necessary, set a higher value for the monitoring time. Recompile the hardware configuration, and download it to the F-CPU. Recompile the safety program.<br>• Check the diagnostics buffer of the F-I/O.<br>• Turn the power of the F-I/O off and back on. |
| | | Internal F-I/O fault<br>or | Replace F-I/O |
| | | Internal F-CPU fault | Replace F-CPU |
| Bit 1 | F-I/O fault or channel fault detected by F-I/O[1] | See *F-I/O manuals* | See *F-I/O manuals* |
| Bit 2 | CRC error or sequence number error detected by F-I/O | See description for bit 0 | See description for bit 0 |
| Bit 3 | Reserved | — | — |
| Bit 4 | Timeout detected by F-system | See description for bit 0 | See description for bit 0 |
| Bit 5 | Sequence number error detected by F-system[2] | See description for bit 0 | See description for bit 0 |
| Bit 6 | CRC-error detected by F-system | See description for bit 0 | See description for bit 0 |
| Bit 7 | Addressing error[3] | — | Contact Service & Support |

[1] Not for F-I/O that support the "RIOforFA-Safety" profile.

[2] For S7-300/400 F-CPUs only

[3] For S7-1200/1500 F-CPUs only

## 6.4.2 Accessing tags of the F-I/O DB

### Name of the F-I/O DB

An F-I/O DB is automatically generated for each F-I/O when the F-I/O is configured in the *hardware and network editor* and a name is generated at the same time.

The name is formed by combining the fixed prefix "F", the start address of the F-I/O, and the names entered in the properties for the F-I/O in the *hardware and network editor* (example: F00005_4_8_F_DI_DC24V_1).

### Changing the name and number of the F-I/O DB

The name and number of the F-I/O DB are assigned automatically when the F-I/O are configured. You can change the number in the "Properties" tab of the associated F-I/O. You can also change the name and number in the project tree.

### Rule for accessing tags of the F-I/O DB

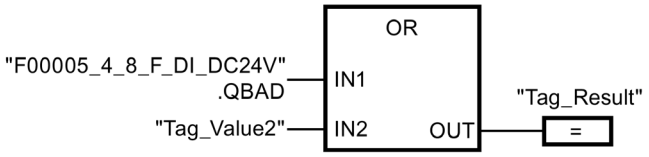
Tags of the F-I/O DB of an F-I/O can only be accessed from the F-runtime group from which the channels of this F-I/O are accessed (if access is made).

### "Fully qualified DB access"

You can access the tags of the F-I/O DB with "fully qualified DB access" (that is, by specifying the name of the F-I/O DB and by specifying the name of the tag).

### Example of evaluating the QBAD tag

Network 4: fully qualified access to the QBAD tag



# 6.5 Passivation and reintegration of F-I/O

## Overview

In the following you can find information on passivation and reintegration of F-I/O.

## Signal sequence charts

The signal sequences presented below represent typical signal sequences for the indicated behavior.

Actual signal sequences and, in particular, the relative position of the status change of individual signals can deviate from the given signal sequences within the scope of known "fuzzy" cyclic program execution factors, depending on the following:

- The F-I/O used
- The F-CPU used
- The cycle time of the (F-)OB in which the associated F-runtime group is called
- The target rotation time of the PROFIBUS DP or the update time of the PROFINET IO

**Note**

The signal sequences shown refer to the status of signals in the user's safety program.

Contrary to what is shown in the signal charts, the status changes shown between process data and fail-safe values that are transmitted to the fail-safe outputs ("To outputs" signal sequence) can occur before the status change of the associated QBAD signal or value status.

## 6.5.1 After startup of F-system

### Behavior after a startup

| Fail-safe value output after startup of the F-system | F-I/O with "RIOforFA-Safety" profile with S7-1200/1500 F-CPU | F-I/O without profile "RIOforFA-Safety" with S7-1500 F-CPU | Every F-I/O with S7-300/400 F-CPU |
|---|---|---|---|
| Passivation of the entire F-I/O occurs during startup. | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0* | | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>QBAD_I_xx and QBAD_O_xx = 1* |

* Value status or QBAD_I_xx and QBAD_O_xx are not available for fail-safe GSD based DP slaves and fail-safe GSD based I/O devices without the "RIOforFA-Safety" profile.

### Reintegration of F-I/O

**Reintegration** of the F-I/O, i.e. the provision of process values in the PII or the transfer of process values provided in the PIQ to the fail-safe outputs, takes place **automatically**, regardless of the setting at the tag ACK_NEC or the configuration "Channel failure acknowledge", no sooner than the second cycle of the F-runtime group after startup of the F-system.

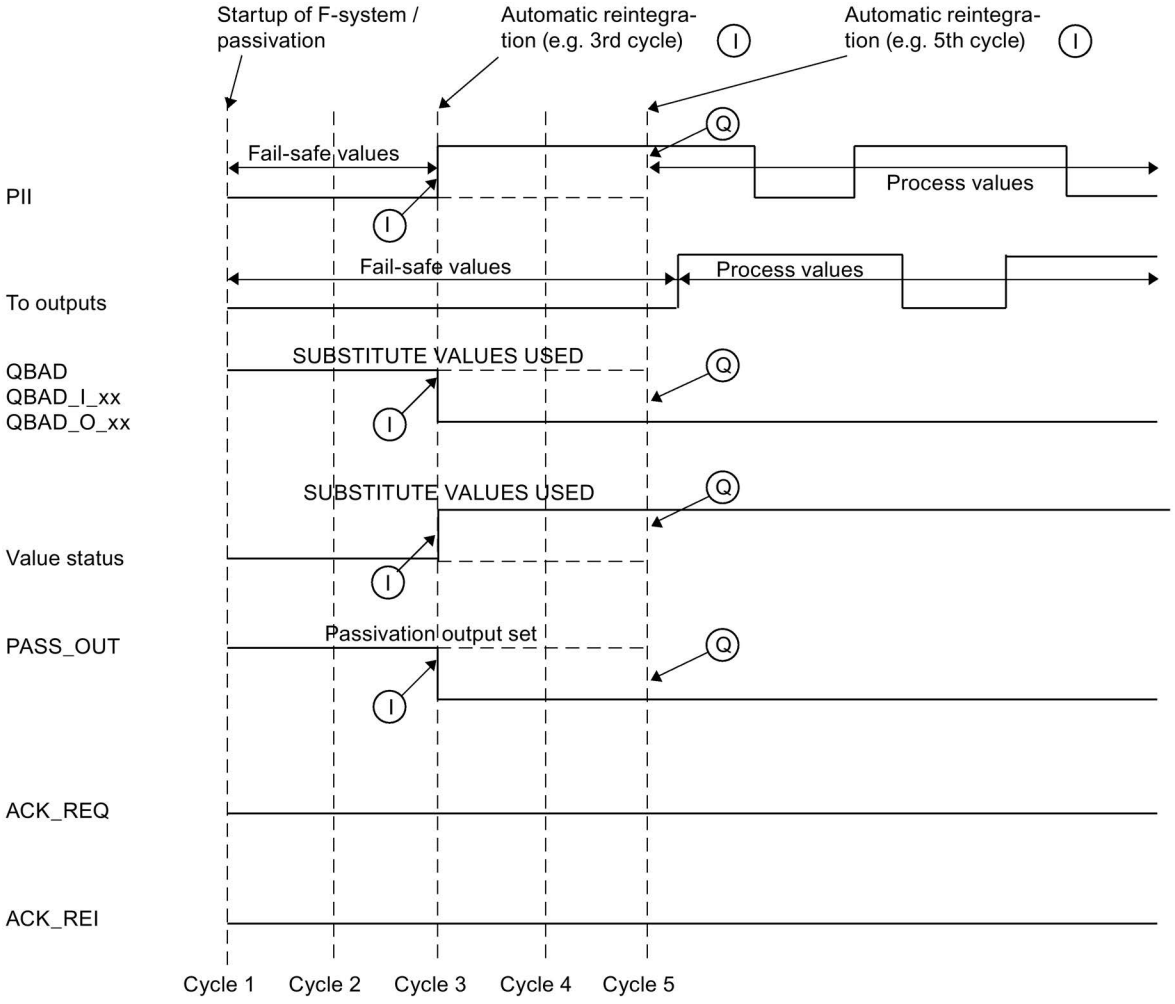
You can find additional information on pending F-communication, F-I/O or channel errors during startup of the F-system in the sections After communication errors (Page 142) and After F-I/O or channel faults (Page 144).

For fail-safe GSD based I/O devices with "RIOforFA-Safety" profile, consult the respective documentation for the fail-safe GSD based I/O device .

Depending on the F-I/O you are using and the cycle time of the F-runtime group and PROFIBUS DP/PROFINET IO, several cycles of the F-runtime group can elapse before reintegration occurs.

If communication between the F-CPU and F-I/O takes longer to establish than the F-monitoring time set in the properties for the F-I/O, automatic reintegration does not take place.

## Signal sequence for passivation and reintegration of F-I/O after F-system startup




(I) For F-I/O with inputs

(Q) For F-I/O with outputs and F-I/O with inputs and outputs

**WARNING**

When an F-CPU is switched from STOP to RUN mode, the standard user program starts up in the usual way. When the safety program is started up, all F-DBs are initialized with the values from the load memory - as is the case with a cold restart. This means that saved error information is lost.

The F-system automatically reintegrates the F-I/O, as described above.

An operating error or an internal error can also trigger a startup of the safety program with the values from the load memory. If your process does not allow such a startup, you must program a restart/startup protection in the safety program: The output of process data must be blocked until manually enabled. This enable must not occur until it is safe to output process data and faults have been corrected. *(S008)*

## 6.5.2 After communication errors

### Behavior after communication errors

| Fail-safe value output after communication errors | F-I/O with "RIOforFA-Safety" profile with S7-1200/1500 F-CPU | F-I/O without profile "RIOforFA-Safety" with S7-1500 F-CPU | Every F-I/O with S7-300/400 F-CPU |
|---|---|---|---|
| If a communication error between the F-CPU and the F-I/O is detected, passivation of the entire F-I/O occurs. | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0* | | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>QBAD_I_xx and QBAD_O_xx = 1* |

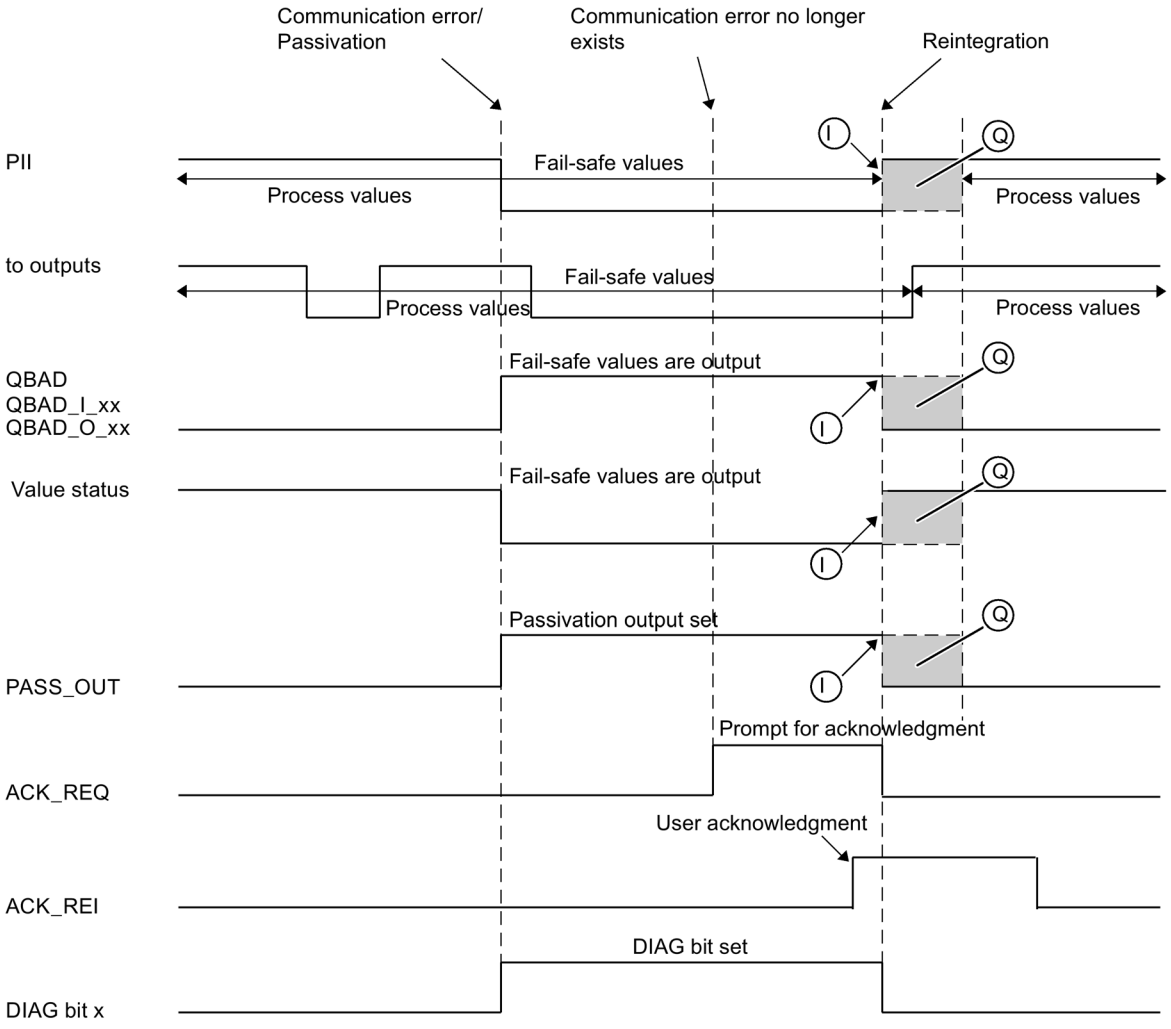
* Value status or QBAD_I_xx and QBAD_O_xx are not available for fail-safe GSD based DP slaves and fail-safe GSD based I/O devices without the "RIOforFA-Safety" profile.

### Reintegration of F-I/O

**Reintegration** of the relevant F-I/O, that is, provision of process data in the PII or transfer of process data provided in the PIQ to the fail-safe outputs, takes place only when the following occurs:

- All communication errors have been eliminated and the F-system has set tag ACK_REQ = 1
- A **user acknowledgment** with a positive edge has occurred:
  - At the ACK_REI tag of the F-I/O DB (Page 132) or
  - At the ACK_REI_GLOB input of the "ACK_GL" instruction (ACK_GL: Global acknowledgment of all F-I/O in an F-runtime group (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 407))

## Signal sequence for passivation and reintegration of F-I/O after communication errors




Ⓘ for F-I/O with inputs

Ⓠ For F-I/O with outputs and F-I/O with inputs and outputs
(signal trend depends on the F-I/O used and on the S7-1200/1500 or S7-300/400 F-CPU used)

## See also

Implementing User Acknowledgment in the Safety Program of the F-CPU of a DP Master or IO controller (Page 151)

Implementing user acknowledgment in the safety program of the F-CPU of a I-slave or I-device (S7-300, S7-400, S7-1500) (Page 156)

## 6.5.3 After F-I/O or channel faults

### Behavior after F-I/O faults

| Fail-safe value output after F-I/O faults | F-I/O with "RIOforFA-Safety" profile with S7-1200/1500 F-CPU | F-I/O without profile "RIOforFA-Safety" with S7-1500 F-CPU | Every F-I/O with S7-300/400 F-CPU |
|---|---|---|---|
| If an F-I/O fault is detected by the F-system, passivation of the entire F-I/O occurs. | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0* | | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>QBAD_I_xx and QBAD_O_xx = 1* |

* Value status or QBAD_I_xx and QBAD_O_xx are not available for fail-safe GSD based DP slaves and fail-safe GSD based I/O devices without the "RIOforFA-Safety" profile.

### Behavior after channel fault

| Fail-safe value output after channel faults | F-I/O with "RIOforFA-Safety" profile with S7-1200/1500 F-CPU | F-I/O without profile "RIOforFA-Safety" with S7-1500 F-CPU | Every F-I/O with S7-300/400 F-CPU |
|---|---|---|---|
| When passivation for **complete** F-I/O is configured:<br>If a channel fault is detected by the F-system, passivation of the entire F-I/O occurs. | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0* | | QBAD and PASS_OUT = 1<br>For **all** channels:<br>Channel value = fail-safe value (0)<br>QBAD_I_xx and QBAD_O_xx = 1* |
| With configuration of **channel-granular** passivation:<br>If a channel fault is detected by the F-system, passivation of the affected channels occurs. | QBAD and PASS_OUT **unchanged**<br>For the **affected** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0 | QBAD and PASS_OUT = 1<br>For the **affected** channels:<br>Channel value = fail-safe value (0)<br>Value status = 0* | QBAD and PASS_OUT = 1<br>For the **affected** channels:<br>Channel value = fail-safe value (0)<br>QBAD_I_xx and QBAD_O_xx = 1* |

* Value status or QBAD_I_xx and QBAD_O_xx are not available for fail-safe GSD based DP slaves and fail-safe GSD based I/O devices without the "RIOforFA-Safety" profile.

## Reintegration of F-I/O

**Reintegration** of the relevant F-I/O or the relevant channels of the F-I/O, that is, provision of process data in the PII or transfer of process data provided in the PIQ to the fail-safe outputs, takes place only when the following occurs:

- All F-I/O or channel faults have been eliminated.

If you have configured channel-granular passivation for the F-I/O, the relevant channels are reintegrated once the fault is corrected; any faulty channels remain passivated.

Reintegration is performed depending on your setting for the ACK_NEC tag or the "Channel failure acknowledge" parameter (configuration of the ET 200MP F-module and S7-1200 F-module)

- With ACK_NEC = 0 or the configuration "Channel failure acknowledge = automatic", **automatic reintegration** is performed as soon as the F-system detects that the fault has been corrected. For F-I/O with inputs, reintegration takes place right away. For F-I/O with outputs or F-I/O with inputs and outputs, depending on the F-I/O you are using, reintegration can take several minutes, first after the necessary test signals have been applied, which are used by the F-I/O to determine that the fault has been eliminated.
- With ACK_NEC = 1 or the configuration "Channel failure acknowledge = manual", reintegration is performed only as a result of user acknowledgment with a positive edge at the ACK_REI tag of the F-I/O DB or at the ACK_REI_GLOB input of the "ACK_GL" instruction. Acknowledgment can be made as soon as the F-system detects that the fault has been eliminated and tag ACK_REQ = 1 has been set.

  For fail-safe GSD based I/O devices with "RIOforFA-Safety" profile, consult the respective documentation for the fail-safe GSD based I/O device .

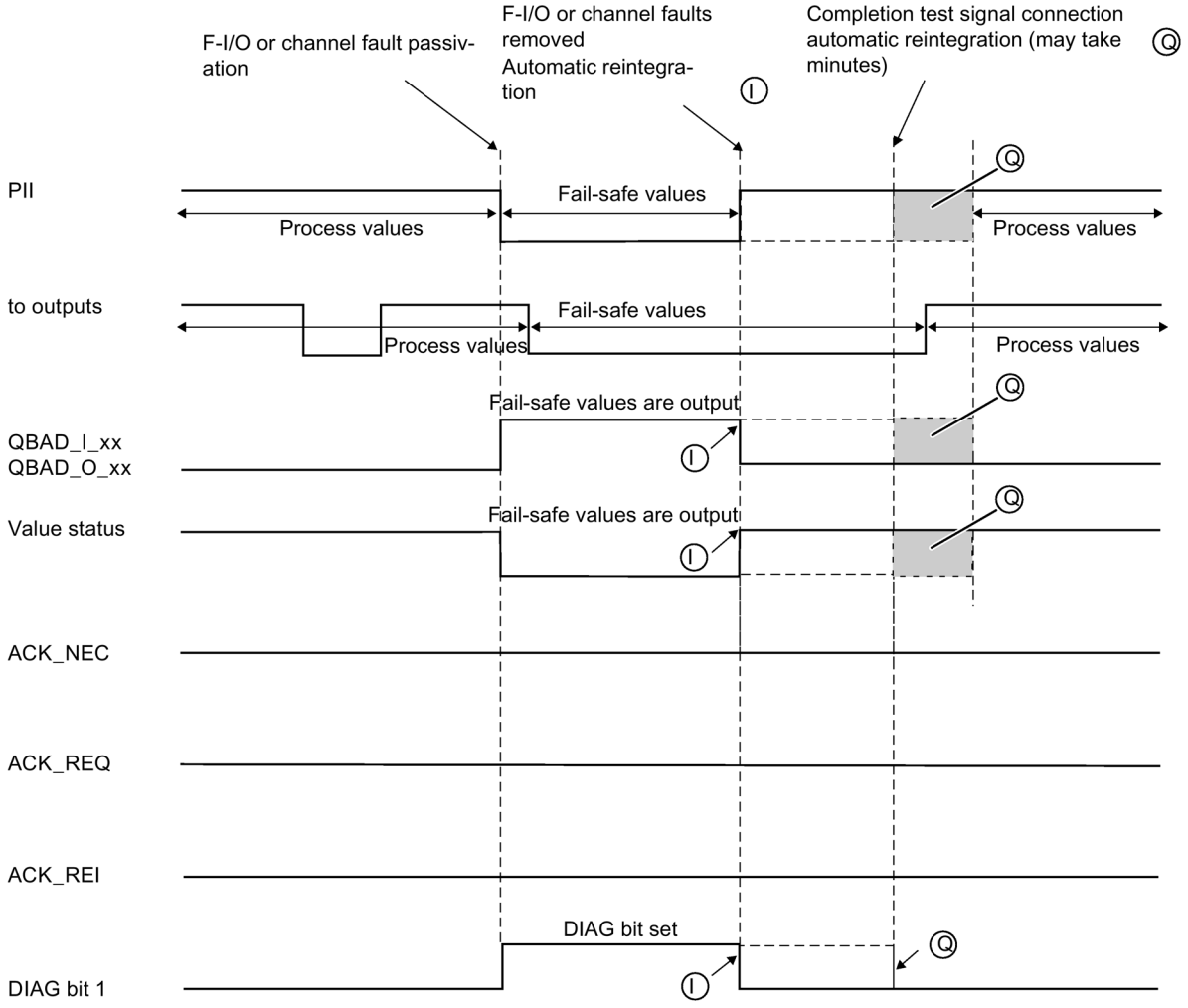
**⚠ WARNING**

Following a power failure of the F-I/O lasting less than the assigned F-monitoring time for the F-I/O, automatic reintegration can occur regardless of your setting for the ACK_NEC tag, as described for the case when ACK_NEC = 0 or configuration "Channel failure acknowledge = automatic".

If automatic reintegration is not permitted for the relevant process in this case, you must program startup protection by evaluating tags QBAD or QBAD_I_xx and QBAD_O_xx or value status or PASS_OUT.

In the event of a power failure of the F-I/O lasting longer than the specified F-monitoring time for the F-I/O, the F-system detects a communication error. *(S012)*

## Signal sequence for passivation and reintegration of F-I/O after F-I/O faults and channel faults when ACK_NEC = 0 or configuration "Channel failure acknowledge = automatic" (for passivation of entire F-I/O after channel faults)




Ⓘ for F-I/O with inputs

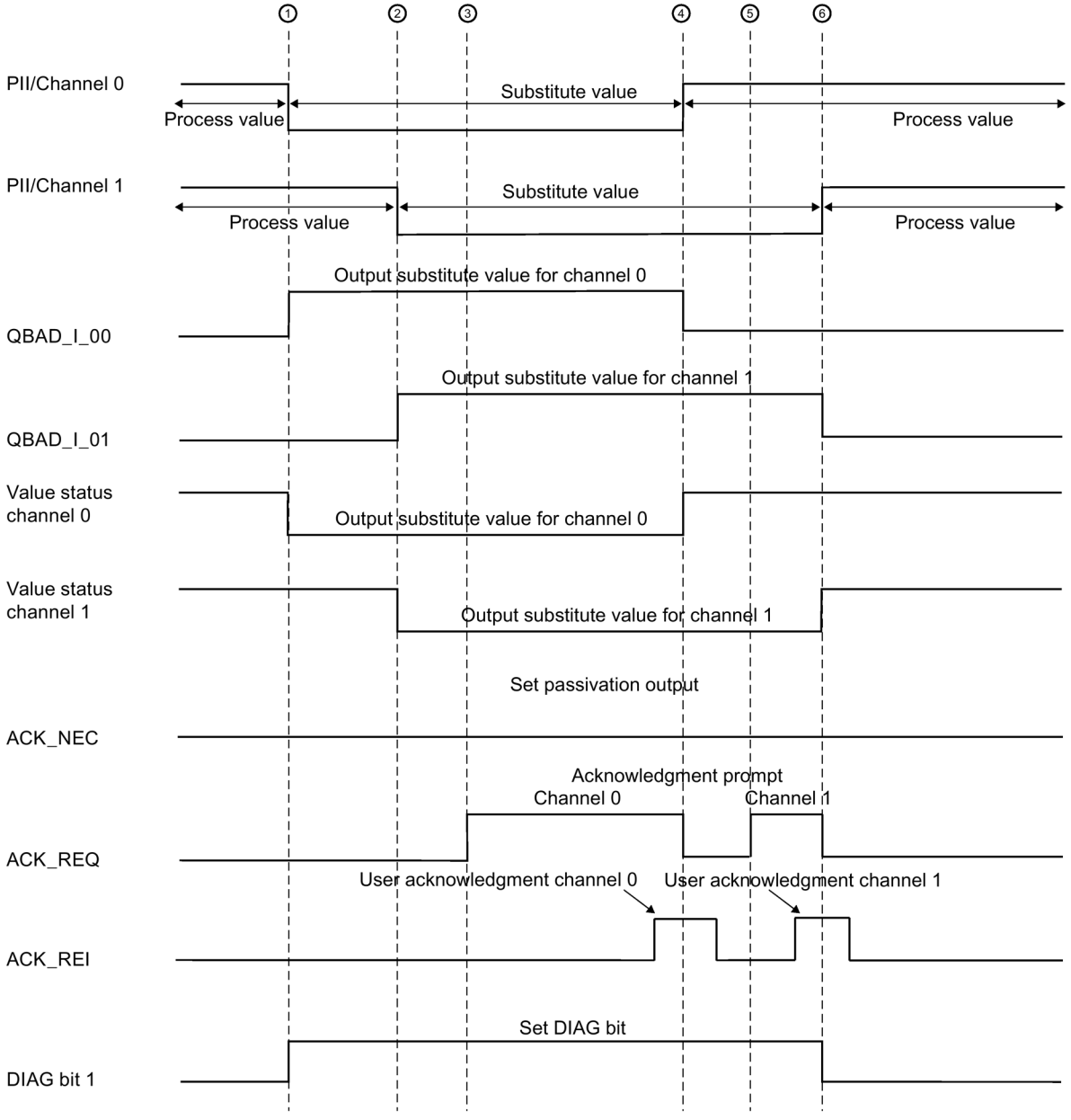
Ⓠ For F-I/O with outputs and F-I/O with inputs and outputs
(signal trend depends on the F-I/O used and on the S7-1200/1500 or S7-300/400 F-CPU used)

### Signal sequence for passivation and reintegration of F-I/O after F-I/O faults and channel faults when ACK_NEC = 1 or configuration "Channel failure acknowledge = manual" (for passivation of entire F-I/O after channel faults)

For the signal sequence for passivation and reintegration of F-I/O after F-I/O or channel faults when ACK_NEC = 1 or configuration "Channel failure acknowledge = manual" (initial value), see Passivation and reintegration of F-I/O (Page 139).

## Signal sequence for passivation and reintegration of F-I/O after channel faults when ACK_NEC = 1 or configuration "Channel failure acknowledge = manual" (for channel-granular passivation)

Example for an F-I/O with inputs:




① Channel fault for channel 0/passivation of channel 0
② Channel fault for channel 1/passivation of channel 1
③ Channel fault eliminated for channel 0
④ Reintegration of channel 0
⑤ Channel fault eliminated for channel 1
⑥ Reintegration of channel 1

## 6.5.4 Group passivation

### Programming a group passivation

If you want to enable passivation of additional F-I/O when an F-I/O or a channel of an F-I/O is passivated by the F-system, you can use the PASS_OUT/PASS_ON tags to perform **group passivation** of the associated F-I/O.

Group passivation by means of PASS_OUT/PASS_ON can, for example, be used to force simultaneous reintegration of all F-I/O after startup of the F-system.

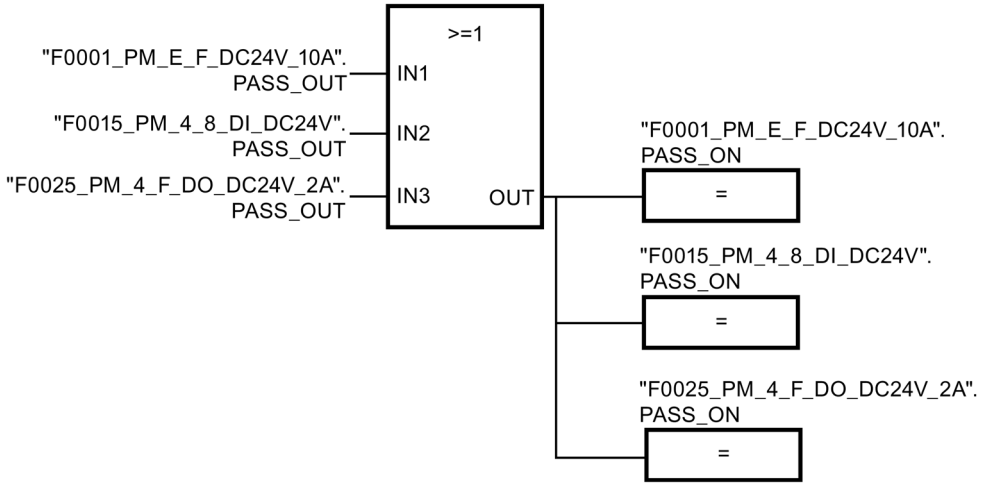
For group passivation, you must OR all PASS_OUT tags of the F-I/O in the group and assign the result to all PASS_ON tags of the F-I/O in the group.

During use of fail-safe values (0) due to group passivation by means of PASS_ON = 1, the QBAD tag of the F-I/O of this group = 1.

**Note**

Note the different behavior of PASS_OUT for F-I/O with/without RIOforFA-Safety profile (see table in chapter PASS_OUT/QBAD/QBAD_I_xx/QBAD_O_xx and value status (Page 135)).

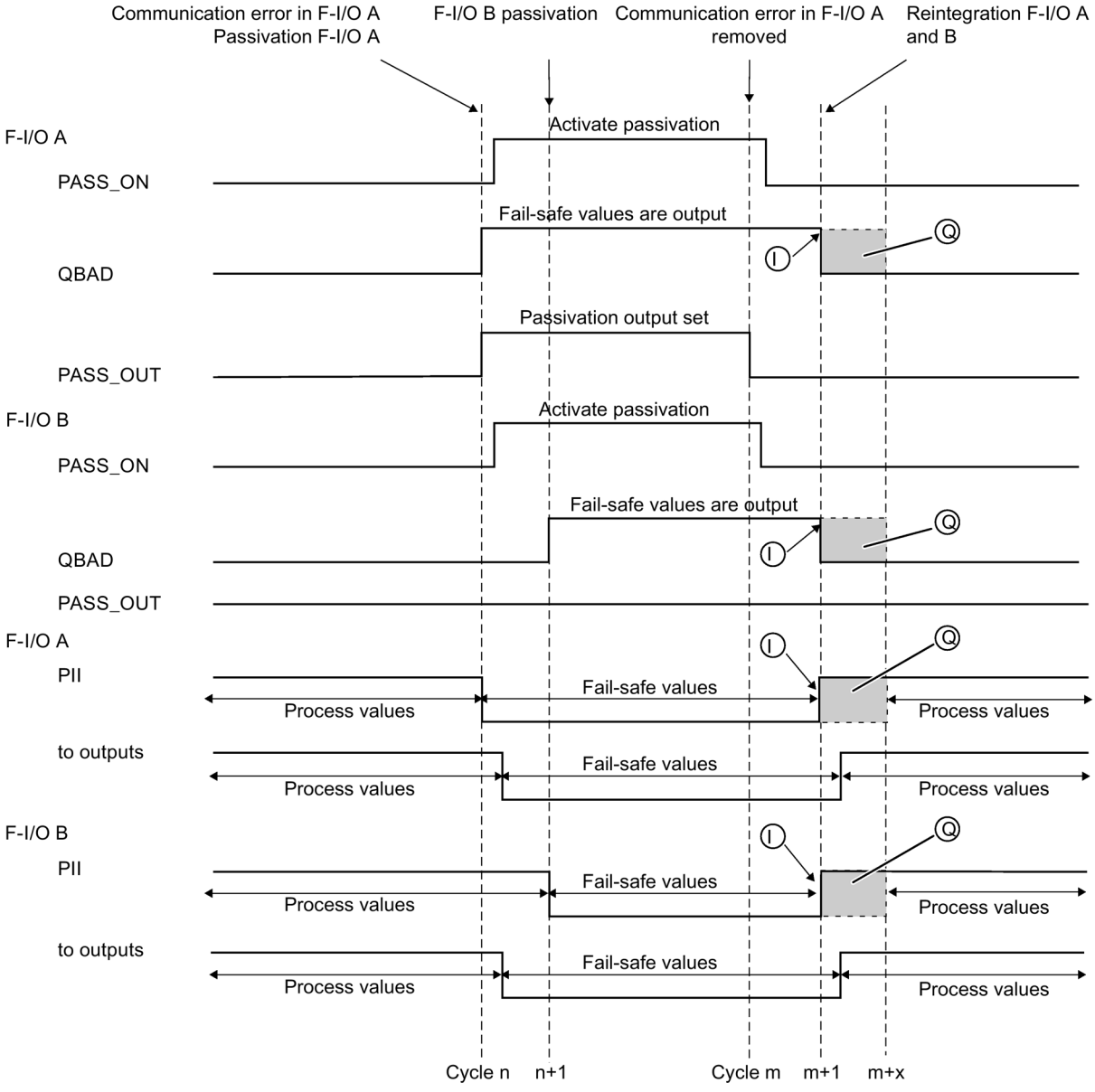
### Example of group passivation



## Reintegration of F-I/O

**Reintegration** of F-I/O passivated by group passivation occurs **automatically**, if a reintegration **automatic** or **through user acknowledgment**) occurs for the F-I/O that triggered the group passivation (PASS_OUT = 0).

## Signal sequence for group passivation following communication error




(I) for F-I/O with inputs

(Q) For F-I/O with outputs and F-I/O with inputs and outputs
(signal trend depends on the F-I/O used and on the S7-1200/1500 or S7-300/400 F-CPU used)

# Implementation of user acknowledgment 7

## 7.1 Implementing User Acknowledgment in the Safety Program of the F-CPU of a DP Master or IO controller

### Options for user acknowledgment

You have the following options for implementing a user acknowledgment:

- An acknowledgment key that you connect to an F-I/O with inputs
- An HMI system

### User acknowledgment by means of acknowledgment key

**Note**

When you implement user acknowledgment by means of acknowledgment key, and a communication error, an F-I/O fault, or a channel fault occurs in the F-I/O to which the acknowledgment key is connected, then it will not be possible to acknowledge the reintegration of this F-I/O.

This "blocking" can only be removed by a STOP-to-RUN transition of the F-CPU.

Consequently, it is recommended that you also provide for an acknowledgment by means of an HMI system, in order to acknowledge reintegration of an F-I/O to which an acknowledgment key is connected.

### User acknowledgment by means of an HMI system

For implementation of a user acknowledgment by means of an HMI system, the ACK_OP: Fail-safe acknowledgment (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 471) instruction is required.

### Procedure for programming user acknowledgment by means of an HMI system (S7-300, S7-400)

1. Select the "ACK_OP" instruction in the "Instructions" task card and place it in your safety program. The acknowledgment signal for evaluating user acknowledgments is provided at output OUT of ACK_OP.
2. On your HMI system, set up a field for manual entry of an "acknowledgment value" of "6" (1st step in acknowledgment) and an "acknowledgment value" of "9" (2nd step in acknowledgment).

   or

   Assign function key 1 to transfer an "acknowledgment value" of "6" (1st step in acknowledgment) once, and function key 2 to transfer an "acknowledgment value" of "9" (2nd step in acknowledgment) once. You need to assign the in/out IN (in the data area of the ACK_OP instruction) to the field or the function keys.
3. Optional: on your HMI system, evaluate output Q in the instance DB of F_ACK_OP to indicate the time frame within which the 2nd step in acknowledgment must occur or to indicate that the 1st step in acknowledgment has already occurred.

If you want to perform a user acknowledgment exclusively from a programming device or PC using the watch table (monitor/modify tag) without having to disable safety mode, then you must transfer an operand (memory word or DBW of a DB of the standard user program) at in/out IN when calling ACK_OP. You can then transfer "acknowledgment values" "6" and "9" on the programming device or PC by modifying the memory word or DBW of a DB once. The memory word or DBW of a DB must not be written by the program.

**Note**

If you connect the in/out IN to a memory word or DBW of a DB, you have use a separate memory word or DBW of a DB of the standard user program for each instance of the ACK_OP instruction at the in/out IN.

**⚠ WARNING**

The two acknowledgment steps must **not** be triggered by one single operation, for example by automatically storing them along with the time conditions in a program and using a function key to trigger them.
Having two separate acknowledgment steps also prevents erroneous triggering of an acknowledgment by your non-fail-safe HMI system. *(S013)*

**⚠ WARNING**

If you have HMI systems and F-CPUs that are interconnected and use the ACK_OP instruction for fail-safe acknowledgment, you need to ensure that the intended F-CPU will be addressed **before** you perform the two acknowledgment steps.

- To do this, store a network-wide* unique name for the F-CPU in a DB of your standard user program in each F-CPU.
- In your HMI system, set up a field from which you can read out the F-CPU name from the DB online before executing the two acknowledgment steps.
- Optional:

  in your HMI system, set up a field to permanently store the F-CPU name. Then, you can determine whether the intended F-CPU is being addressed by simply comparing the F-CPU name read out online with the permanently stored name.*(S014)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

### Procedure for programming user acknowledgment by means of an HMI system (S7-1200, S7-1500)

1. Select the "ACK_OP" instruction in the "Instructions" task card and place it in your safety program. The acknowledgment signal for evaluating user acknowledgments is provided at output OUT of ACK_OP.
2. Assign the ACK_ID input an identifier between 9 and 30000 for the acknowledgment.
3. Assign the in/out IN a memory word or DBW of a DB of the standard user program.

   **Note**

   You need to provide the in/out IN with a separate memory word or DBW of a DB of the standard user program supply for each instance of the ACK_OP instruction.

4. On your HMI system, set up a field for manual entry of an "acknowledgment value" of "6" (1st step in acknowledgment) and the "Identifier" configured at the ACK_ID input (2nd step in acknowledgment).

   or

   Allocate a function key 1 for a one-time transfer of the "acknowledgment value" of "6" (1st step in acknowledgment) and a function key 2 for a one-time transfer of the "Identifier" set at the ACK_ID input (2nd step in acknowledgment).
   You need to assign memory word or the DBW of the DB of the standard user program assigned to the in/out IN to the field or the function keys.
5. Optional: In your HMI system, evaluate output Q in the instance DB of F_ACK_OP to see the time frame within which the 2nd step in acknowledgment must occur or to indicate that the 1st step in acknowledgment has already occurred.

**⚠ WARNING**

The two acknowledgment steps must **not** be triggered by one single operation, for example by automatically storing them along with the time conditions in a program and using a function key to trigger them.
Having two separate acknowledgment steps also prevents erroneous triggering of an acknowledgment by your non-fail-safe HMI system. *(S013)*

**⚠ WARNING**

If you have HMI systems and F-CPUs that are interconnected and use the ACK_OP instruction for fail-safe acknowledgment, you need to ensure that the intended F-CPU will be addressed **before** you perform the two acknowledgment steps.

Alternative 1:

- The value for each identifier of the acknowledgment (ACK_ID input; data type: INT) can be freely selected in the range from 9 to 30000, but must be unique network-wide* for all instances of the ACK_OP instruction.
  You must supply the ACK_ID input with constant values when calling the instruction.
  Direct read or write access in the associated instance DB is not permitted in the safety program!

Alternative 2:

- To do this, store a network-wide* unique name for the F-CPU in a DB of your standard user program in each F-CPU.
- In your HMI system, set up a field from which you can read out the F-CPU name from the DB online before executing the two acknowledgment steps.
- Optional:
  in your HMI system, set up a field to permanently store the F-CPU name. Then, you can determine whether the intended F-CPU is being addressed by simply comparing the F-CPU name read out online with the permanently stored name. *(S047)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

### Example of procedure for programming a user acknowledgment for reintegrating an F-I/O

1. Optional: set the ACK_NEC tag in the respective F-I/O DB (Page 132) to "0" if automatic reintegration (without user acknowledgment) is to take place after an F-I/O fault or a channel fault.

   **⚠ WARNING**

   Assignment of the ACK_NEC = 0 tag is only allowed if automatic reintegration is permitted for the relevant process from a safety standpoint. *(S010)*

2. Optional: Evaluate the QBAD or QBAD_I_xx/QBAD_O_xx (S7-300/400) tags or value status (S7-1200, S7-1500) or DIAG in the respective F-I/O DB to trigger an indicator light in the event of an error, and/or generate error messages on the HMI system in your standard user program by evaluating the above tags or the value status. These messages can be evaluated before performing the acknowledgment operation. Alternatively, you can evaluate the diagnostic buffer of the F-CPU.
3. Optional: Evaluate the ACK_REQ tag in the respective F-I/O DB, for example, in the standard user program or on the HMI system, to query or to indicate whether user acknowledgment is required.
4. Assign the input of the acknowledgment key or the OUT output of the ACK_OP instruction to the ACK_REI tag in the respective F-I/O DB or the ACK_REI_GLOB input of the ACK_GL instruction (see above).

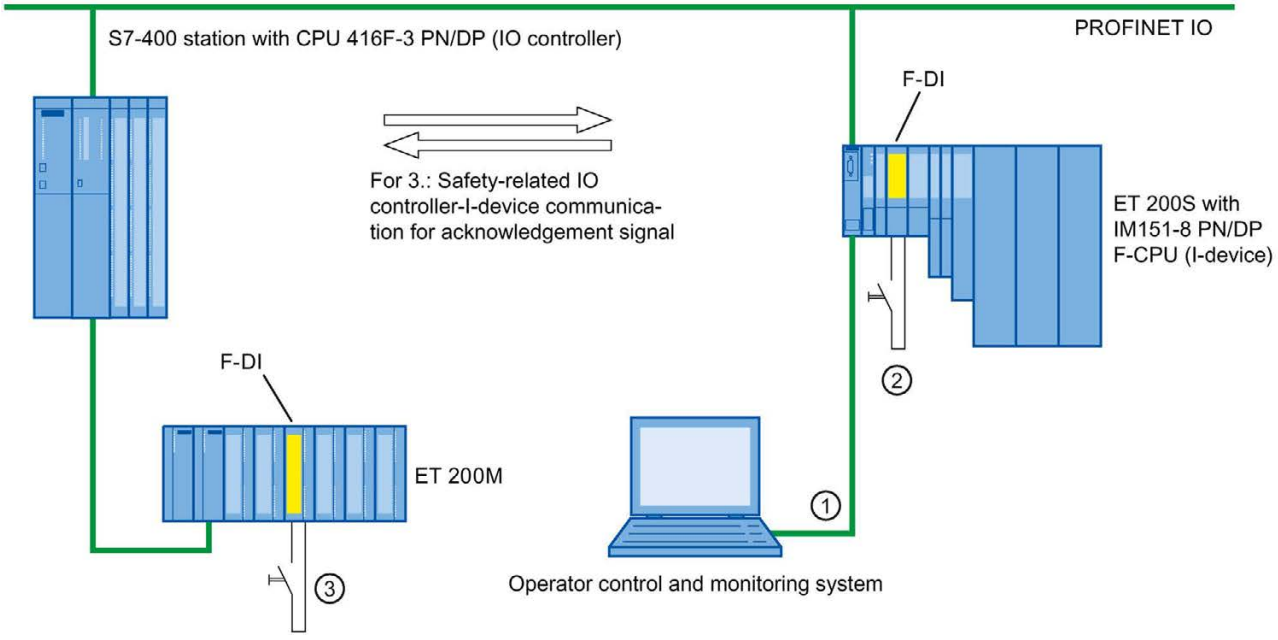
# 7.2 Implementing user acknowledgment in the safety program of the F-CPU of a I-slave or I-device (S7-300, S7-400, S7-1500)

## Options for user acknowledgment

You can implement a user acknowledgment by means of:

- An HMI system that you can use to access the F-CPU of the I-slave/I-Device
- An acknowledgment key that you connect to an F-I/O with inputs that is assigned to the F-CPU of the I-slave/I-Device
- An acknowledgment key that you connect to an F-I/O with inputs that is assigned to the F-CPU of the DP master/IO controller

These three options are illustrated in the figure below.



### 1. User acknowledgment by means of an HMI system that you can use to access the F-CPU of the I-slave/I-Device

The ACK_OP: Fail-safe acknowledgment (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 471) instruction is required to implement user acknowledgment with an HMI system that you can use to access the F-CPU of the I-slave/I-device.

**Programming procedure**

Follow the procedure described in "Implementing User Acknowledgment in the Safety Program of the F-CPU of a DP Master or IO controller (Page 151)" under "Programming procedure ...".

From your HMI system, you can then directly access the instance DB of ACK_OP in the I-slave/I-Device.

### 2. User acknowledgment by means of an acknowledgment key at an F-I/O with inputs that is assigned to the F-CPU of the I-slave/I-Device

**Note**

In the event of a communication error, F-I/O fault, or channel fault in the F-I/O to which the acknowledgment key is connected, an acknowledgment for reintegration of this F-I/O is no longer possible.

This "blocking" can only be removed by a STOP-to-RUN transition of the F-CPU of the I-slave/I-Device.

Consequently, it is recommended that you also provide for an acknowledgment by means of an HMI system that you can use to access the F-CPU of the I-slave/I-Device, in order to acknowledge reintegration of an F-I/O to which an acknowledgment key is connected (see 1).

### 3. User acknowledgment by means of acknowledgment key at an F-I/O with inputs that is assigned to the F-CPU of the DP master/IO controller

If you want to use the acknowledgment key that is assigned to the F-CPU at the DP master/IO controller to perform user acknowledgment in the safety program of the F-CPU of an I-slave/I-device as well, you must transmit the acknowledgment signal from the safety program in the F-CPU of the DP master/IO controller to the safety program in the F-CPU of the I-slave/I-device using safety-related master-I-slave/IO controller-I-device communication.

**Programming procedure**

1. Place the SENDDP (Page 480) instruction in the safety program in the F-CPU of the DP master/IO controller.
2. Place the RCVDP (Page 480) instruction in the safety program in the F-CPU of the I-slave/I-Device.
3. Supply an input SD_BO_xx of SENDDP with the input of the acknowledgment key.

4. The acknowledgment signal for evaluating user acknowledgments is now available at the corresponding RD_BO_xx output of RCVDP.

   The acknowledgment signal can now be read in the program sections in which further processing is to take place with fully qualified access directly in the associated instance DB (for example, "RCVDP_DB".RD_BO_02).

5. Supply the corresponding input SUBBO_xx of RCVDP with FALSE (fail-safe value 0) to ensure that user acknowledgment is not accidentally triggered before communication is established for the first time after startup of the sending and receiving F-systems, or in the event of a safety-related communication error.

**Note**

If a communication error, an F-I/O error, or a channel fault occurs at the F-I/O to which the acknowledgment key is connected, then an acknowledgment for reintegration of this F-I/O will no longer be possible.

This "blocking" can only be removed by a STOP-to-RUN transition of the F-CPU of the DP master/IO controller.

Consequently, it is recommended that you also provide for an acknowledgment by means of an HMI system that you can use to access the F-CPU of the DP master/IO controller, in order to acknowledge reintegration of the F-I/O to which an acknowledgment key is connected.

If a safety-related master-I-slave/IO controller-I-Device communication error occurs, the acknowledgment signal cannot be transmitted, and an acknowledgment for reintegration of safety-related communication is no longer possible.

This "blocking" can only be removed by a STOP-to-RUN transition of the F-CPU of the I-slave/I-Device.

Consequently, it is recommended that you also provide for an acknowledgment by means of an HMI system that you can use to access the F-CPU of the I-slave/I-Device, in order to acknowledge reintegration of the safety-related communication for transmission of the acknowledgment signal (see 1).

# Data exchange between standard user program and safety program

# 8

You have the option of transferring data between the safety program and the standard user program. Tags can be transferred using DBs, F-DBs and bit memory:

| | From the standard user program | | From the safety program | |
|---|---|---|---|---|
| | Read access | Write access | Read access | Write access |
| Tag from DB | Permitted | Permitted | A tag from the DB can be read-accessed *or* write-accessed | |
| Tag from F-DB | Permitted | **Not permitted** | Permitted | Permitted |
| Bit memory | Permitted | Permitted | Bit memory can be read-accessed *or* write-accessed | |

You can also access the process image of the standard I/O and F-I/O:

| | | From the standard user program | | From the safety program | |
|---|---|---|---|---|---|
| | | Read access | Write access | Read access | Write access |
| Process image of standard I/O | PII | Permitted | Permitted | Permitted | **Not permitted** |
| | PIQ | Permitted | Permitted | **Not permitted** | Permitted |
| Process image of F-I/O | PII | Permitted | **Not permitted** | Permitted | **Not permitted** |
| | PIQ | Permitted | **Not permitted** | **Not permitted** | Permitted |

## Decoupling of the safety program from the standard program in case of data transfer

For data exchange between standard user program and safety program, we recommend that you define special data blocks (transfer blocks) in which the data to be exchanged is stored. This action allows you to decouple the blocks of the standard and the safety program. The changes in the standard program do not affect the safety program (and vice versa) provided these data blocks are not modified.

# 8.1 Data Transfer from the Safety Program to the Standard User Program

## Data transfer from the safety program to the standard user program

The standard user program can read all data of the safety program, for example using symbolic (fully qualified) accesses to the following:

- The instance DBs of the F-FBs ("Name of instance DB".Signal_x)
- F-DBs (for example "Name of F_DB".Signal_1)
- The process image of the inputs and process image of the outputs of F-I/O (for example "Emergency_Stop_Button_1" (I 5.0))

**Note**

**For S7-300 and S7-400 F-CPUs**

The process image of the inputs of F-I/O is updated not only at the start of the main safety block, but also by the standard operating system.

To find the standard operating system update times, refer to the *Help on STEP 7* under "Process image of the inputs and output". For F-CPUs that support process image partitions, also bear in mind the update times when process image partitions are used. For this reason, when the process image of the inputs of F-I/O is accessed in the standard user program, it is possible to obtain different values than in the safety program. The differing values can occur due to:

- Different update times
- Use of fail-safe values in the safety program

To obtain the same values in the standard user program as in the safety program, you must not access the process image of the inputs in the standard program until after execution of an F-runtime group. In this case, you can also evaluate the QBAD or QBAD_I_xx tag in the associated F-I/O DB in the standard user program, in order to find out whether the process image of the inputs is receiving fail-safe values (0) or process data. When using process image partitions, also make sure that the process image is not updated by the standard operating system or by the UPDAT_PI instruction between execution of an F-runtime group and evaluation of the process image of the inputs in the standard user program.

**Note**

**For S7-1200/1500 F-CPUs**

The process image of the inputs of F-I/O is updated prior to processing the main safety block.

You can also write safety program data directly to the standard user program (see also the table of supported operand areas in: Restrictions in the programming languages FBD/LAD (Page 90)):

### Data block/bit memory

In order to write safety program data directly to the standard user program (e.g., DIAG output of the SENDDP instruction), you can write to data blocks of the standard user program from the safety program. However, a written tag must not be read in the safety program itself.

You can also write to bit memory in the safety program. However, written bit memory must not be read in the safety program itself.

### Process image of the outputs

You can write to the process image of the outputs (PIQ) of standard I/O in the safety program, for example for display purposes. The PIQ must not be read in the safety program.

## 8.2 Data Transfer from Standard User Program to Safety Program

### Data transfer from standard user program to safety program

As a basic principle, only fail-safe data or fail-safe signals from F-I/O and other safety programs (in other F-CPUs) can be processed in the safety program, as standard tags are unsafe.

If you have to process tags from the standard user program in the safety program, however, you can evaluate either bit memory from the standard user program, tags from a standard DB, or the process image input (PII) of standard I/O in the safety program (see table of supported operand areas in: Restrictions in the programming languages FBD/LAD (Page 90)).

**⚠ WARNING**

Because these tags are not generated safely, you must carry out additional process-specific validity checks in the safety program to ensure that no dangerous states can arise. If bit memory, a tag of a standard DB, or an input of standard I/O is used in both F-runtime groups, you must perform the validity check separately in each F-runtime group. *(S015)*

To facilitate checks, all PLC tags from the standard user program that are evaluated in the safety program are included in the safety summary (Page 290).

### Bit memory

In order to process tags of the standard user program in the safety program, you can also read bit memory in the safety program. However, read bit memory must not be written in the safety program itself.

### Data block

In order to process tags of the standard user program in the safety program, you can read tags from data blocks of the standard user program in the safety program. However, a read tag must not be written in the safety program itself.

### Process image of the inputs

You can read the process image of the inputs (PII) of standard I/O in the safety program. The PII must not be written in the safety program.

### Examples: Programming validity checks

- Use Comparison (Page 597) instructions to check whether tags from the standard user program exceed or fall below permitted high and low limits. You can then influence your safety function with the result of the comparison.
- Use the S: Set output (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 505), R: Reset output (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 504) or SR: Set/reset flip-flop (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 507) instructions, for example, with tags from the standard user program to allow a motor to be switched off, but not switched on.
- For switch-on sequences, use the AND logic operation instruction, for example, to logically combine tags from the standard user program with switch-on conditions that you derive from fail-safe tags.

If you want to process tags from the standard user program in the safety program, please bear in mind that there is not a sufficiently simple method of checking validity for all tags.

### Reading tags from the standard user program that can change during the runtime of an F-runtime group

If you want to read tags from the standard user program (bit memory, tags of a standard DB, or PII of standard I/O) in the safety program, and these tags can be changed - either by the standard user program or an operator control and monitoring system - during runtime of the F-runtime group in which they are read (for example because your standard user program is being processed by a higher-priority cyclic interrupt), you must use bit memory or tags of a standard DB for this purpose. You must write the bit memory or tags of a standard DB with the tags from the standard user program immediately before calling the F-runtime group. You are then permitted to access only this bit memory or these tags of a standard DB in the safety program.

Also note that **clock memory** that you defined when configuring your F-CPU in the "Properties" tab can change during runtime of the F-runtime group, since clock memory runs asynchronously to the F-CPU cycle.

**Note**

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

# Safety-related communication (S7-300, S7-400, S7-1500)

# 9

## 9.1 Configuring and programming communication (S7-300, S7-400)

### 9.1.1 Overview of communication

#### Introduction

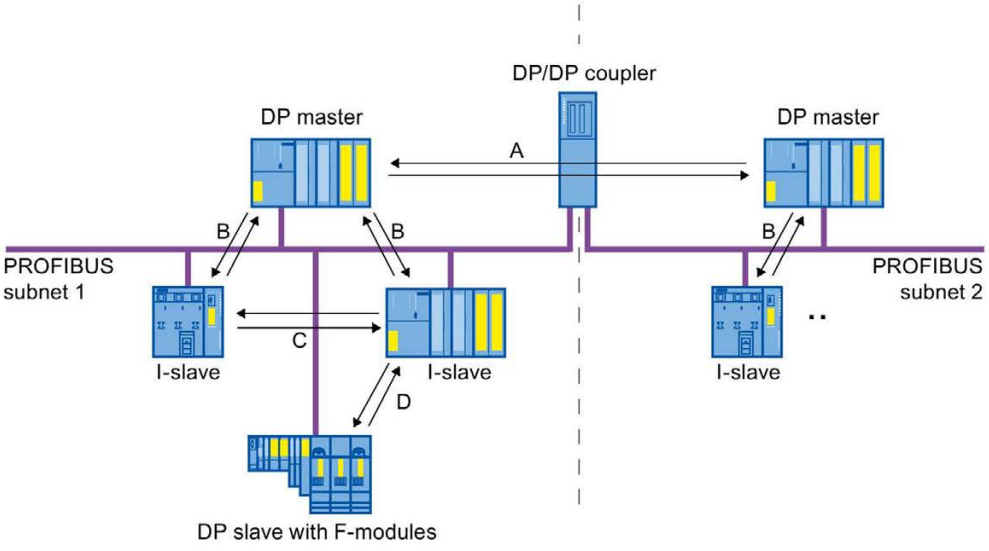
This section gives an overview of the safety-related communication options in SIMATIC Safety F-systems.

#### Options for safety-related communication

| Safety-related communication | On subnet | Additional hardware required |
|---|---|---|
| I-slave-slave communication | PROFIBUS DP | — |
| **Safety-related CPU-CPU communication:** | | |
| IO controller-IO controller communication | PROFINET IO | PN/PN coupler |
| Master-master communication | PROFIBUS DP | DP/DP coupler |
| IO controller-I-device communication | PROFINET IO | — |
| Master-I-slave communication | PROFIBUS DP | — |
| I-slave-I-slave communication | PROFIBUS DP | — |
| IO controller-I-slave communication | PROFINET IO and PROFIBUS DP | IE/PB link |
| Safety-related communication via S7 connections | Industrial Ethernet | — |
| IO controller-IO controller communication for S7 Distributed Safety | PROFINET IO | PN/PN coupler |
| Master-master communication for S7 Distributed Safety | PROFIBUS DP | DP/DP coupler |
| Safety-related communication to S7 Distributed Safety or S7 F Systems via S7 connections | Industrial Ethernet | — |

## Overview of safety-related communication via PROFIBUS DP

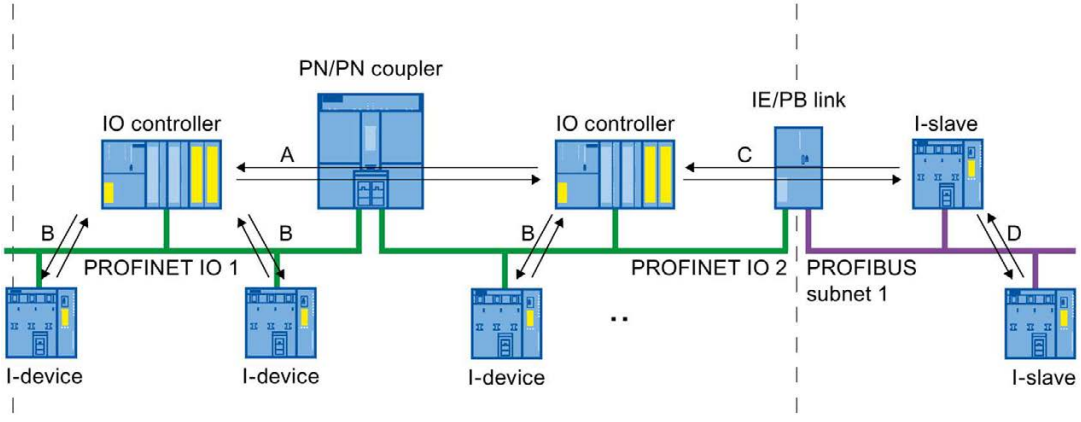
The figure below presents an overview of the 4 options for safety-related communication via PROFIBUS DP in SIMATIC Safety F-systems with S7-300/400 F-CPUs.




A Safety-related master-master communication
B Safety-related master-I-slave communication
C Safety-related I-slave-I-slave communication
D Safety-related I-slave-slave communication

## Overview of safety-related communication via PROFINET IO

The figure below presents an overview of the 4 options for safety-related communication via PROFINET IO in SIMATIC Safety F-systems with S7-300/400 F-CPUs. If an IE/PB link is used, safety-related communication is possible between assigned I-slaves.




A safety-related IO controller-IO controller communication
B safety-related IO controller-I-device communication
C safety-related IO controller-I-slave communication
D safety-related I-slave-I-slave communication integrating an IO controller

### Safety-related CPU-CPU communication via PROFIBUS DP or PROFINET IO

In safety-related CPU-CPU communication, a fixed amount of fail-safe data of the data type INT or BOOL is transmitted fail-safe between the safety programs in F-CPUs of DP masters/I-slaves or IO controllers/I-devices.

The data are transferred using the SENDDP instruction for sending and the RCVDP instruction for receiving. The data are stored in configured transfer areas of the devices. Each transfer area consists of one input and one output address area.

### Safety-related I-slave-slave communication via PROFIBUS DP

Safety-related I-slave-slave communication is possible with F-I/O in a DP slave that supports safety-related I-slave-slave communication, for example, with all ET 200S F-modules and with all S7-300 fail-safe signal modules with IM 153-2.

Safety-related communication between the safety program of the F-CPU of an I-slave and F-I/O of a DP slave takes place using direct data exchange, as in the standard program. The process image is used to access the channels of the F-I/O in the safety program of the F-CPU of the I-slave.

### Safety-related CPU-CPU communication via Industrial Ethernet

Safety-related CPU-CPU communication via Industrial Ethernet is possible using S7 connections, both from and to the following:

- S7-300 F-CPUs via the integrated PROFINET interface
- S7-400 F-CPUs via the integrated PROFINET interface or a CP 443-1 Advanced-IT

In safety-related communication via S7 connections, a specified amount of fail-safe data of data type BOOL, INT, WORD, DINT, DWORD, or TIME is transferred fail-safe between the safety programs of the F-CPUs linked by the S7 connection.

The data transfer makes use of the SENDS7 instruction for sending and the RCVS7 instruction for receiving. Data are exchanged using one F-DB ("F-communication DB") each at the sender and receiver ends.

### Safety-related CPU-CPU communication to *S7 Distributed Safety* or *F-systems*

Safety-related communication is possible from F-CPUs in *SIMATIC Safety* to F-CPUs in *S7 Distributed Safety* or *S7 F-systems*.

## 9.1.2 Safety-related IO controller-IO controller communication

### 9.1.2.1 Configure safety-related IO controller-IO controller communication

#### Introduction

Safety-related communication between safety programs of the F-CPUs of IO controllers takes place over a PN/PN coupler that you set up between the F-CPUs.

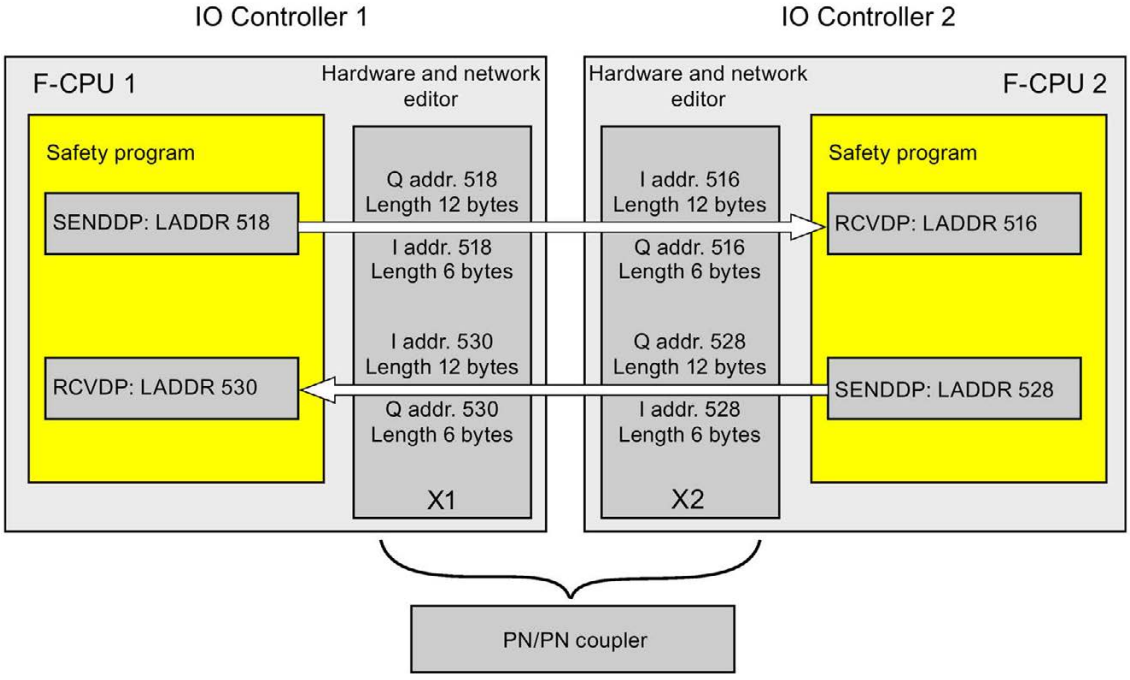
In the case of a CPU 416F-2 DP without an integrated PROFINET interface, you use a 443-1 Advanced-IT.

**Note**

Deactivate the "Data validity display DIA" parameter in the properties for the PN/PN coupler in the *hardware and network editor*. This is the default setting. Otherwise, safety-related IO controller-IO controller communication is not possible.

#### Configuring transfer areas

You must configure one transfer area for output data and one transfer area for input data in the *hardware and network editor* for each safety-related communication connection between two F-CPUs in the PN/PN coupler. The figure below shows how both of the F-CPUs are able to send **and** receive data (bidirectional communication). One transfer area for output data and one transfer area for input data must be configured in the PN/PN coupler for each of the two communication connections.



## Rules for defining transfer areas

The transfer area for output data and the transfer area for input data for the **data to be sent** must begin with the same start address. A total of 12 bytes (consistent) is required for the transfer area for output data; 6 bytes (consistent) are required for the transfer area for input data.

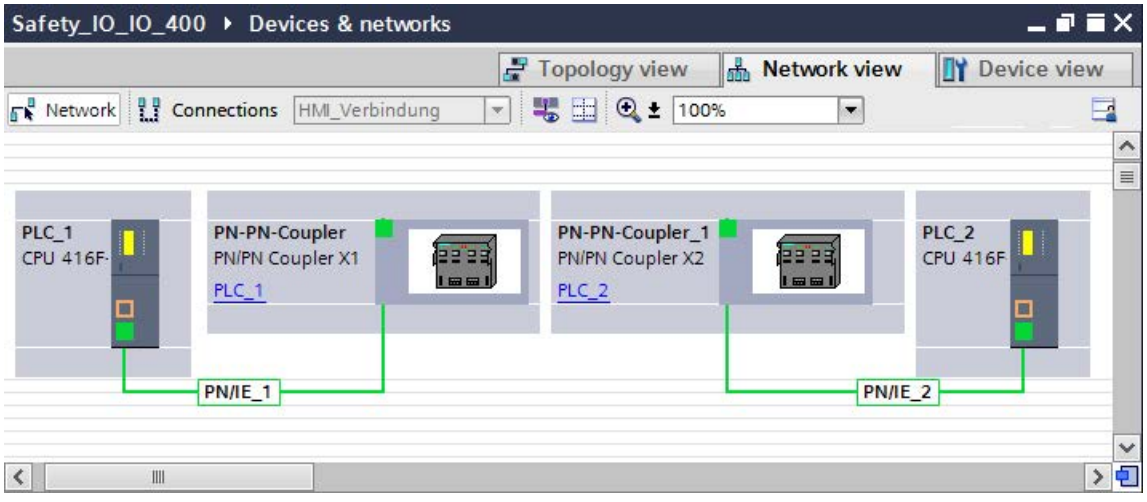
The transfer area for input data and the transfer area for output data for the **data to be received** must begin with the same start address. A total of 12 bytes (consistent) is required for the transfer area for input data; 6 bytes (consistent) are required for the transfer area for output data.

## Procedure for configuration

The procedure for configuring safety-related IO controller-IO controller communication is identical to that in the standard system.

Proceed as follows:

1. Insert two F-CPUS from the "Hardware catalog" task card into the project.
2. Switch to the network view of the *hardware and network editor*.
3. Insert a PN/PN coupler X1 and a PN/PN coupler X2 from "Other field devices\PROFINET IO\Gateway\Siemens AG\PN/PN Coupler" in the "Hardware catalog" task card.
4. Connect the PN interface of the F-CPU 1 with the PN interface of the PN/PN Coupler X1 and the PN interface of the F-CPU 2 with the PN interface of PN/PN Coupler X2.




5. Switch to the device view of PN/PN coupler X1 for bidirectional communication connections i.e. where each F-CPU is both to send and to receive data. Select the following modules from "IN/OUT" in the "Hardware catalog" task card (with filter activated), and insert them in the "Device overview" tab:
   - One "IN/OUT 6 bytes / 12 bytes" module and
   - One "IN/OUT 12 bytes / 6 bytes" module

6. In the properties of the modules, assign the addresses outside the process image as follows:

   For the "IN/OUT 6 bytes / 12 bytes" module for sending data for example:

   – Input addresses: Start address 518

   – Output addresses: Start address 518

   For the "IN/OUT 12 bytes / 6 bytes" module for receiving data for example:

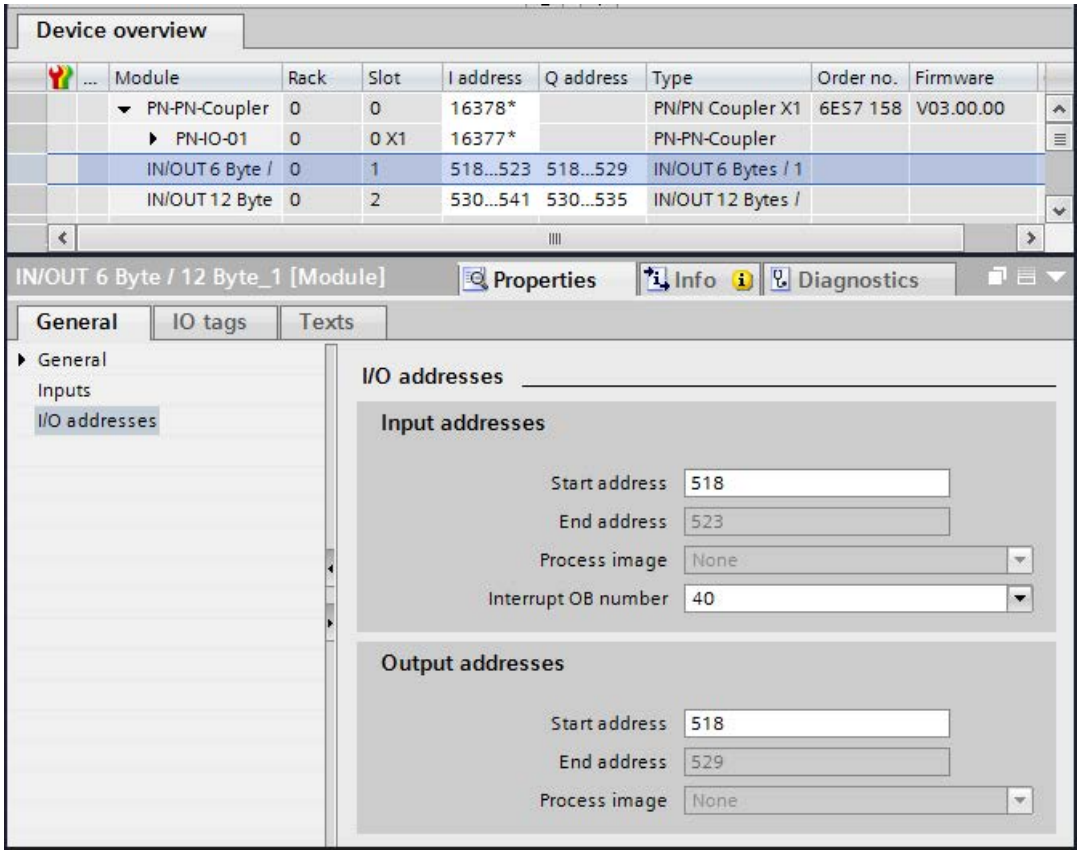
   – Input addresses: Start address 530

   – Output addresses: Start address 530

**Note**

Make sure that you assign identical start addresses for the address areas of the output and input data.

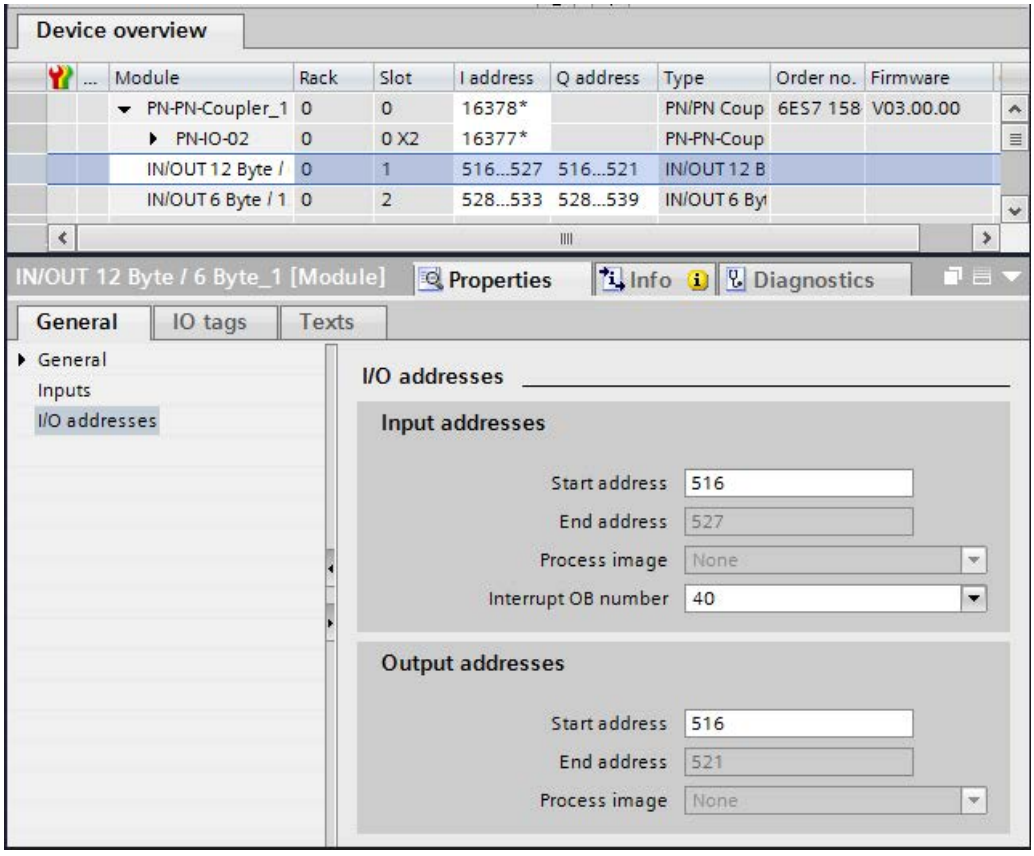
**Tip:** Make a note of the start addresses of the transfer areas. You need these to program the SENDDP and RCVDP blocks (LADDR input).



7. Select the following modules from "IN/OUT" in the device view of PN/PN coupler X2 and insert them in the "Device overview" tab:
   - One "IN/OUT 12 bytes / 6 bytes" module and
   - One "IN/OUT 6 bytes / 12 bytes" module
8. In the properties of the modules, assign the addresses outside the process image as follows:

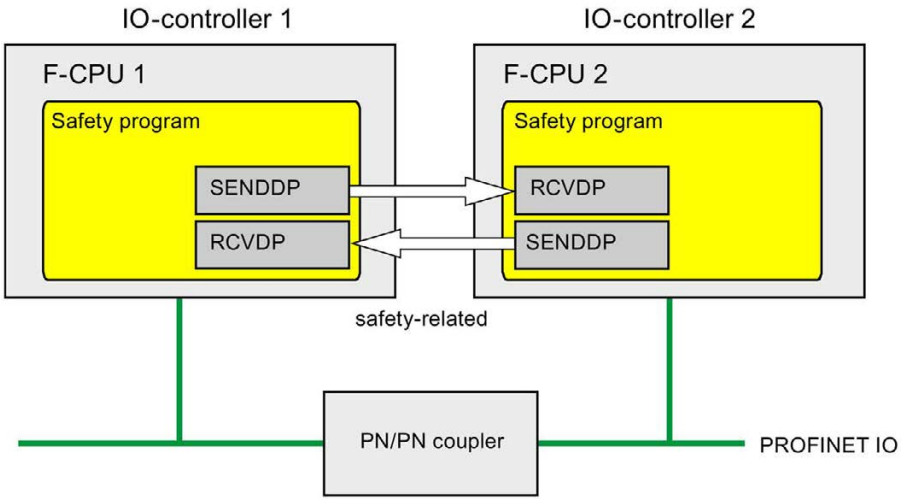
   For the "IN/OUT 12 bytes / 6 bytes" module for receiving data for example:
   - Input addresses: Start address 516
   - Output addresses: Start address 516

   For the "IN/OUT 6 bytes / 12 bytes" module for sending data for example:
   - Input addresses: Start address 528
   - Output addresses: Start address 528



### 9.1.2.2 Safety-related IO controller-IO controller communication via SENDDP and RCVDP

#### Communication via SENDDP and RCVDP instructions




Safety-related communication between the F-CPUs of the IO controllers uses the SENDDP and RCVDP instructions for sending and receiving respectively. These can be used to perform a fail-safe transfer of a *fixed* amount of fail-safe data of the data type INT or BOOL.

You can find these instructions in the "Instructions" task card under "Communication". The RCVDP instruction **must** be called at the start of the main safety block. The SENDDP instruction **must** be called at the end of the main safety block.

Note that the send signals are not sent until after the SENDDP instruction call at the end of execution of the relevant F-runtime group.

A detailed description of the SENDDP and RCVDP instructions can be found in SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500) (Page 646).

### 9.1.2.3 Program safety-related IO controller-IO controller communication

#### Requirement for programming

The transfer areas for input and output data for the PN/PN coupler must be configured.

#### Programming procedure

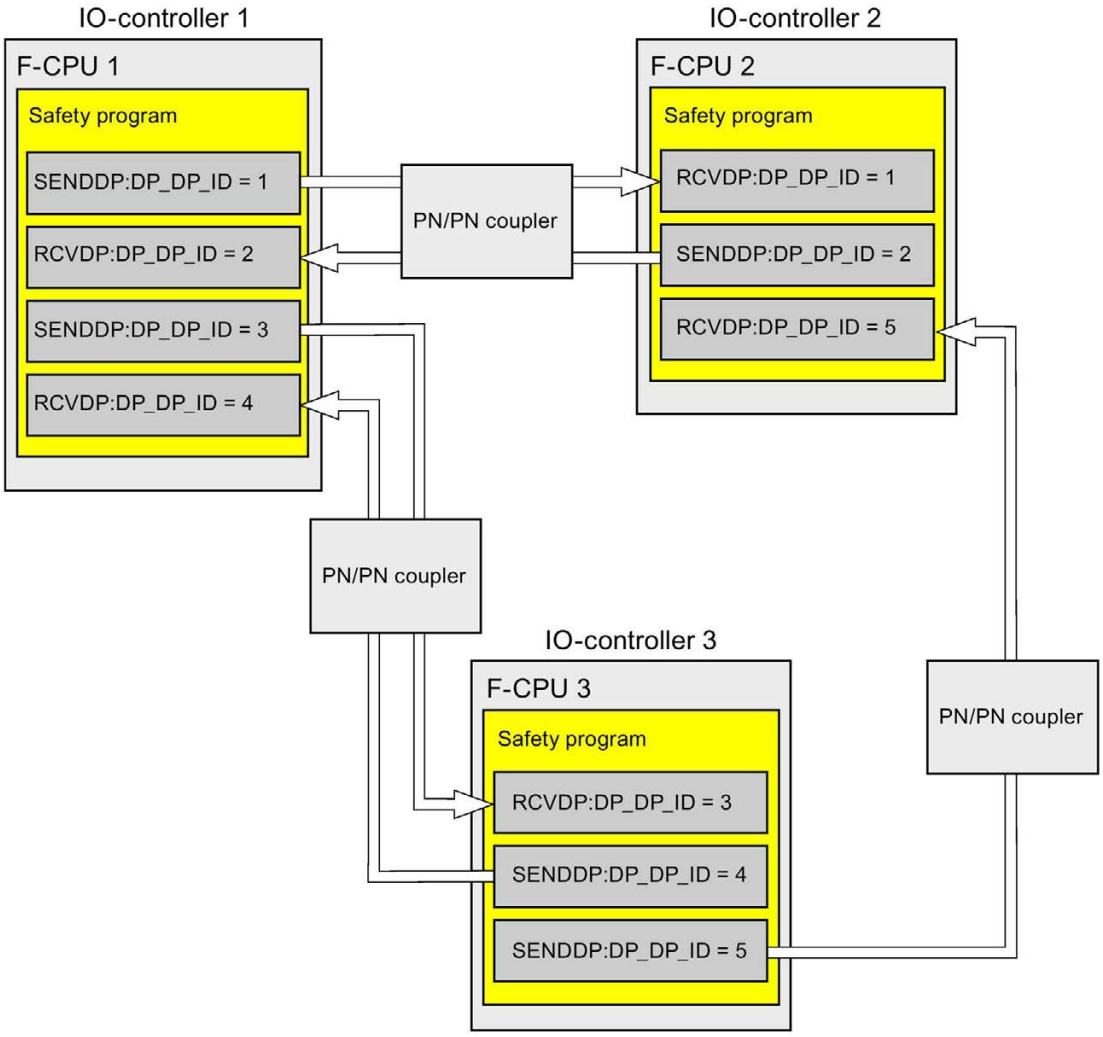
You program safety-related IO controller-IO controller communication as follows:

1. In the safety program from which data are to be sent, call the SENDDP instruction (Page 646) for sending at the end of the main safety block.
2. In the safety program in which data are to be received, call the RCVDP instruction (Page 646) for receiving at the start of the main safety block.
3. Assign the start addresses of the output and input data transfer areas of the PN/PN coupler configured in the *hardware and network editor* to the respective LADDR inputs.

   You must carry out this assignment for every communication connection for each of the F-CPUs involved.

4. Assign the value for the respective address relationship to the DP_DP_ID inputs. This establishes the communication relationship between the SENDDP instruction in one F-CPU and the RCVDP instruction in the other F-CPU: The associated instructions receive the same value for DP_DP_ID.

   The figure below contains an example of how to specify the address relationships for the inputs of the SENDDP and RCVDP instructions for five safety-related IO controller-IO-controller communication relationships.




**WARNING**

The value for each address relationship (DP_DP_ID input; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

5. Supply the SD_BO_xx and SD_I_xx inputs of SENDDP with the send signals. To cut down on intermediate signals when transferring parameters, you can write the value directly to the instance DB of SENDDP using fully qualified access (for example, "Name SENDDP_1".SD_BO_02) before calling SENDDP.
6. Supply the RD_BO_xx and RD_I_xx outputs of RCVDP with the signals that you want to process further in other program sections or use fully qualified access to read the received signals directly in the associated instance DB in the program sections to be processed further (e.g., "Name RCVDP_1".RD_BO_02).
7. Supply the SUBBO_xx and SUBI_xx inputs of RCVDP with the fail-safe values that are to be output by RCVDP in place of the process data until communication is established for the first time after startup of the sending and receiving F-systems or in the event of an error in safety-related communication.
   - Specification of constant fail-safe values:

     For data of data type INT, you can enter constant fail-safe values directly as constants in the SUBI_xx input (initial value = "0"). If you want to specify a constant fail-safe value "TRUE" for data of the data type BOOL, set TRUE for the SUBBO_xx input (initial value = "FALSE").
   - Specification of dynamic fail-safe values:

     If you want to specify dynamic fail-safe values, define a tag that you change dynamically through your safety program in an F-DB and specify this tag (fully qualified) in the SUBI_xx or SUBBO_xx input.

     **⚠ WARNING**

     Note that your safety program for dynamically changing the tag for a dynamic fail-safe value can only be processed after the call of the RCVDP, because prior to the RCVDP call, there must not be any network and no more than one other RCVDP instruction in the main safety block. You must therefore assign appropriate start values for these tags to be output by RCVDP in the first cycle after a startup of the F-system. *(S017)*
8. Configure the TIMEOUT inputs of the RCVDP and SENDDP instructions with the required monitoring time.

   **⚠ WARNING**

   It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal level is pending for at least as long as the assigned monitoring time. *(S018)*

   Information on calculating the monitoring times can be found in Monitoring and response times (Page 661).

9. Optional: Evaluate the ACK_REQ output of the RCVDP instruction, for example, in the standard user program or on the HMI system in order to query or to indicate whether user acknowledgment is required.
10. Supply the ACK_REI input of the RCVDP instruction with the acknowledgment signal for reintegration.
11. Optional: Evaluate the SUBS_ON output of the RCVDP or SENDDP instruction in order to query whether the RCVDP instruction is outputting the fail-safe values assigned in the SUBBO_xx and SUBI_xx inputs.
12. Optional: Evaluate the ERROR output of the RCVDP or SENDDP instruction, for example, in the standard user program or on the HMI system in order to query or to indicate whether a communication error has occurred.
13. Optional: Evaluate the SENDMODE output of the RCVDP instruction in order to query whether the F-CPU with the associated SENDDP instruction is in disabled safety mode (Page 293).

#### 9.1.2.4 Safety-related IO controller-IO controller communication - Limits for data transfer

**Note**

If the data quantities to be transmitted exceed the capacity of the SENDDP / RCVDP correlated instructions, a second (or third) SENDDP / RCVDP call can be used. This requires configuration of an additional connection via the PN/PN coupler. Whether or not this is possible with one single PN/PN coupler depends on the capacity restrictions of the PN/PN coupler.

## 9.1.3 Safety-related master-master communication

### 9.1.3.1 Configure safety-related master-master communication

#### Introduction

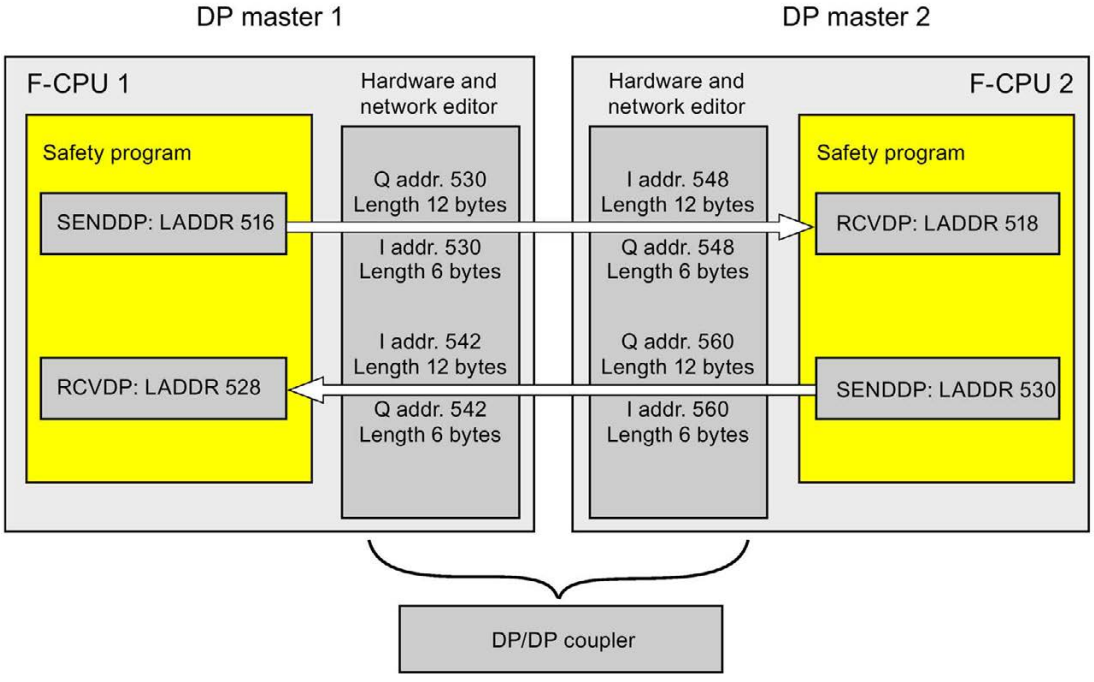
Safety-related communication between safety programs of the F-CPUs of DP masters takes place via a DP/DP coupler.

**Note**

Switch the data validity indicator "DIA" on the DIP switch of the DP/DP coupler to "OFF". Otherwise, safety-related CPU-CPU communication is not possible.

#### Configuring transfer areas

You must configure one transfer area for output data and one transfer area for input data in the *hardware and network editor* for each safety-related communication connection between two F-CPUs in the DP/DP coupler. The figure below shows how both of the F-CPUs are able to send and receive data (bidirectional communication). One transfer area for output data and one transfer area for input data must be configured in the DP/DP coupler for each of the two communication connections.



## Rules for defining transfer areas

The transfer area for input data and the transfer area for output data for the **data to be sent** must begin with the same start address. A total of 6 bytes (consistent) is required for the transfer area for input data; 12 bytes (consistent) are required for the transfer area for output data.

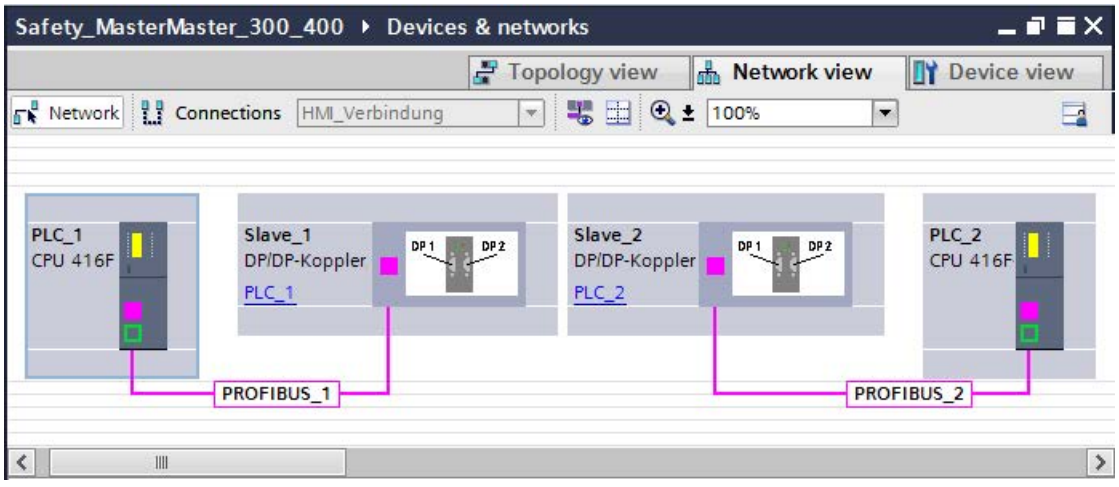
The transfer area for input data and the transfer area for output data for the **data to be received** must begin with the same start address. A total of 12 bytes (consistent) is required for the transfer area for input data; 6 bytes (consistent) are required for the transfer area for output data.

## Procedure for configuration

The procedure for configuring safety-related master-master communication is identical to that in the standard system.

Proceed as follows:

1. Insert two F-CPUS from the "Hardware catalog" task card into the project.
2. Switch to the network view of the *hardware and network editor*.
3. Select a DP/DP coupler from "Other field devices\PROFIBUS DP\Gateways\Siemens AG\DP/DP Coupler" in the "Hardware catalog" task card and insert it into the network view of the hardware and network editor.
4. Insert a second DP/DP coupler.
5. Connect a DP interface of F-CPU 1 to the DP interface of a DP/DP coupler and a DP interface of F-CPU 2 to the DP interface of the other DP/DP coupler.




6. A free PROFIBUS address is assigned automatically in the properties of the DP/DP-coupler in the device view. You must set this address on the DP/DP coupler of PLC 1, either using the DIP switch on the device or in the configuration of the DP/DP coupler (see DP/DP Coupler (http://support.automation.siemens.com/WW/view/en/1179382) manual).

7. Switch to the device view of the DP/DP coupler for PLC1 for bidirectional communication connections i.e. where each F-CPU is both to send and to receive data. Select the following modules from the "Hardware catalog" task card (with filter activated), and insert them in the "Device overview" tab:
   - One "6 bytes I/12 bytes Q consistent" module, and
   - One "12 bytes I/6 bytes Q consistent" module

8. In the properties of the modules, assign the addresses outside the process image as follows:

   For "6 bytes I/12 bytes Q consistent" module for sending data for example:

   – Input addresses: Start address 530

   – Output addresses: Start address 530

   For "12 bytes I/6 bytes Q consistent" module for receiving data for example:

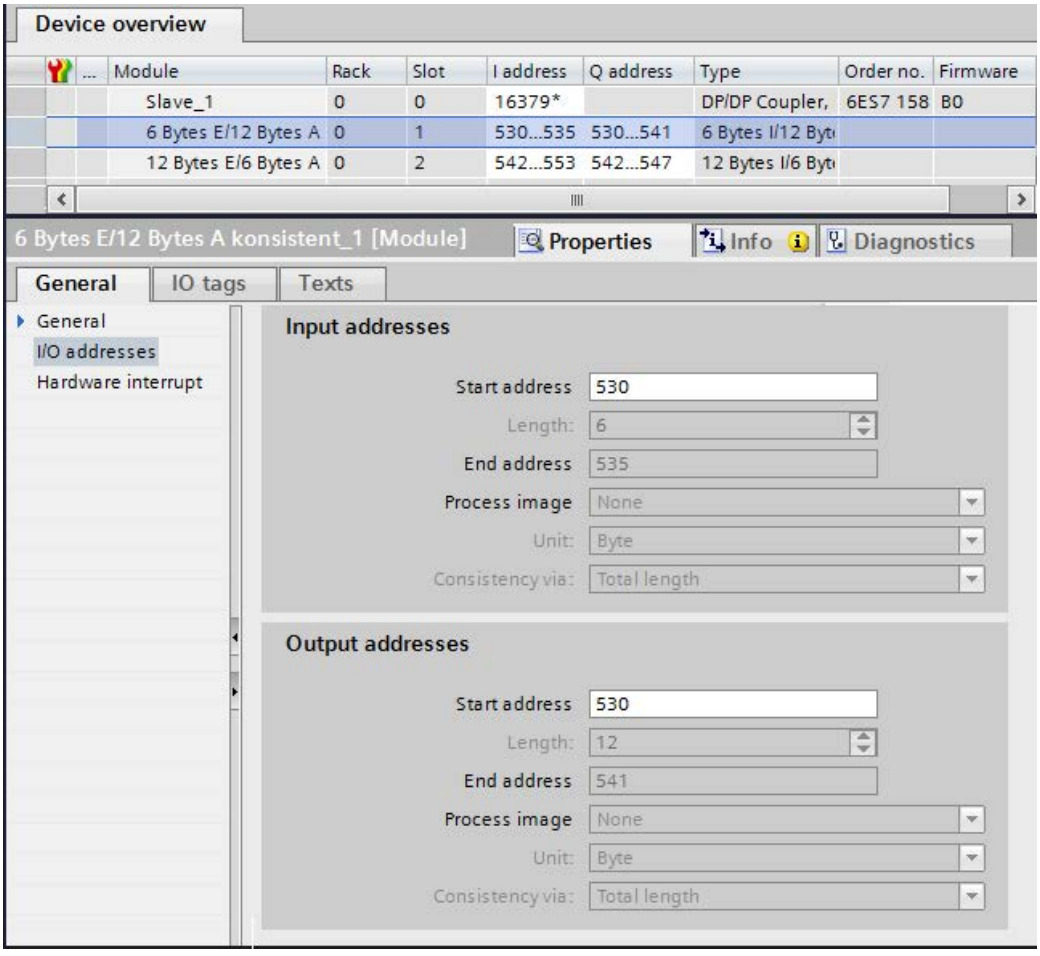
   – Input addresses: Start address 542

   – Output addresses: Start address 542

   **Note**

   Make sure that you assign identical start addresses for the address areas of the output and input data.

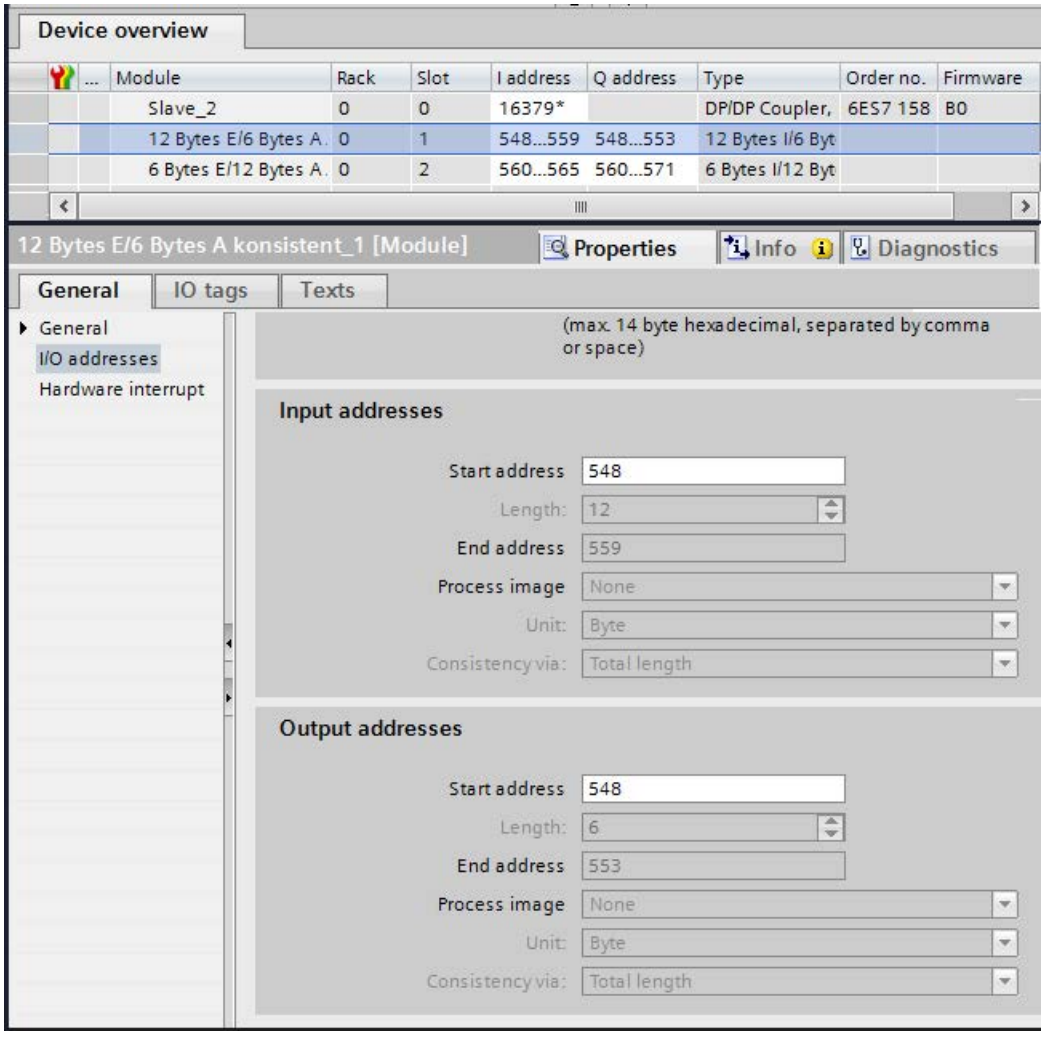
   **Tip:** Make a note of the start addresses of the transfer areas. You need these to program the SENDDP and RCVDP blocks (LADDR input).



9. Select the following modules from the "Hardware catalog" task card (with filter activated) in the device view of DP/DP coupler PLC2, and insert them in the "Device overview" tab:
   - One "12 bytes I/6 bytes Q consistent" module, and
   - One "6 bytes I/12 bytes Q consistent" module
10. In the properties of the modules, assign the addresses outside the process image as follows:

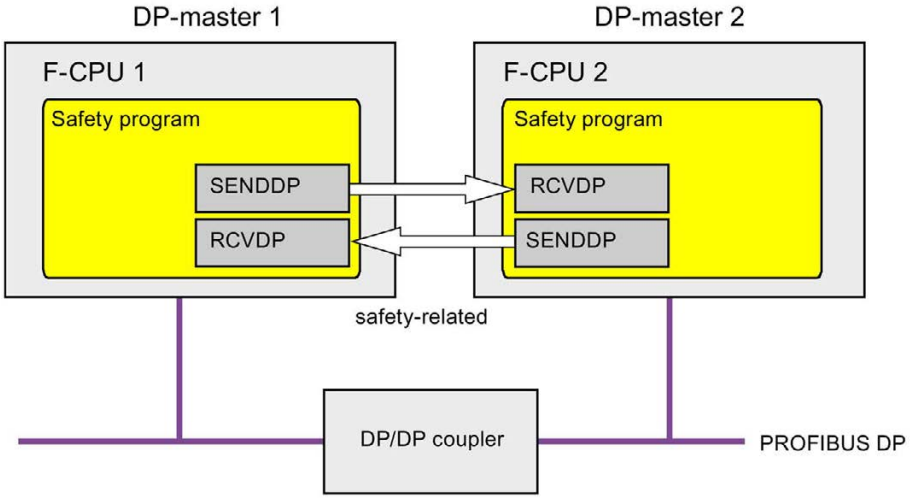
    For "12 bytes I/6 bytes Q consistent" module for receiving data for example:
    - Input addresses: Start address 548
    - Output addresses: Start address 548

    For "6 bytes I/12 bytes Q consistent" module for sending data for example:
    - Input addresses: Start address 560
    - Output addresses: Start address 560



### 9.1.3.2 Safety-related master-master communication via SENDDP and RCVDP

#### Communication via SENDDP and RCVDP instructions




Safety-related communication between the F-CPUs of the DP masters uses the SENDDP and RCVDP instructions for sending and receiving respectively. These can be used to perform a fail-safe transfer of a *fixed* amount of fail-safe data of the data type INT or BOOL.

You can find these instructions in the "Instructions" task card under "Communication". The RCVDP instruction **must** be called at the start of the main safety block. The SENDDP instruction **must** be called at the end of the main safety block.

Note that the send signals are not sent until after the SENDDP instruction call at the end of execution of the relevant F-runtime group.

A detailed description of the SENDDP and RCVDP instructions can be found in SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500) (Page 646).

### 9.1.3.3 Program safety-related master-master communication

#### Requirement for programming

The transfer areas for input and output data for the DP/DP coupler must be configured.

#### Programming procedure

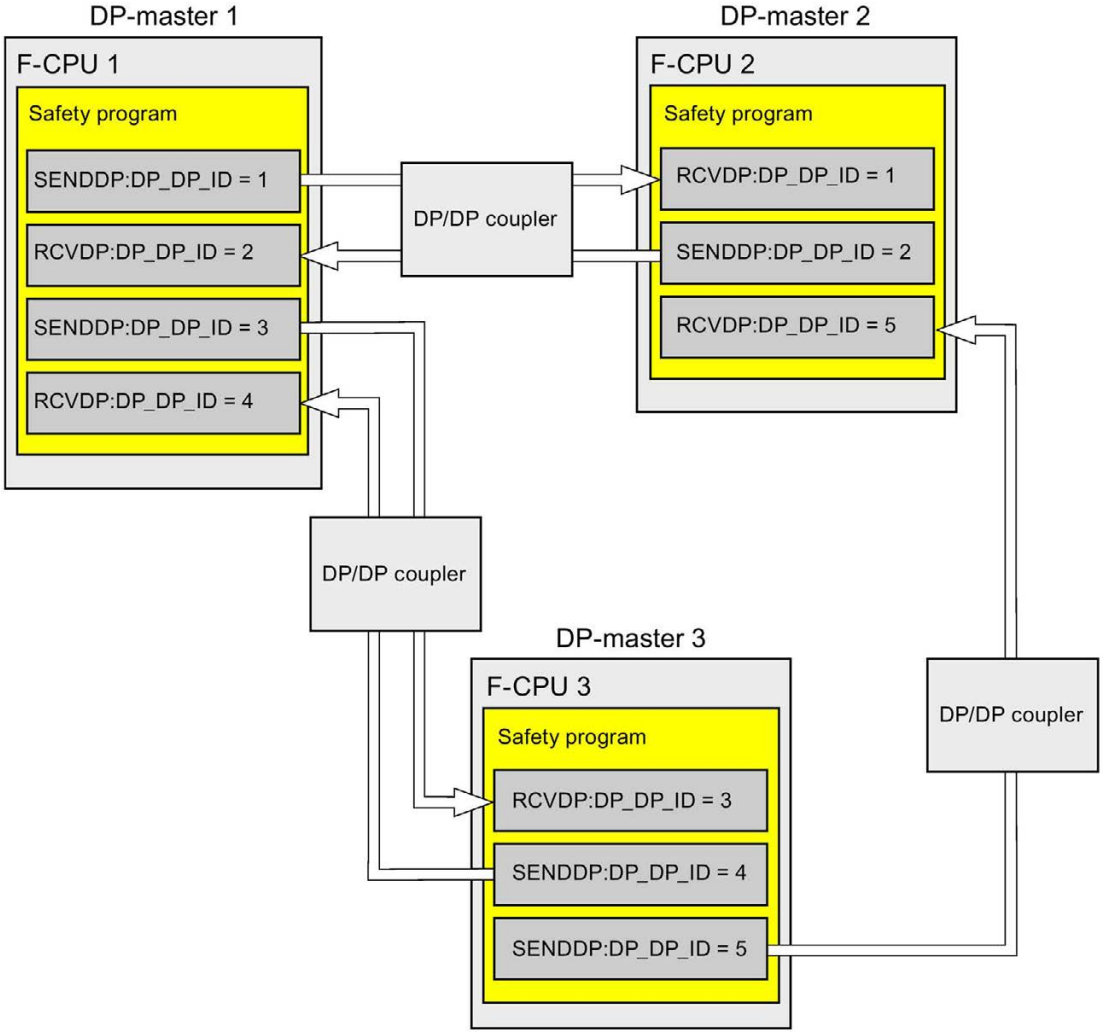
You program safety-related master-master communication as follows:

1. In the safety program from which data are to be sent, call the SENDDP instruction (Page 646) for sending at the end of the main safety block.
2. In the safety program from which data are to be received, call the RCVDP instruction (Page 646) for receiving at the start of the main safety block.
3. Assign the start addresses of the transfer areas for output and input data of the DP/DP coupler configured in the *hardware and network editor* to the respective LADDR inputs.

   You must carry out this assignment for every communication connection for each of the F-CPUs involved.

4. Assign the value for the respective address relationship to the DP_DP_ID inputs. This establishes the communication relationship between the SENDDP instruction in one F-CPU and the RCVDP instruction in the other F-CPU: The associated instructions receive the same value for DP_DP_ID.

   The figure below contains an example of how to specify the address relationships at the inputs of SENDDP and RCVDP instructions for five safety-related master-master communications relationships.




**WARNING**

The value for each address relationship (DP_DP_ID input; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

5. Supply the SD_BO_xx and SD_I_xx inputs of SENDDP with the send signals. To cut down on intermediate signals when transferring parameters, you can write the value directly to the instance DB of SENDDP using fully qualified access (for example, "Name SENDDP_1".SD_BO_02) before calling SENDDP.
6. Supply the RD_BO_xx and RD_I_xx outputs of RCVDP with the signals that you want to process further in other program sections or use fully qualified access to read the received signals directly in the associated instance DB in the program sections to be processed further (e.g., "Name RCVDP_1".RD_BO_02).
7. Supply the SUBBO_xx and SUBI_xx inputs of RCVDP with the fail-safe values that are to be output by RCVDP in place of the process data until communication is established for the first time after startup of the sending and receiving F-systems or in the event of an error in safety-related communication.
   - Specification of constant fail-safe values:

     For data of data type INT, you can enter constant fail-safe values directly as constants in the SUBI_xx input (initial value = "0"). If you want to specify a constant fail-safe value for data of data type BOOL, enter the "F_GLOBDB".VKE1 tag (fully qualified) in the SUBBO_xx input (initial value = "FALSE").
   - Specification of dynamic fail-safe values:

     If you want to specify dynamic fail-safe values, define a tag that you change dynamically through your safety program in an F-DB and specify this tag (fully qualified) in the SUBI_xx or SUBBO_xx input.

     **⚠ WARNING**

     Note that your safety program for dynamically changing the tag for a dynamic fail-safe value can only be processed after the call of the RCVDP, because prior to the RCVDP call, there must not be any network and no more than one other RCVDP instruction in the main safety block. You must therefore assign appropriate start values for these tags to be output by RCVDP in the first cycle after a startup of the F-system. *(S017)*
8. Configure the TIMEOUT inputs of the RCVDP and SENDDP instructions with the required monitoring time.

   **⚠ WARNING**

   It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal level is pending for at least as long as the assigned monitoring time. *(S018)*

   Information on calculating the monitoring times can be found in Monitoring and response times (Page 661).

9. Optional: Evaluate the ACK_REQ output of the RCVDP instruction, for example, in the standard user program or on the HMI system in order to query or to indicate whether user acknowledgment is required.
10. Supply the ACK_REI input of the RCVDP instruction with the acknowledgment signal for reintegration.
11. Optional: Evaluate the SUBS_ON output of the RCVDP or SENDDP instruction in order to query whether the RCVDP instruction is outputting the fail-safe values assigned in the SUBBO_xx and SUBI_xx inputs.
12. Optional: Evaluate the ERROR output of the RCVDP or SENDDP instruction, for example, in the standard user program or on the HMI system in order to query or to indicate whether a communication error has occurred.
13. Optional: Evaluate the SENDMODE output of the RCVDP instruction in order to query whether the F-CPU with the associated SENDDP instruction is in disabled safety mode (Page 293).

#### 9.1.3.4 Safety-related master-master communication:Limits for data transfer

**Note**

If the data quantities to be transmitted exceed the capacity of the SENDDP / RCVDP correlated instructions, a second (or third) SENDDP / RCVDP call can be used. This requires configuration of an additional connection via the DP/DP coupler. Whether or not this is possible with one single DP/DP coupler depends on the capacity restrictions of the DP/DP coupler.

## 9.1.4 Safety-related communication between IO controller and I-device

### 9.1.4.1 Configuring safety-related communication between IO controller and I-device

#### Introduction

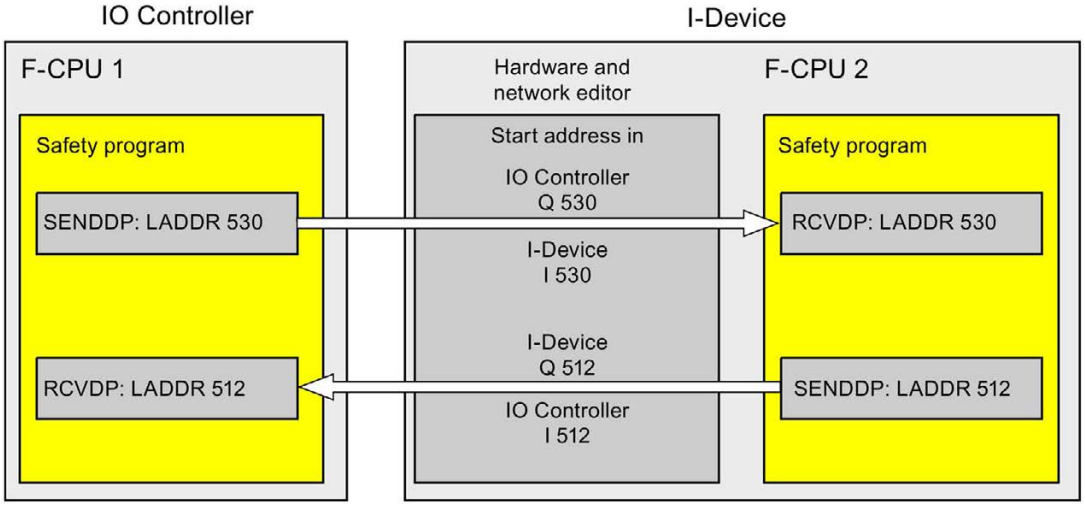
Safety-related communication between the safety program of the F-CPU of an IO controller and the safety program(s) of the F-CPU(s) of one or more I-devices takes place via IO controller-I-device connections (**F**-CD) in PROFINET IO, in the same way as in standard systems.

You do not need any additional hardware for IO controller-I-device communication.

The F-CPU to be used as an I-device must support the "IO-device" operating mode.

#### Configuring transfer areas

For every safety-related communication connection between two F-CPUs, you must configure transfer areas in the *hardware and network editor*. The figure below shows how both of the F-CPUs are able to send and receive data (bidirectional communication).




The transfer area is assigned a label when it is created to identify the communication relationship, for example "F-CD_PLC_2 PLC_1_1" for the first F-CD connection between IO-controller F-CPU 1 and I-device F-CPU 2.

You assign the start addresses of the transfer areas to the LADDR parameter of the SENDDP and RCVDP instructions in the safety programs.

### Procedure for configuration

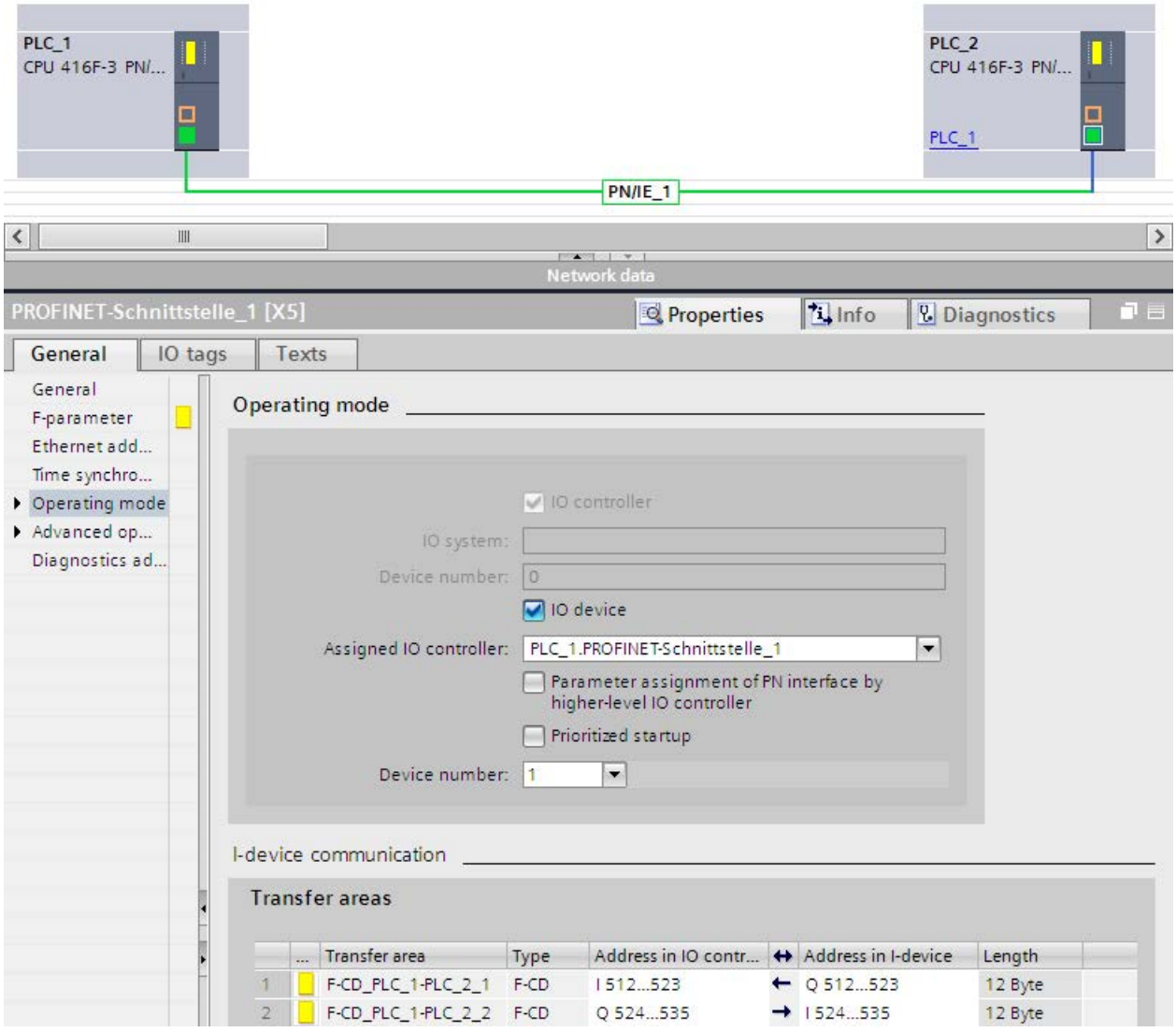
The procedure for configuring safety-related IO controller-I-device communication is identical to that in the standard system.

Proceed as follows:

1. Insert two F-CPUS from the "Hardware catalog" task card into the project.
2. Activate the "IO Device" mode for F-CPU 2 in the properties of its PN interface and assign this PN interface to a PN interface of F-CPU 1.
3. Select the PROFINET interface of F-CPU 2. Under "Transfer areas", you create an F-CD connection (type "F-CD") for sending to the IO controller (←). The F-CD connection is shown in yellow in the table and the address areas in the I-device and IO controller assigned outside of the process image are displayed.

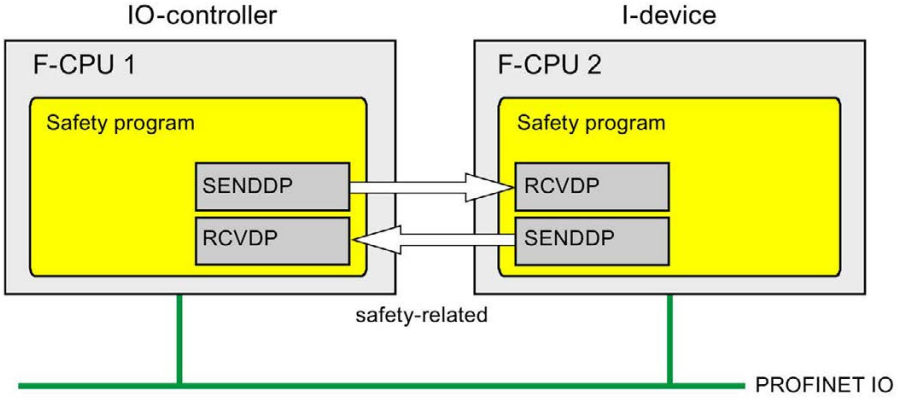
   In addition, an acknowledgment connection is created automatically for each F-CD connection. (see "Transfer area details").

4. Create an additional F-CD connection for receiving from the IO controller.
5. In the transfer area you just created, click the arrow in order to change the transfer direction to receiving from IO controller (→).



### 9.1.4.2 Safety-related IO controller-I-device communication via SENDDP and RCVDP

#### Communication via SENDDP and RCVDP instructions




Safety-related communication between the F-CPUs of the IO controller and an I-device makes use of the SENDDP and RCVDP instructions for sending and receiving, respectively. These can be used to perform a fail-safe transfer of a *fixed* amount of fail-safe data of the data type INT or BOOL.

You can find these instructions in the "Instructions" task card under "Communication". The RCVDP instruction **must** be called at the start of the main safety block. The SENDDP instruction **must** be called at the end of the main safety block.

Note that the send signals are not sent until after the SENDDP instruction call at the end of execution of the relevant F-runtime group.

A detailed description of the SENDDP and RCVDP instructions can be found in SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500) (Page 646).

### 9.1.4.3 Programming safety-related IO controller I-device communication

#### Requirement for programming

The transfer areas must be configured.

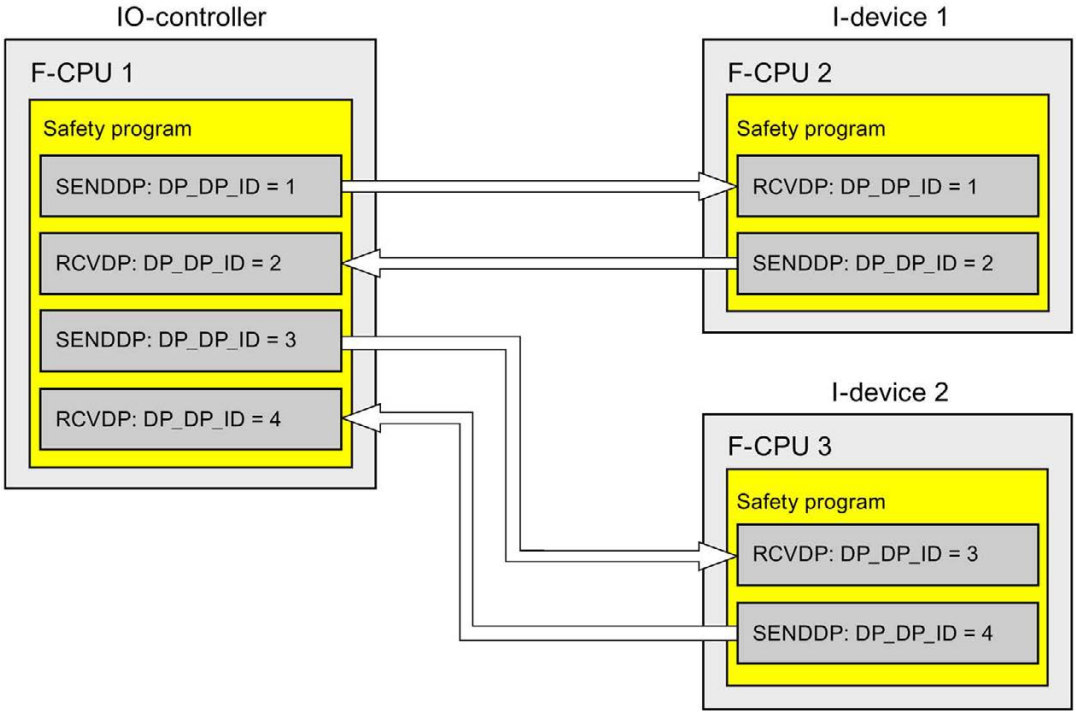
#### Programming procedure

The procedure for programming safety-related IO controller-I-device communication is the same as that for programming safety-related IO controller-IO controller communication (see Program safety-related IO controller-IO controller communication (Page 171)).

The assignment of the start addresses of the transfer areas to the LADDR parameter of the SENDDP/RCVDP instructions can be obtained from the following table.

| Instruction | Start address LADDR | |
|---|---|---|
| | From row | From column |
| SENDDP in the IO controller | → | Address in the IO controller |
| RCVDP in the IO controller | ← | Address in the IO controller |
| SENDDP in the I-device | ← | Address in the IO device |
| RCVDP in the I-device | → | Address in the IO device |

The figure below contains an example of how to specify the address relationships for the inputs of the SENDDP and RCVDP instructions for four safety-related IO controller-I-device communication relationships.



**⚠ WARNING**

The value for each address relationship (DP_DP_ID input; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

**⚠ WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal state is pending for at least as long as the assigned monitoring time. *(S018)*

Information on calculating the monitoring times can be found in Monitoring and response times (Page 661).

### 9.1.4.4 Safety-related IO-Controller-IO-Device communication - Limits for data transfer

## Limits for data transfer

If the amount of data to be transferred is greater than the capacity of related SENDDP/RCVDP instructions, you can use additional SENDDP/RCVDP instructions. Configure additional transfer areas for this purpose. Remember the maximum limit of 1440 bytes of input data or 1440 bytes of output data for transfer between an I-device and a IO controller.

The following table shows the amount of output and input data assigned in safety-related communication connections:

| Safety-related communication | Communication connection | Assigned input and output data | | | |
|---|---|---|---|---|---|
| | | In the IO controller | | In the I-device | |
| | | Output data | Input data | Output data | Input data |
| IO controller-I-Device | **Sending:**<br>I-Device 1 to IO controller | 6 bytes | 12 bytes | 12 bytes | 6 bytes |
| | **Receiving:**<br>I-Device 1 from IO controller | 12 bytes | 6 bytes | 6 bytes | 12 bytes |

Consider all additional configured safety-related and standard communication connections (transfer areas of type F-CD and CD) for the maximum limit of 1440 bytes of input data or 1440 bytes of output data for transfer between an I-device and an IO controller . In addition, data are assigned for internal purposes such that the maximum limit may be reached sooner.

When the limit is exceeded, a corresponding error message is displayed.

## 9.1.5 Safety-related master-I-slave communication

### 9.1.5.1 Configuring safety-related master-I-slave communication

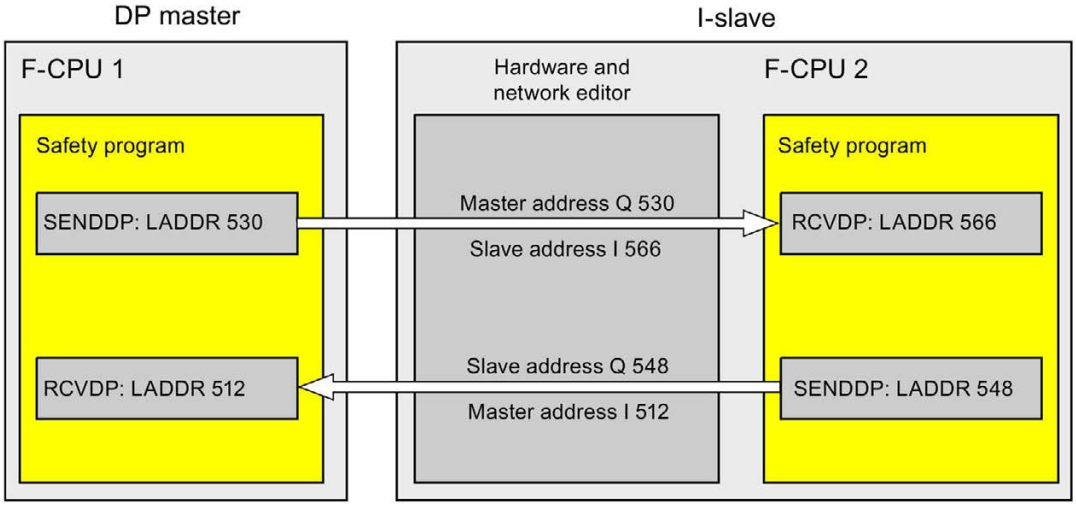
#### Introduction

Safety-related communication between the safety program of the F-CPU of a DP master and the safety program(s) of the F-CPU(s) of one or more I-slaves takes place over master-I-slave connections (**F-**MS), as in standard systems.

You do not need a DP/DP coupler for master-I-slave communication.

#### Configuring transfer areas

For every safety-related communication connection between two F-CPUs, you must configure transfer areas in the *hardware and network editor*. The figure below shows how both of the F-CPUs are able to send and receive data (bidirectional communication).




The transfer area is assigned a label when it is created to identify it as the communication relationship, for example "F-MS_PLC_2-PLC_1_1" for the first F-MS connection between DP master F-CPU 1 and I-slave F-CPU 2.

You assign the start addresses of the transfer areas to the LADDR parameter of the SENDDP and RCVDP instructions in the safety programs.

## Procedure for configuration

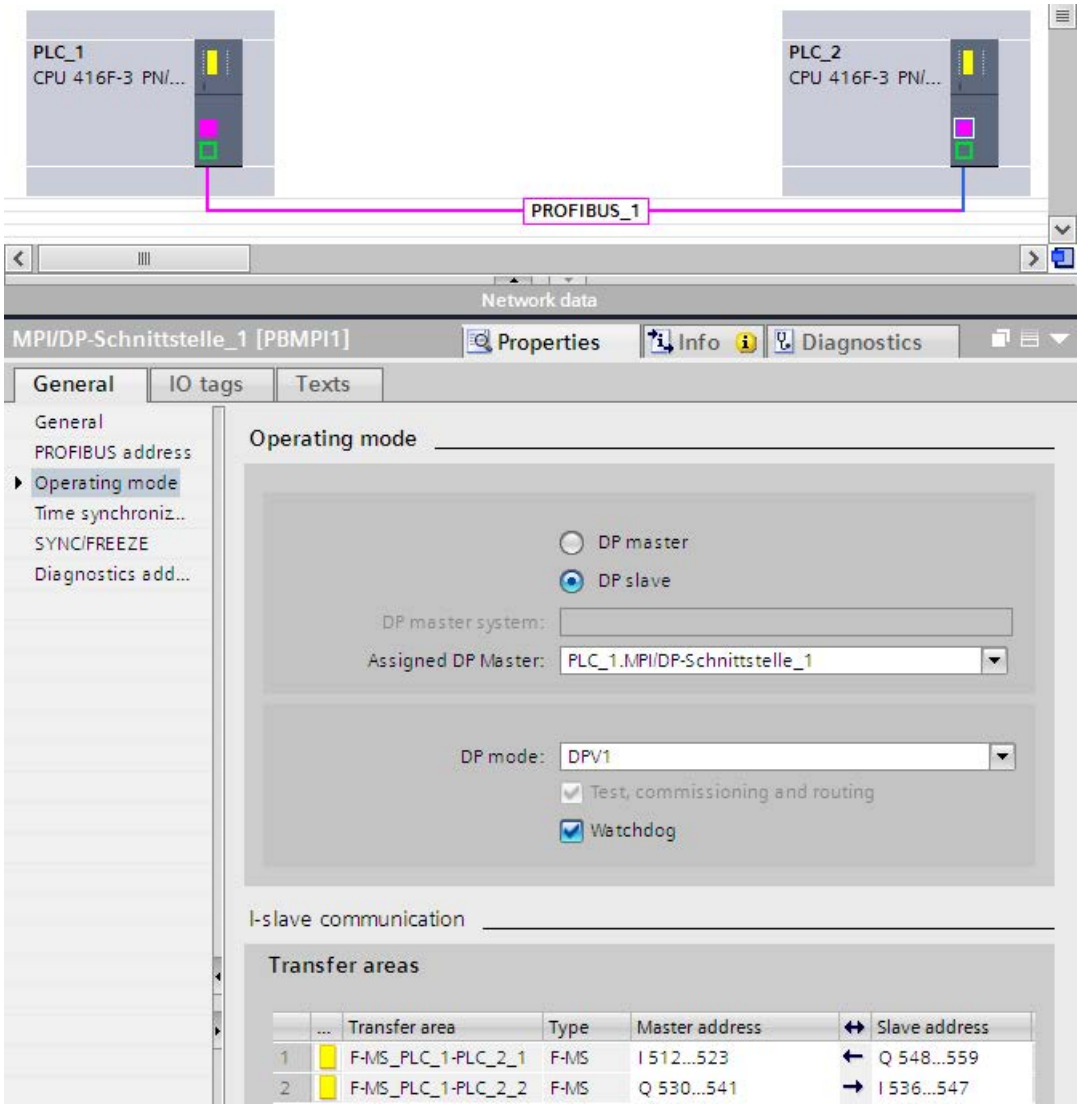
The procedure for configuring safety-related master-I-slave communication is identical to that in the standard system.

Proceed as follows:

1. Insert two F-CPUS from the "Hardware catalog" task card into the project.
2. Activate the "DP slave" mode (I-slave) for F-CPU 2 in the properties of its DP interface and assign this DP interface to a DP interface of F-CPU 1.
3. Select the PROFIBUS interface of F-CPU 2. Under "Transfer areas", you create an F-MS connection (type "F-MS") for sending to the DP master (←). The F-MS connection is shown in yellow in the table and the transfer areas in the I-slave and DP master assigned outside of the process image are displayed.

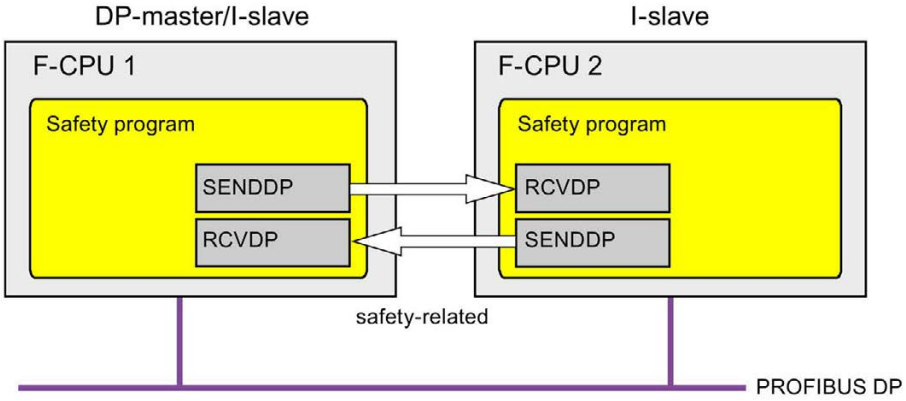
   In addition, an acknowledgment connection is created automatically for each F-MS connection. (see "Transfer area details").

4. Create an additional F-MS connection for receiving from the DP master.
5. In the transfer area you just created, click the arrow in order to change the transfer direction to receiving from DP master (→).



### 9.1.5.2 Safety-related master-I-slave or I-slave-I-slave communication via SENDDP and RCVDP

#### Communication via SENDDP and RCVDP instructions




Safety-related communication between the F-CPUs of the DP master and an I-slave or between the F-CPUs of several I-slaves makes use of the SENDDP and RCVDP instructions for sending and receiving, respectively. These can be used to perform a fail-safe transfer of a *fixed* amount of fail-safe data of the data type INT or BOOL.

You can find these instructions in the "Instructions" task card under "Communication". The RCVDP instruction **must** be called at the start of the main safety block. The SENDDP instruction **must** be called at the end of the main safety block.

Note that the send signals are not sent until after the SENDDP instruction call at the end of execution of the relevant F-runtime group.

A detailed description of the SENDDP and RCVDP instructions can be found in SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500) (Page 646).

### 9.1.5.3 Program the safety-related master-I-slave or I-slave-I-slave communication

#### Requirements

The transfer areas must be configured.

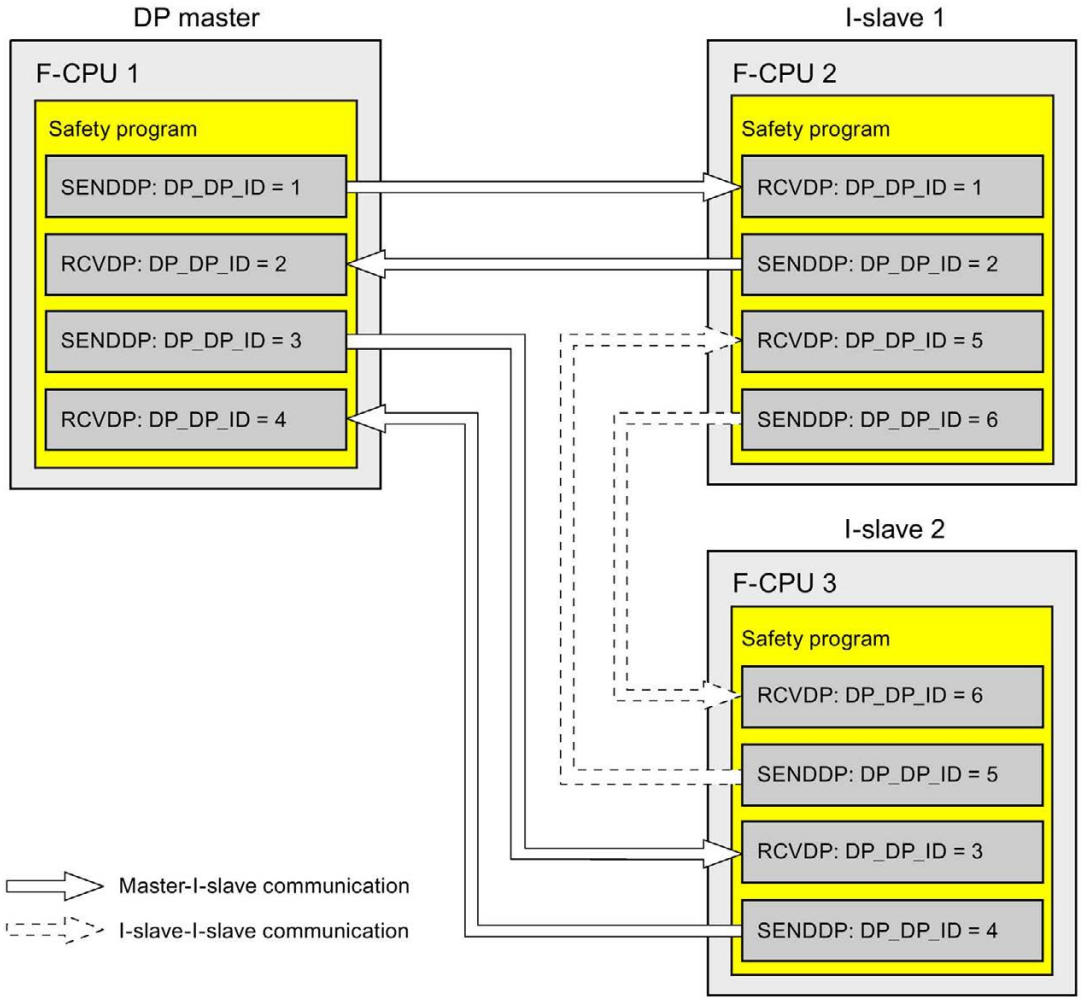
#### Programming procedure

The procedure for programming safety-related master-I-slave communication or I-slave-I-slave communication is the same as that for programming safety-related master-master communication (see Program safety-related master-master communication (Page 181)).

The assignment of the start addresses of the transfer areas to the LADDR parameter of the SENDDP/RCVDP instructions can be obtained from the following table.

| Instruction | Start address LADDR | |
|---|---|---|
| | From row | From column |
| SENDDP in the DP master | → | Master address |
| RCVDP in the DP master | ← | Master address |
| SENDDP in the I-slave | ← | Slave address |
| RCVDP in the I-slave | → | Slave address |

The figure below contains an example of how to specify the address relationships at the inputs of SENDDP and RCVDP instructions for four safety-related master-I-slave and two I-slave-I-slave communications relationships.




**WARNING**

The value for each address relationship (DP_DP_ID input; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

**WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal level is pending for at least as long as the assigned monitoring time. *(S018)*

Information on calculating the monitoring times can be found in Monitoring and response times (Page 661).

### 9.1.5.4 Limits for data transfer of safety-related master-I-slave or I-slave-I-slave communication

#### Limits for data transfer

If the amount of data to be transferred is greater than the capacity of related SENDDP/RCVDP instructions, you can use additional SENDDP/RCVDP instructions. Configure additional transfer areas for this purpose. Note the maximum limit of 244 bytes of input data or 244 bytes of output data for transfer between an I-slave and a DP master.

The following table shows the amount of output and input data assigned in safety-related communication connections:

| Safety-related communication | Communication connection | Assigned input and output data | | | | | |
|---|---|---|---|---|---|---|---|
| | | DP master | | I-slave 1 | | I-slave 2 | |
| | | Output data | Input data | Output data | Input data | Output data | Input data |
| Master-I-slave | **Sending:**<br>I-slave 1 to DP master | 6 bytes | 12 bytes | 12 bytes | 6 bytes | — | — |
| | **Receiving:**<br>I-slave 1 from DP master | 12 bytes | 6 bytes | 6 bytes | 12 bytes | — | — |
| I-slave-I-slave | **Sending:**<br>I-slave 1 to I-slave 2 | — | 18 bytes | 12 bytes | 6 bytes | 6 bytes | 12 bytes |
| | **Receiving:**<br>I-slave 1 from I-slave 2 | — | 18 bytes | 6 bytes | 12 bytes | 12 bytes | 6 bytes |

Consider all additional configured safety-related and standard communication connections (transfer areas of type F-MS-, F-DX-, F-DX-Mod., MS-, DX- and DX-Mod) for the maximum limit of 244 bytes of input data or 244 bytes of output data for transfer between an I-device and a DP master F-MS, F-DX, F-DX-Mod., MS, DX and DX-Mod). If the maximum limit of 244 bytes of input data or 244 bytes of output data is exceeded, you will receive a corresponding error message.

## 9.1.6 Safety-related I-slave-I-slave communication

### 9.1.6.1 Configure safety-related I-slave-I-slave communication

#### Introduction

Safety-related communication between the safety program of the F-CPUs of I-slaves takes place using direct data exchange (**F-**DX) – same as in standard programs.

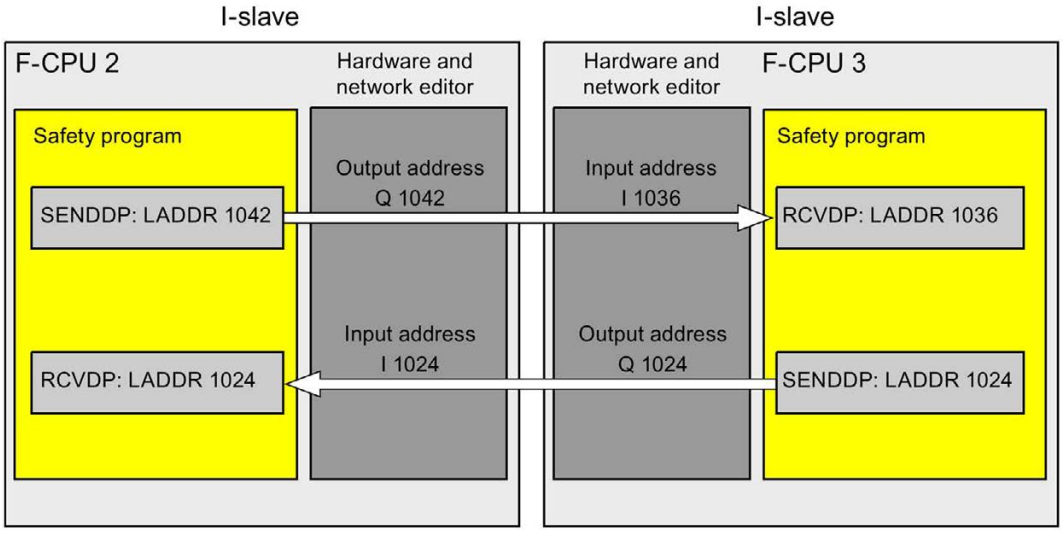
You do not need any additional hardware for I-slave-I-slave communication.

I-slave-I-slave communication is also possible:

- If the assigned DP master is a standard CPU that supports direct data exchange
- when instead of a DP master, an IO controller is networked with the I-slaves via an IE/PB link

#### Configuring transfer areas

For every safety-related communication connection between two I-slaves, you must configure transfer areas in the *hardware and network editor*. In the figure below, both of the I-slaves are to be able to send and receive data (bidirectional communication).




The transfer area is assigned a label when it is created to identify it as the communication relationship, for example "F-DX_PLC_2-PLC_1_1" for the first F-DX connection between F-CPU 1 and CPU 2.

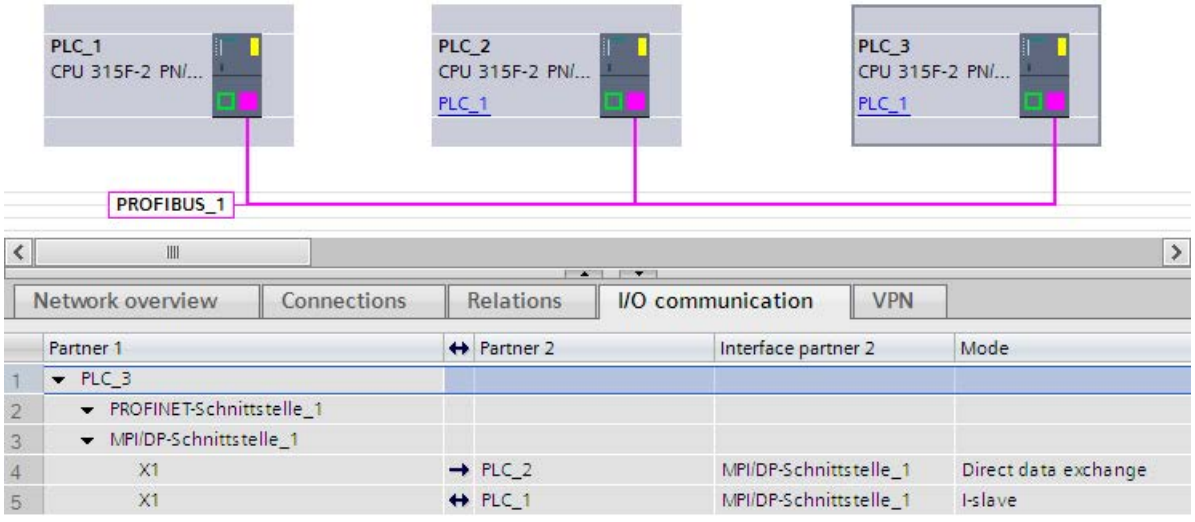
You assign the start addresses of the transfer areas to the LADDR parameter of the SENDDP and RCVDP instructions in the safety programs.

## Procedure for configuration

The procedure for configuring safety-related I-slave-I-slave communication is identical to that in the standard system. Proceed as follows:

1. Insert three F-CPUS from the "Hardware catalog" task card in the project.
2. Activate "DP slaves" mode (I-slave) for F-CPU 2 and F-CPU 3 in the properties of their DP interfaces and assign these DP interfaces to a DP interface of F-CPU 1.
3. Select the DP interface of F-CPU 3 in the network view.
4. Select the "I/O communication" tab.
5. Use a drag-and-drop operation in the network view to move F-CPU 2 to the "Partner 2" column on the "I/O-communication" tab.

   This creates a line with "Direct data exchange" mode for sending to the I-slave (F-CPU 2) (→).

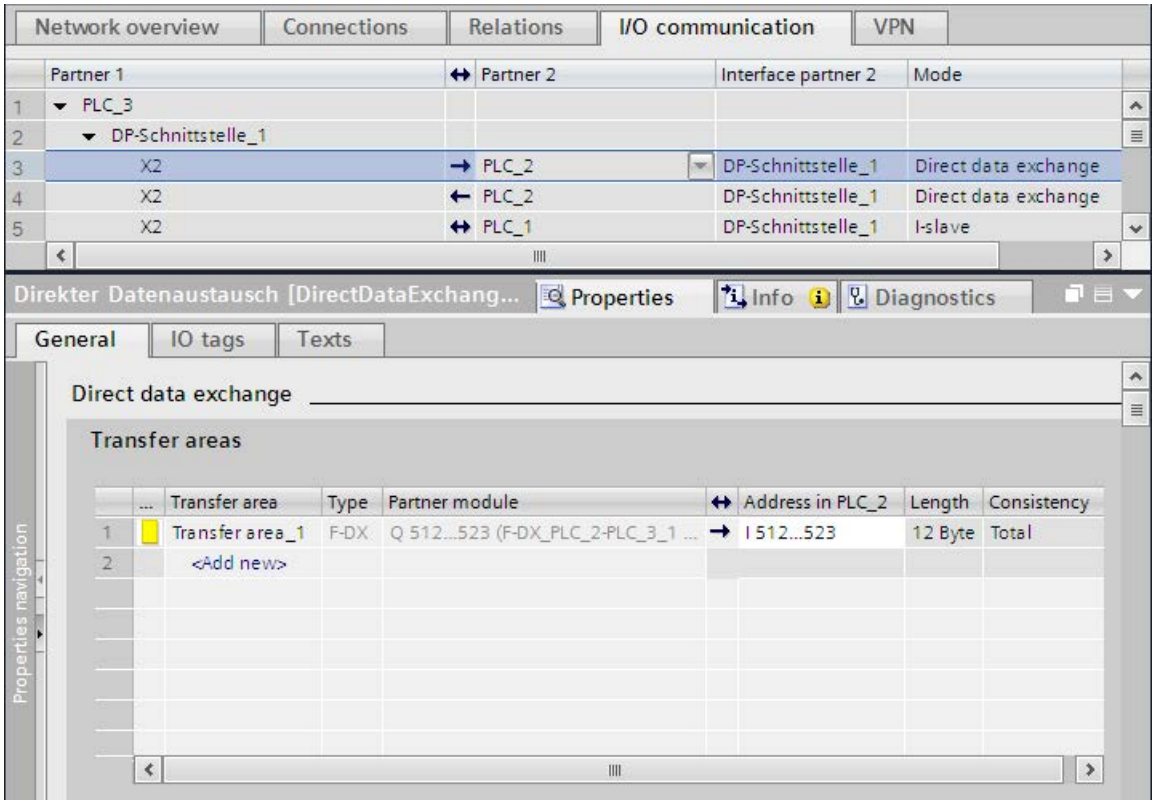


6. In "Transfer areas" ("Direct data exchange" table), create an F-DX connection (type "F-DX") for sending to the I-slave (F-CPU 2) (→). The F-DX connection is shown in yellow in the table and the transfer areas in the I-slaves assigned outside of the process image (PLC_2 and PLC_3) are displayed.

   In addition, a line with "Direct data exchange" mode for receiving from the I-slave (F-CPU 2) (→) is created automatically in the "I/O communication" tab, and an acknowledgment connection (←, transfer area x_Ack) is created automatically in the associated "Direct data exchange" table.

   One transfer area (type F-MS) for the master CPU (disabled in display) is created for in the "I-slave communication table" of each I-slave.

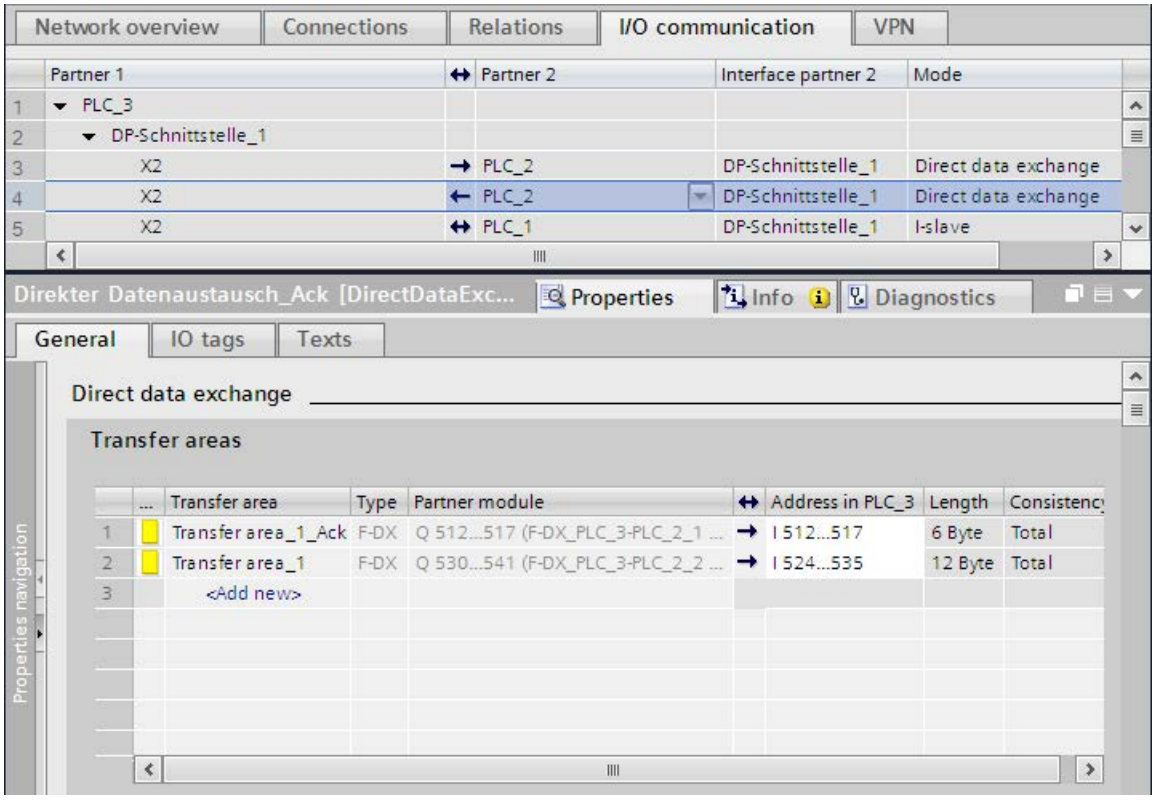
   This completes the configuration for sending to F-CPU 2.



7. In the "I/O communication" tab, select the automatically created line with "Direct data exchange" mode for receiving from the I-slave (F-CPU 3) (←).
8. In "Transfer areas" ("Direct data exchange" table), create another F-DX connection for receiving from the I-slave (F-CPU 3).

   In this case, as well, an acknowledgment connection (←, transfer area x_Ack) is created automatically in the "Data exchange table" and two transfer areas (type F-MS) for the master CPU (disabled in display) are created in the "I-slave communication" table of both I-slaves.

   This completes the configuration for receiving from F-CPU 2.




## Changing disabled local address areas of the transfer areas

In order to change the disabled local address area of "Transfer area x", you must change the address area of the corresponding acknowledgment connection "Transfer area x_Ack".

1. In "I/O communication", select the line with the arrow pointing in the same direction as the arrow of "Transfer area x" in the "Direct data exchange" table.
2. Then select the line with "Transfer area x_Ack" in the "Direct data exchange" table.
3. Change the address area there.

### 9.1.6.2 Safety-related I-slave-I-slave communication via SENDDP and RCVDP

#### Reference

The description of the communication via SENDDP and RCVDP for safety-related I-slave-I-slave communication can be found in Safety-related master-I-slave or I-slave-I-slave communication via SENDDP and RCVDP (Page 195).

### 9.1.6.3 Programming safety-related I-slave-I-slave communication

#### Reference

The description of the programming of safety-related I-slave-I-slave communication can be found in Program the safety-related master-I-slave or I-slave-I-slave communication (Page 196).

The assignment of the start addresses of the transfer areas to the LADDR parameter of the SENDDP/RCVDP instructions can be obtained from the following table.

| Instruction | Start address LADDR | |
|---|---|---|
| | From row | From column |
| SENDDP in the 1st I-slave | → | Address in the <1st I-slave> (in example column "Address in PLC_2") |
| RCVDP in the 1st I-slave | ← | Address in the <1st I-slave> (in example column "Address in PLC_2") |
| SENDDP in the 2nd I-slave | ← | Address in the <2nd I-slave> (in example column "Address in PLC_3") |
| RCVDP in the 2nd I-slave | → | Address in the <2nd I-slave> (in example column "Address in PLC_3") |

### 9.1.6.4 Limits for data transfer of safety-related I-slave-I-slave communication

#### Limits for data transfer

The description of the limits for the data transfer of safety-related I-slave-I-slave communication can be found in Limits for data transfer of safety-related master-I-slave or I-slave-I-slave communication (Page 199).

## 9.1.7 Safety-Related I-Slave-Slave Communication

### 9.1.7.1 Configuring Safety-Related I-Slave-Slave Communication

#### Introduction

Safety-related communication between the safety program of the F-CPU of an I-slave and F-I/O in a DP slave takes place using direct data exchange (**F-**DX-Mod), same as in standard programs.

You do not need any additional hardware for I-slave-slave communication.

I-slave-slave communication is also possible:

- when the assigned DP master is a standard CPU, if the standard CPU supports direct data exchange
- when instead of a DP master, an IO controller is networked with the I-slaves via an IE/PB link

An F-I/O DB is automatically generated for each F-I/O when it is configured in the *hardware and network editor*; this is required for the F-I/O access via safety-related I-slave-slave communication. The F-I/O DB is initially created in the safety program of the DP master, provided it is an F-CPU with F-activation. Only with the setup of the F-DX-Mod connection is the F-I/O DB created in the safety program of the I-slave.

The process image of the inputs is used to access the channels of the F-I/O in the safety program of the F-CPU of the I-slave (see description in Safety-Related I-Slave-Slave Communication - F-I/O Access (Page 210)).

#### Restrictions

**Note**

Safety-related I-slave-slave communication is possible for F-I/O in a DP slave that supports safety-related I-slave-slave communication, e.g. for all ET 200S F-modules with IM 151-x HIGH FEATURE and all S7-300 fail-safe signal modules with IM 153-2, as of item no. 6ES7153-2BA01-0XB0, firmware > V4.0.0.

Safety-related I-slave-slave communication is not supported for ET 200SP F-modules and ET 200MP F-modules.

**Note**

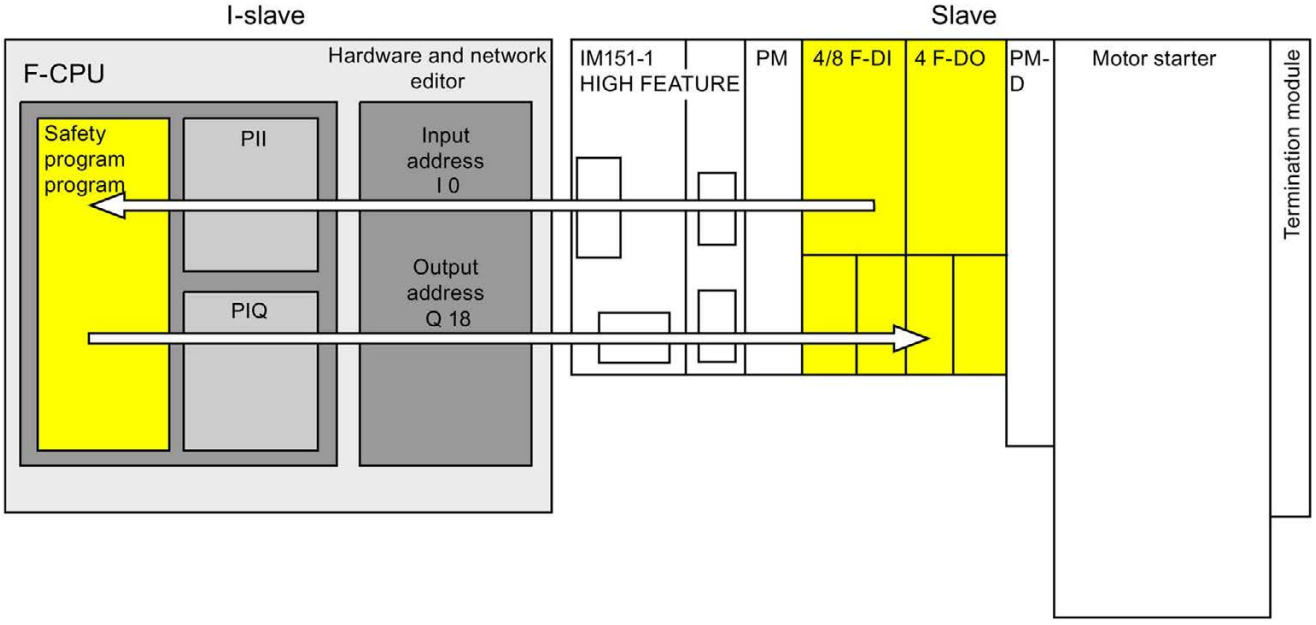
With safety-related I-slave-slave communication, make sure that the CPU of the DP master is powered up before the F-CPU of the I-slave.

Otherwise, depending on the F-monitoring time specified for the F-I/O, the F-system can detect an error in safety-related communication (communication error) between the F-CPU and the F-I/O assigned to the I-slave. This means that the F-I/O are not reintegrated automatically after F-system startup. They are instead only reintegrated after a user acknowledgment with a positive edge in the ACK_REI tag of the F-I/O DB (see also After communication errors (Page 142) and After startup of F-system (Page 140)).

## Configuring transfer areas

For every safety-related communication connection between an I-slave and slave, you must configure transfer areas in the *hardware and network editor*.

The transfer area is assigned a label when it is created to identify it as the communication relationship, for example "F-DX-Mod_PLC_2-PLC_1_1" for the first F-DX-Mod connection between F-CPU 1 and F-CPU 2.

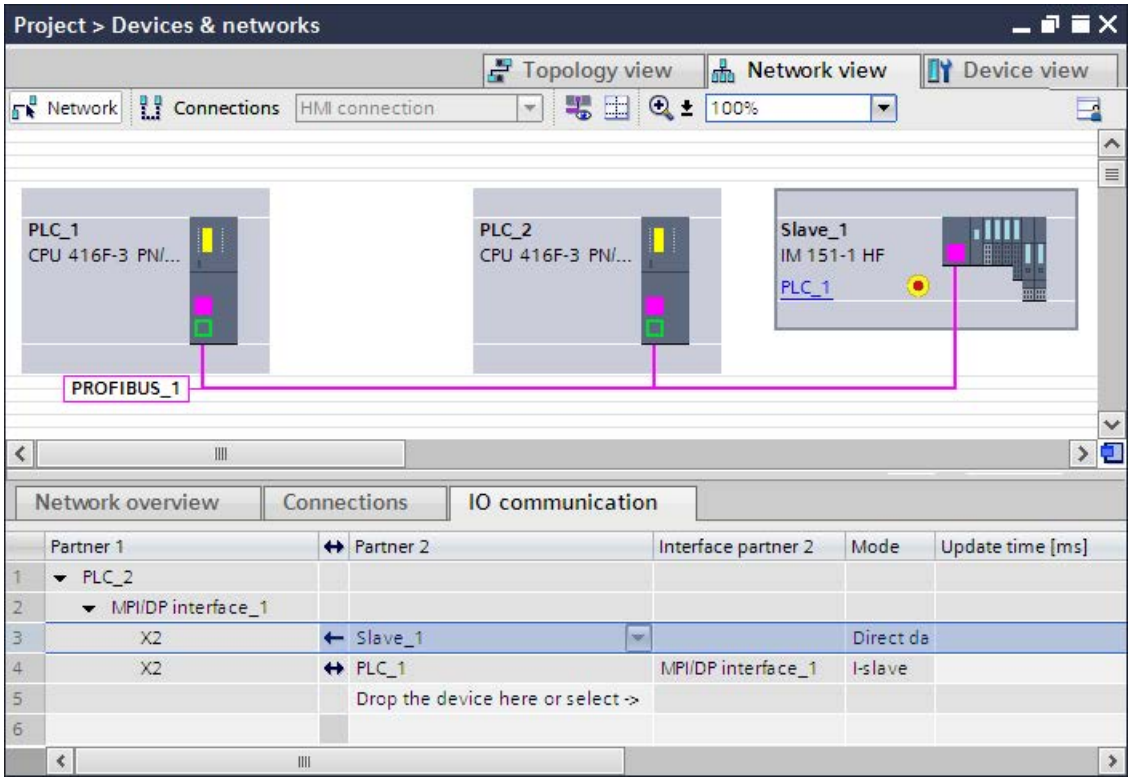


## Configuration procedure using the example of an ET 200S with F-modules in the slave

The procedure for configuring safety-related I-slave-slave communication is identical to that in the standard system.

Proceed as follows:

1. Insert two F-CPUS from the "Hardware catalog" task card into the project.
2. Insert a suitable DP slave, e.g. IM 151-1 HF, article no. 6ES7151-1BA0... from the "Hardware catalog" task card into the network view of the *hardware and network editor*.
3. Assign a 4/8 F-DI module and a 4 F-DO-module in the device view of the ET 200S.
4. Activate "DP slave" mode (I-slave) for F-CPU 2 in the properties of its DP interface and assign this to F-CPU 1.
5. Assign the DP interface of the IM 151-1 HF to the DP master (F-CPU 1).
6. Select the DP interface of F-CPU 2 (I-slave) in the network view.
7. Select the "I/O communication" tab.
8. Use a drag-and-drop operation in the network view to move the ET 200S to the "Partner 2" column in the "I/O-communication" tab.

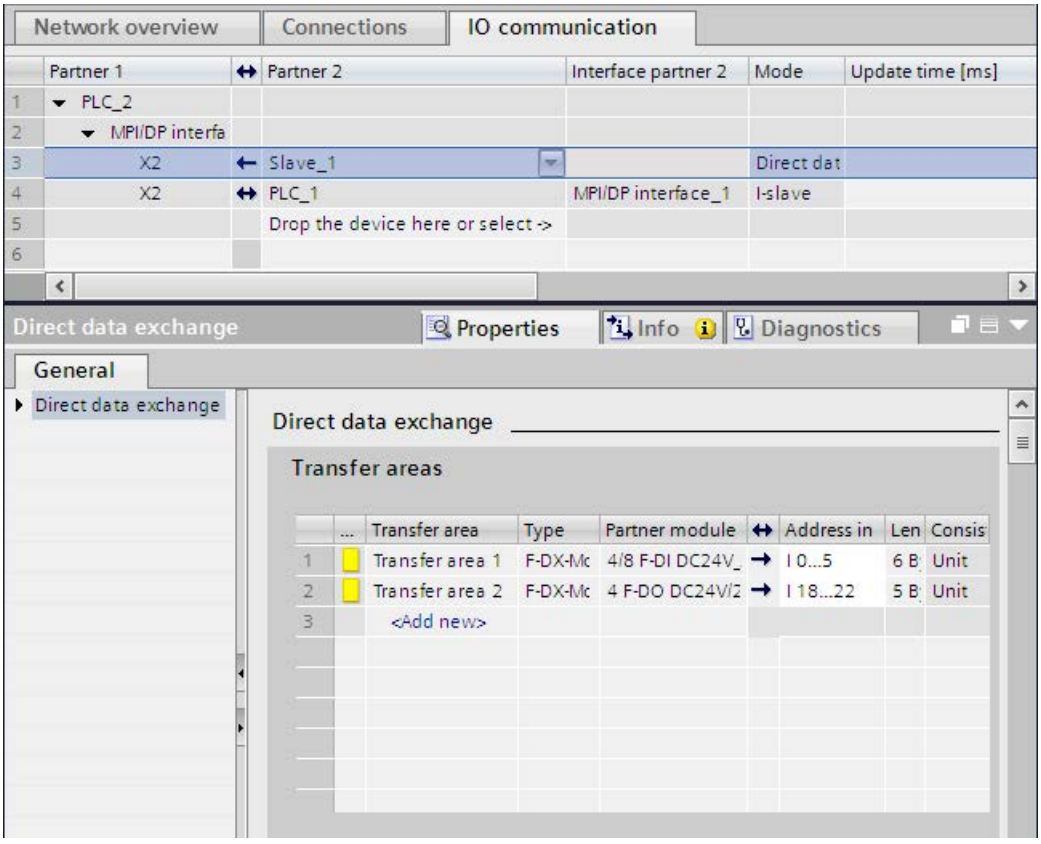



9. In "Transfer areas", create an F-DX-Mod connection (type "F-DX-Mod"). The F-DX-Mod connection is marked in yellow in the table. The addresses for the "partner module" 4/8 F-DI in the I-Slave (PLC_2) are displayed. You can change the addresses directly in the table, if required.

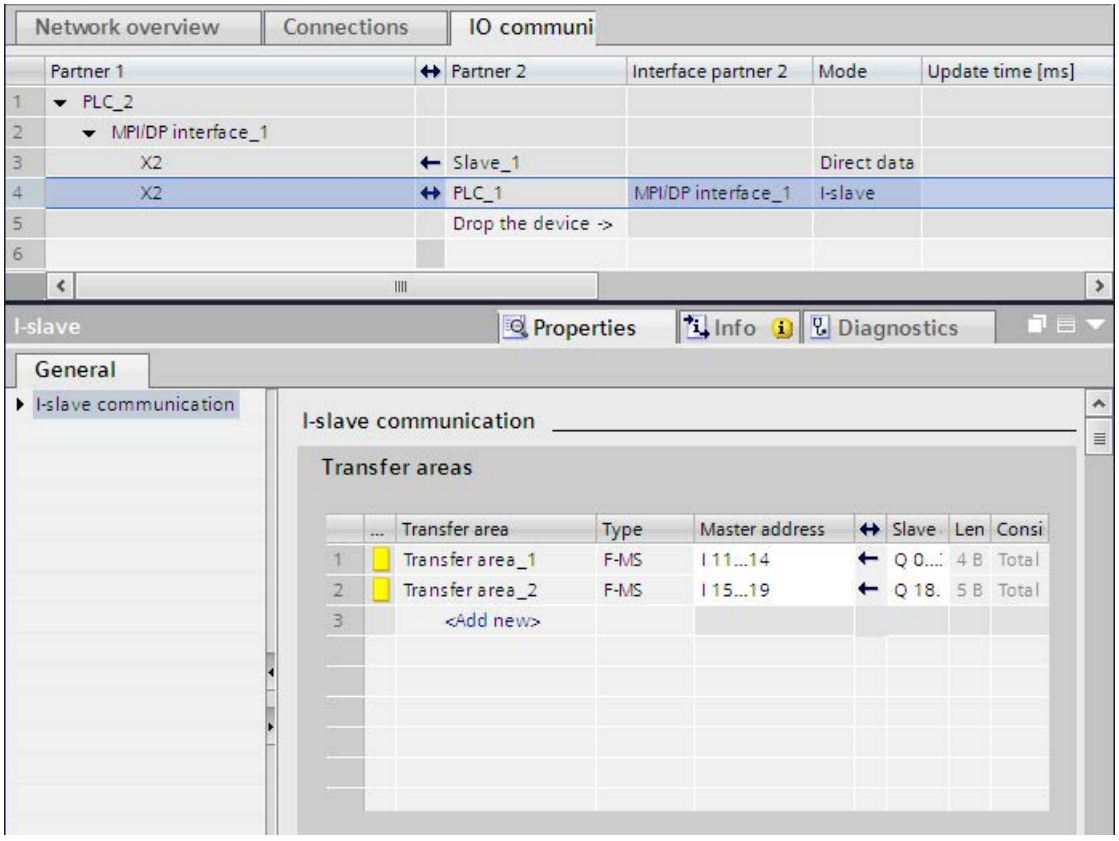
   This completes the configuration for the 4/8 F-DI module.

10. In "Transfer areas", create another F-DX-Mod connection.
11. Change the partner module to the 4 F-DO module, either directly in the "Transfer areas" table or in the details of transfer area 2, if the 4 F-DO module was not already selected.

    This completes the configuration for the 4 F-DO module.



In the "I-slave communication table" of the I-slave, a transfer area (type F-MS) for the master CPU (disabled in display) is created for each F-DX-Mod connection.




## Change in configuration of I-slave-slave communication

**⚠ WARNING**

If you have reconfigured I-slave-slave communication for an F-I/O or have deleted an existing I-slave-slave communication, you must compile the hardware configuration of the DP master as well as the hardware configuration of the I-slave and download them to the DP master and I-slave, respectively.

The collective F-signature in the F-CPU of the I-slave and the collective F-signature in the F-CPU of the DP master (if a safety program exists there, too) are set to "0". You must then recompile the safety program(s). *(S019)*

### 9.1.7.2 Safety-Related I-Slave-Slave Communication - F-I/O Access

#### Access via the process image

In safety-related I-slave-slave communication, you use the process image (PII or PIQ) to access the F-I/O in the safety program of the F-CPU of the I-slave. This is the same as F-I/O access to F-I/O that are directly assigned to an I-slave or DP master. In the I-slave you access the F-I/O with the addresses that were assigned for the F-DX-Mod connection in "Transfer areas" ("Direct data exchange" table).

In this case, ignore the displayed operand area. Access F-I/O with inputs using the PII and F-I/O with outputs using PIQ.

Information on I/O access can be found in F-I/O access (Page 122).

### 9.1.7.3 Limits for data transfer of safety-related I-slave-I-slave communication

#### Limits for data transfer

Note the maximum limit of 244 bytes of input data or 244 bytes of output data for transfer between an I-slave and a DP master.

An example of the amount of output data and input data that are assigned for safety-related communication is shown in the table below for an ET 200S 4/8 F-DI and an ET 200S 4 F-DO:

| Safety-related communication | Communication connection | Assigned input and output data* | |
|---|---|---|---|
| | | Between I-slave and DP master | |
| | | Output Data in the I-slave | Input data in the I-slave |
| I-slave-slave | I-slave-slave communication with 4/8 F-DI | 4 bytes | 6 bytes |
| | I-slave-slave communication with 4 F-DO | 5 bytes | 5 bytes |

* Example for 4/8 F-DI and 4 F-DO of ET 200S

Consider all additional configured safety-related and standard communication connections (F-MS, F-DX, F-DX-Mod., MS, DX und DX-Mod. connections) for the maximum limit of 244 bytes of input data or 244 bytes of output data for transfer between an I-slave and a DP master. If the maximum limit of 244 bytes of input data or 244 bytes of output data is exceeded, you will receive a corresponding error message.

## 9.1.8 Safety-related IO controller-I-slave communication

### Introduction

Safety-related communication between the safety program of the F-CPU of an IO-controller and the safety program(s) of the F-CPU(s) of one or more I-slaves takes place over master-I-slave connections (**F-**MS), as in standard systems.

### IE/PB link

For the safety-related IO-controller-I-slave communication, the IE/PB link is mandatory. Each of the two F-CPUs is linked to the IE/PB link by means of its PROFIBUS DP or PROFINET-interface.

**Note**

If you are using an IE/PB link, you must take this into account when configuring the F-specific monitoring times and when calculating the maximum response time of your F-system (see also Monitoring and response times (Page 661)).

Note that the Excel file for calculating response times (http://support.automation.siemens.com/WW/view/en/49368678/133100) for S7-300/400 F-CPUs does not support all conceivable configurations.

### Reference

The information on safety-related master-I-slave communication in Safety-related master-I-slave communication (Page 192) also applies.

## 9.1.9 Safety-related communication via S7 connections

### 9.1.9.1 Configuring safety-related communication via S7 connections

#### Introduction

Safety-related communication between the safety programs of F-CPUs via S7 connections takes place by means of established S7 connections that you create in the network view of the *hardware and network editor* - same as in standard programs.

#### Restrictions

**Note**

In SIMATIC Safety, S7 connections are generally permitted only via Industrial Ethernet.

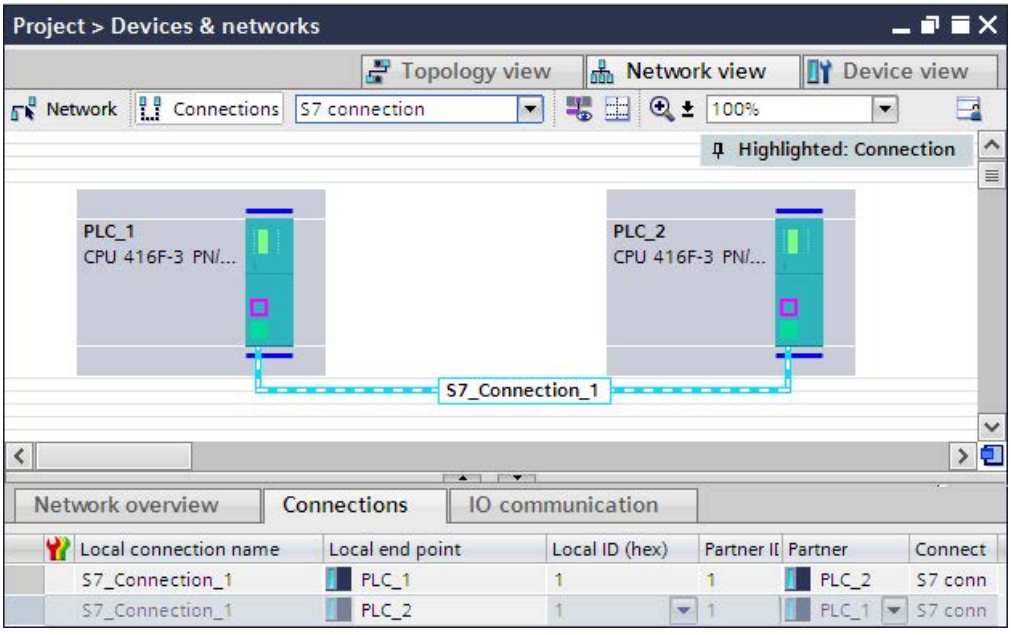
Safety-related communication via S7 connections is possible from and to the following CPUs:

- S7-300 F-CPUs via the integrated PROFINET interface
- S7-400 F-CPUs via the integrated PROFINET interface or a CP 443-1 Advanced-IT

## Creating S7 connections

For each connection between two F-CPUs, you must create an S7 connection in the network view of the *hardware and network editor*.

For every end-point of a connection, a local and a partner ID is automatically assigned from the perspective of the end-point (the F-CPU). If necessary, you can change both IDs in the "Connections" tab. You assign the local ID to the "ID" parameter of the SENDS7 and RCVS7 instructions in the safety programs.

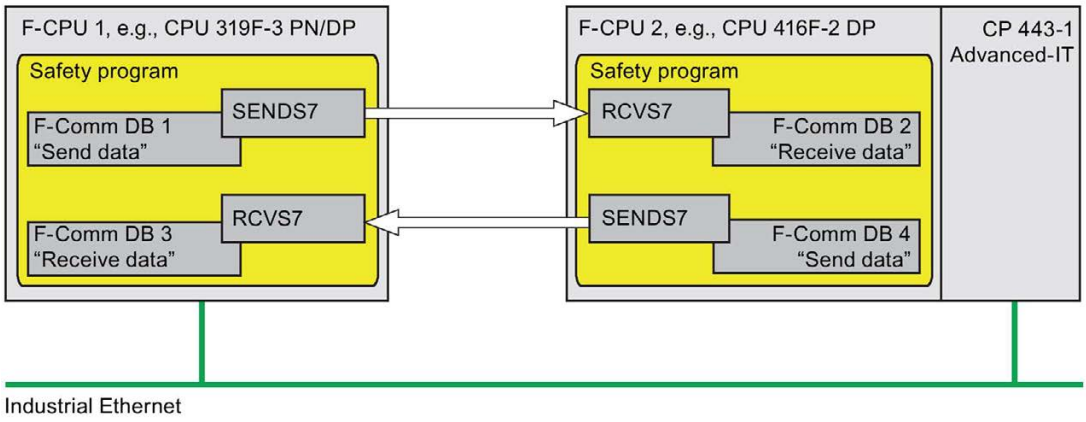



## Procedure for configuring S7 connections

You configure the S7 connections for safety-related CPU-to-CPU communication in the same way as in *STEP 7 Professional* (see *Help on STEP 7 Professional* "S7 connections").

### 9.1.9.2 Communication via SENDS7, RCVS7, and F-Communication DB

#### Communication via the SENDS7 and RCVS7 instructions




You use the **SENDS7 and RCVS7** instructions for fail-safe sending and receiving of data via S7 connections.

These instructions can be used to transmit a specified amount of fail-safe data of data types BOOL, INT, WORD, DINT, DWORD, and TIME in a fail-safe manner. The fail-safe data are stored in F-DBs (F-communication DBs) that you have created.

You can find these instructions in the "Instructions" task card under "Communication". The RCVS7 instruction **must** be called at the start of the main safety block. The SENDS7 instruction **must** be called at the end of the main safety block.

Note that the send signals are sent only after calling the SENDS7 instruction at the end of the relevant F-runtime group execution.

A detailed description of the SENDS7 and RCVS7 instructions is found in SENDS7 and RCVS7: Communication via S7 connections (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 654).

#### F-communication DB

For each connection, send data are stored in an F-DB (F-communication DBx) and receive data are stored in an F-DB (F-communication DBy).

You can assign the F-communication DB numbers in the SENDS7 and RCVS7 instructions.

### 9.1.9.3 Programming safety-related communication via S7 connections

#### Introduction

The programming of safety-related CPU-CPU communication via S7 connections is described below. You must set up the following in the safety programs of the relevant F-CPUs:

- Create F-DBs (F-Communication-DBs) in which send/receive data for communication are stored.
- Call and assign parameters for instructions for communication from the "Instructions" Task Card in the safety program.

#### Requirement for programming

The S7 connections between the relevant F-CPUs must be configured in the network view in the "Connections" tab of the *hardware and network editor*.

#### Creating and Editing an F-Communication DB

F-communication DBs are F-DBs that you create and edit in the same way as other F-DBs in the project tree. You can assign the F-communication DB numbers in the SENDS7 and RCVS7 instructions.

**Note**

The length and structure of the F-communication DB at the receiver end must match the length and structure of the associated F-communication DB at the sender end.

If the F-communication DBs do not match, the F-CPU can go to STOP mode. A diagnostic event is entered in the diagnostic buffer of the F-CPU.

For this reason, we recommend that you use the following procedure:

1. Create an F-communication DB in the project tree in or below the "Program blocks" folder of the F-CPU at the sender end.
2. Specify the appropriate structure of the F-communication DB, taking into account the data to be transferred.
3. Copy this F-communication DB to the project tree in or below the "Program blocks" folder of the F-CPU at the receiver end, and change the name, if necessary.

### Other requirements for F-communication DBs

F-communication DBs must also conform to the following properties:

- They are not permitted to be instance DBs.
- Their length is not permitted to exceed 100 bytes.
- In F-communication DBs, only the following data types may be declared: BOOL, INT, WORD, DINT, DWORD, and TIME.
- The data types must be arranged block-wise and in the following order: BOOL, data types with bit length of 16 bits (INT, WORD), and data types with bit length of 32 bits (DINT, DWORD, and TIME). Within the data blocks with lengths of 16 bits and 32 bits, the data types can be arranged in any order.
- No more than 128 data elements of data type BOOL are permitted to be declared.
- The amount of data of data type BOOL must always be an integer multiple of 16 (word limit). Reserve data must be added, if necessary.

If these criteria are not fulfilled, *STEP 7 Safety Advanced* outputs an error message during compilation.

### Assignment of fail-safe values

Fail-safe values are made available at the receiver end:

- While the connection between the communication partners is being established the first time after startup of the F-systems
- Whenever a communication error occurs

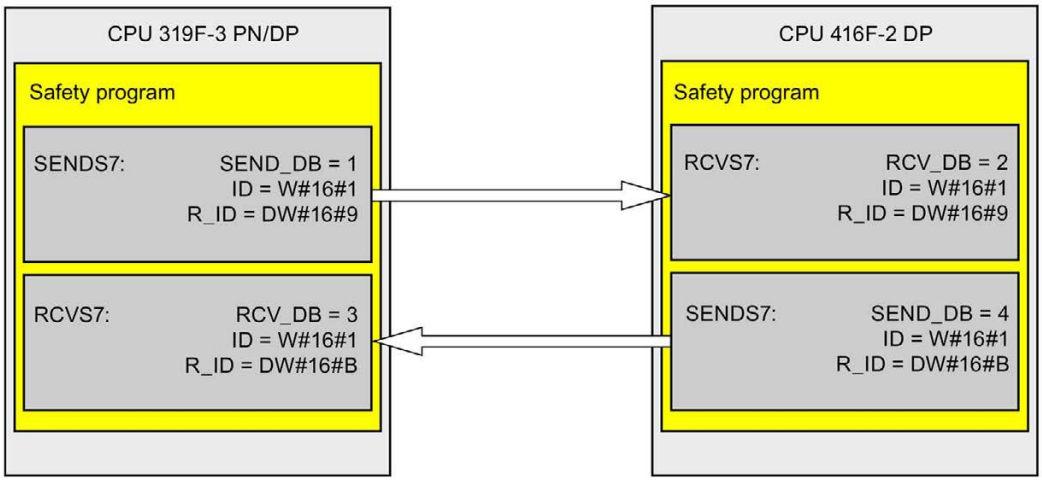
The values you specified as initial values in the F-communication DB at the receiver end are made available as initial values.

### Programming procedure

You program safety-related communication via S7 connections as follows:

1. Supply the tags in the F-communication DB at the sender end with send signals using fully qualified access (e.g., "Name of F-communication DB"."tag name").
2. Read the tags in the F-communication DB at the receiver end (receive signals) that you want to process further in other sections of the program using fully qualified access (e.g., "Name of F-communication DB"."tag name").
3. In the safety program from which data are to be sent, call the SENDS7 instruction for sending at the end of the main safety block.
4. In the safety program from which data are to be received, call the RCVS7 instruction for receiving at the start of the main safety block.
5. Assign F-communication DB numbers to the SEND_DB input of SENDS7 and the RCV_DB input of RCVS7.
6. Assign the local ID of the S7 connection (data type:WORD) from the perspective of the F-CPU that was configured in the "Connections" tab of the network view to the ID input of SENDS7 .

7. Assign the local ID of the S7 connection (data type: WORD) that was configured in the "Connections" tab of the network view to the ID input of RCVS7 .
8. Assign an odd number (data type: DWORD) to the R_ID inputs of SENDS7 and RCVS7 . This serves to specify that a SENDS7 instruction belongs to an RCVS7 instruction. The associated instructions receive the same value for R_ID.




**WARNING**

The value for each address relationship (R_ID input; data type: DWORD) is user-defined; however, it must be an odd number and unique network-wide* for all safety-related communication connections. The value R_ID + 1 is internally assigned and must not be used.

You must supply inputs ID and R_ID with constant values when calling the instruction. Direct read or write access to the associated instance DB is not permitted in the safety program! *(S020)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

9. Assign the TIMEOUT inputs of the SENDS7 and RCVS7 instructions with the required monitoring time.

**WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal level is pending for at least as long as the assigned monitoring time. *(S018)*

Information on calculating the monitoring times can be found in Monitoring and response times (Page 661).

10. To reduce the bus load, you can temporarily shut down communication between the F-CPUs at input EN_SEND of the SENDS7 instruction. To do so, supply input EN_SEND (initial value = "TRUE") with 0. In this case, send data are no longer sent to the F-communication DB of the associated RCVS7 instruction and the receiver RCVS7 provides fail-safe values for this period (initial values in its F-communication DB). If communication was already established between the partners, a communication error is detected.
11. Optional: evaluate the ACK_REQ output of RCVS7, for example, in the standard user program or on the HMI system in order to query or to indicate whether user acknowledgment is required.
12. Supply the ACK_REI input of RCVS7 with the signal for the acknowledgment for reintegration.
13. Optional: evaluate the SUBS_ON output of RCVS7 or SENDS7 in order to query whether the RCVS7 instruction is outputting the fail-safe values you specified as initial values in the F-communication DB.
14. Optional: evaluate the ERROR output of RCVS7 or SENDS7, for example, in the standard user program or on the HMI system in order to query or to indicate whether a communication error has occurred.
15. Optional: evaluate the SENDMODE output of RCVS7 in order to query whether the F-CPU with the associated SENDS7 instruction is in disabled safety mode (Page 293).

### Particularities for migrated projects

If you have migrated a project from *S7 Distributed Safety V5.4* to *STEP 7 Safety Advanced* in which safety-related communication via S7 connections is programmed, you must note the following:

- Do not delete migrated instance DBs for the SENDS7 and RCVS7 instructions in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks".

Otherwise, communication errors may occur in the relevant communication connections.

A migrated instance DB for the SENDS7 and RCVS7 instructions has been deleted if, after compiling the safety program, the "User-defined ID" in the newly generated is not identical to "FRCVS7CL" or "FSNDS7CL".

You can find the "User-defined ID" of a block in its properties in the "Information" area.

### 9.1.9.4 Safety-related communication via S7 connections - Limits of data transfer

**Note**

If the amount of data to be transmitted exceeds the permitted length for the F-communication DB (100 bytes), you can create another F-communication DB that you transfer to additional SENDS7/RCVS7 instructions with modified R_ID.

Note that USEND and URCV instructions are called internally at each SENDS7 or RCVS7 call and use connection resources in the F-CPU. This affects the maximum number of communication connections available (*see manuals for F-CPUs*).

Additional information on the data transfer limits for S7 connections of individual F-CPUs is available on the Internet (http://support.automation.siemens.com/WW/view/en/38549114).

## 9.1.10 Safety-related communication with S7 F-systems

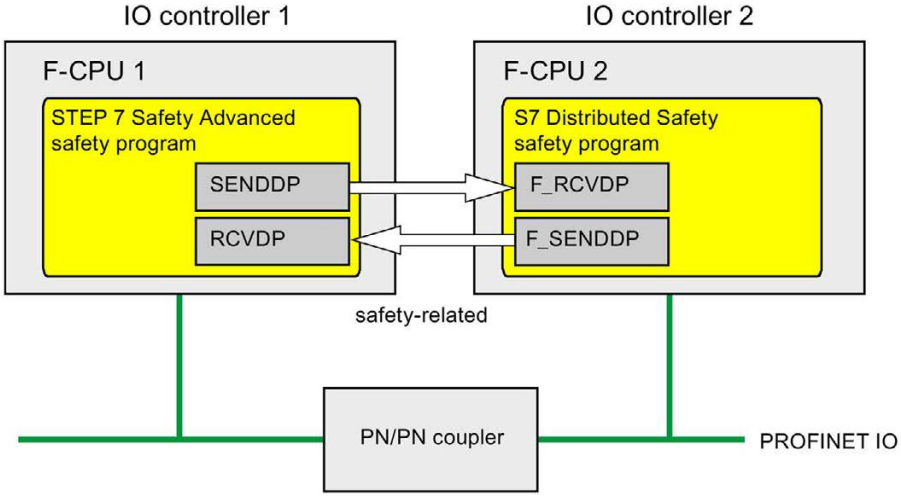
### 9.1.10.1 Introduction

Safety-related communication from F-CPUs in SIMATIC Safety to F-CPUs in S7 Distributed Safety F-systems is possible via a PN/PN coupler or DP/DP coupler that you use between the two F-CPUs as IO controller-IO controller communication, master-master communication or communication via established S7 connections.

Safety-related communication from F-CPUs in SIMATIC Safety to F-CPUs in S7 F/FH Systems F-systems is possible via established S7 connections.

### 9.1.10.2 Communication with S7 Distributed Safety via PN/PN coupler (IO controller-IO controller communication)

Communication functions between SENDDP/RCVDP instructions at the *STEP 7 Safety Advanced* end and F-application blocks F_SENDDP/F_RCVDP at the *S7 Distributed Safety* end:




#### Procedure at the *S7 Distributed Safety* end

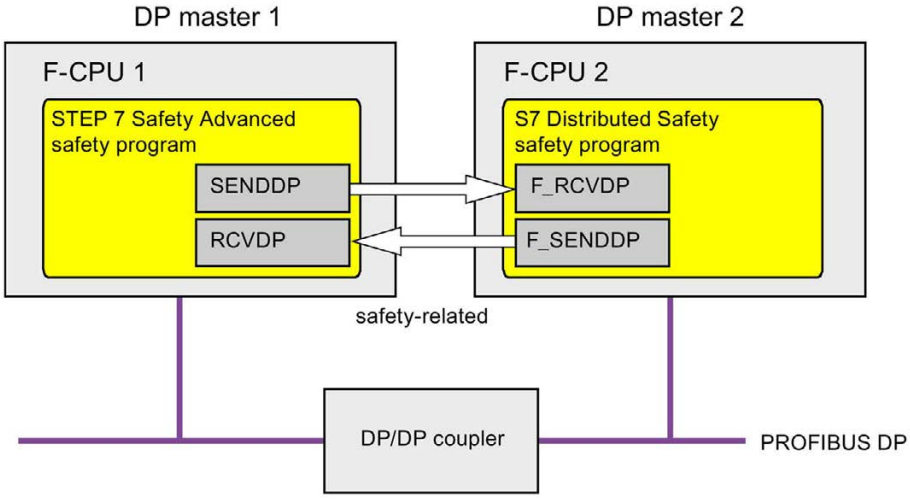
At the *S7 Distributed Safety* end, proceed as described in "Safety-related IO controller-IO controller communication" in the S7 Distributed Safety - Configuring and Programming (http://support.automation.siemens.com/WW/view/en/22099875) manual.

#### Procedure at the *STEP 7 Safety Advanced* end

At the *STEP 7 Safety Advanced* end, proceed as described in Safety-related IO controller-IO controller communication (Page 166).

### 9.1.10.3 Communication with S7 Distributed Safety via DP/DP coupler (master-master communication)

Communication functions between SENDDP/RCVDP instructions at the *STEP 7 Safety Advanced* end and F-application blocks F_SENDDP/F_RCVDP at the *S7 Distributed Safety* end:




#### Procedure at the *S7 Distributed Safety* end

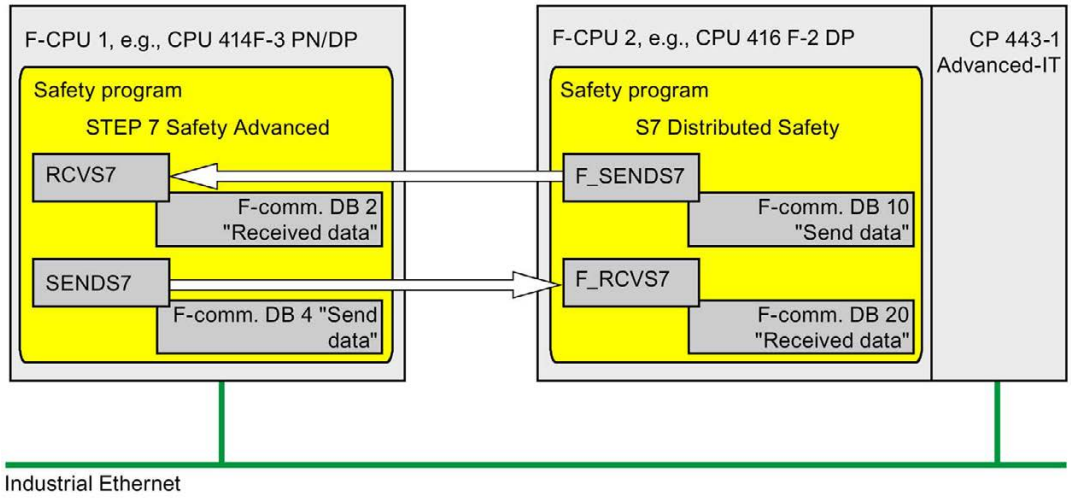
At the *S7 Distributed Safety* end, proceed as described in "Safety-related master-master communication" in the S7 Distributed Safety - Configuring and Programming (http://support.automation.siemens.com/WW/view/en/22099875) manual.

#### Procedure at the *STEP 7 Safety Advanced* end

At the *STEP 7 Safety Advanced* end, proceed as described in Safety-related master-master communication (Page 175).

### 9.1.10.4 Communication with S7 Distributed Safety via S7 connections

Communication functions between SENDS7/RCVS7 instructions at the *STEP 7 Safety Advanced* end and F_SENDS7/F_RCVS7 F-application blocks at the *S7 Distributed Safety* end:




#### Procedure at the *S7 Distributed Safety* end

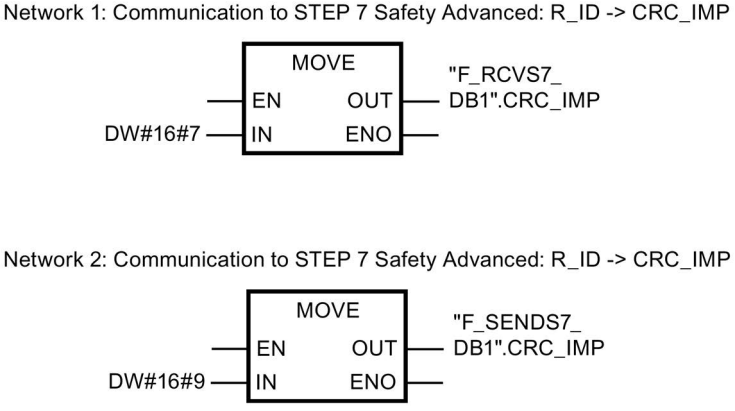
At the *S7 Distributed Safety* end, proceed as described in section "Safety-related communication via S7 communications" in the S7 Distributed Safety - Configuring and Programming (http://support.automation.siemens.com/WW/view/en/22099875) manual.

Because safety-related communication via S7 connections is not possible with unspecified partners in S7 Distributed Safety, you must first create a "virtual" SIMATIC station in *S7 Distributed Safety* in which you configure an F-CPU as a proxy for the F-CPU in *STEP 7 Safety Advanced* with its IP address.

You then insert an S7 connection to this F-CPU in the connection table. Both the local connection and partner connection resources (hex) are thereby fixed. You must then set these in the associated, unspecified S7 connection that you created in *STEP 7 Professional*.

In addition, for all communication connections to this F-CPU, you must transfer the address relationship that you assigned in the R_ID input of the associated calls of the F_SENDS7 and F_RCVS7 F-application blocks additionally to the CRC_IMP tag in the instance DB of F_SENDS7 and F_RCVS7, respectively, in the standard user program immediately before calling the F-CALL.

Program example:




## Procedure at the *STEP 7 Safety Advanced* end

At the *STEP 7 Safety Advanced* end, proceed as described in Safety-related communication via S7 connections (Page 212).

For the F-CPU in *S7 Distributed Safety*, you must create and specify an unspecified S7 connection (links to standard online help: Creating an unspecified connection and Specifying and unspecified connection).

For these you must set the local and partner connection resources (hex) that are fixed as a result of the associated S7 connection that you have created in *S7 Distributed Safety*.

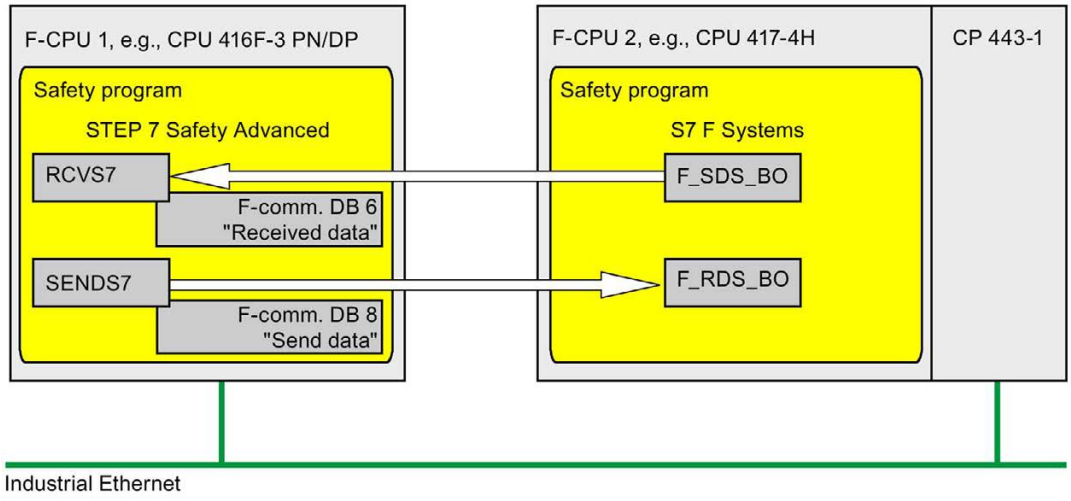
If the local connection resource (hex) is already occupied by an existing connection, you must change the connection resource (hex) for it.

If the instance DBs of the SENDS7 and RCVS7, instructions that you want to use for communication with *S7 Distributed Safety* were migrated from *S7 Distributed Safety*, you must delete them in the project tree in the "STEP 7 Safety" folder, under "Program blocks > System blocks" (contrary to the information in Programming safety-related communication via S7 connections (Page 215), section "Particularities for migrated projects").

### 9.1.10.5 Communication with S7 F/FH Systems via S7 connections

Communication functions between SENDS7/RCVS7 instructions at the *STEP 7 Safety Advanced* end and F_SDS_BO/F_RDS_BO F-blocks at the *S7 F Systems* end.

A maximum of 32 data elements of data type BOOL can be exchanged.




## Procedure at the *S7 F Systems* end

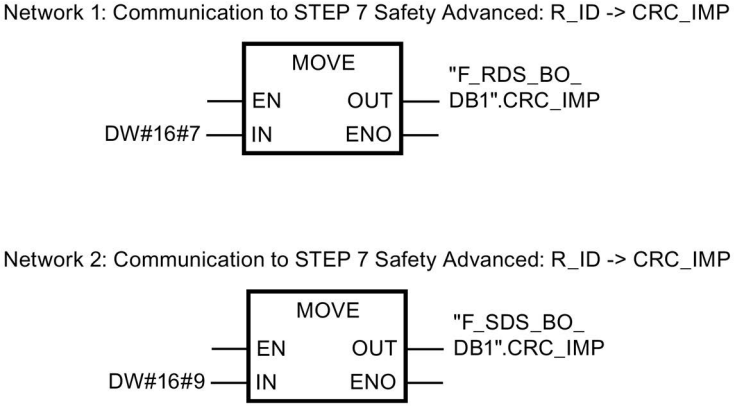
At the *S7 F-systems* end, proceed as described in section "Safety-related communication between F-CPUs" in the S7 F/FH Systems - Configuring and Programming (http://support.automation.siemens.com/WW/view/en/16537972) manual.

Because safety-related communication via S7 connections is not possible with unspecified partners in *S7 F/FH Systems*, you must first create a "virtual" SIMATIC station in *S7 F/FH Systems* in which you configure an F-CPU as a proxy for the F-CPU in *STEP 7 Safety Advanced* with its IP address.

You then insert an S7 connection to this F-CPU in the connection table. Both the local connection and partner connection resources (hex) are thereby fixed. You must then set these in the associated, unspecified S7 connection that you created in *STEP 7 Safety Advanced*.

In addition, you must insert a function in your S7 program (in the area reserved in CFC for other applications), in which, for all communication connections for this F-CPU, you transfer the address relationship that you assigned in the R_ID input of the associated calls of the F_SDS_BO and F_RDS_BO F-blocks additionally to the CRC_IMP tag in the instance DB of the F_SDS_BO and F_RDS_BO, respectively. You obtain the number of the instance DB from the object properties of the block in CFC. Assign descriptive symbolic names for these instance DBs. If you perform a compress operation in CFC, you must check whether the numbers of these instance DBs have changed.

Program example:




You must then import the function in CFC as block type and insert your standard user program in a chart. In the run sequence, make sure that the associated standard runtime group is processed before the F-runtime group.

## Procedure at the *STEP 7 Safety Advanced* end

At the *STEP 7 Safety Advanced* end, proceed as described in "Safety-related communication via S7 connections" (Page 212).

**Particularity:** In *STEP 7 Safety Advanced*, you must create the F-communication DB with exactly 32 data elements of data type BOOL.

For the F-CPU in *S7 F/FH Systems*, you must create and specify an unspecified S7 connection (see online help for *STEP 7* under "Creating an unspecified connection" and "Specifying an unspecified connection").

For these you must set the local and partner connection resources (hex) that are fixed as a result of the associated S7 connection that you have created in *S7 F Systems*.

If the local connection resource (hex) is already occupied by an existing connection, you must change the connection resource (hex) for it.

# 9.2 Configuring and programming communication (S7-1500)

## 9.2.1 Overview of communication

### Introduction

This section gives an overview of the safety-related communication options in SIMATIC Safety F-systems.

### Options for safety-related communication

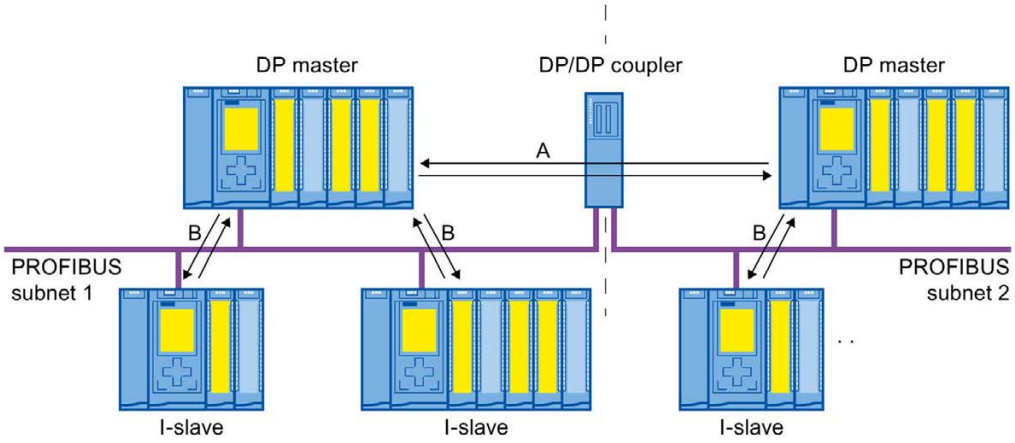
| Safety-related communication | On subnet | Additional hardware required |
|---|---|---|
| **Safety-related CPU-CPU communication:** | | |
| IO controller-IO controller communication | PROFINET IO | PN/PN coupler |
| Master-master communication | PROFIBUS DP | DP/DP coupler |
| IO controller-I-device communication | PROFINET IO | — |
| Master-I-slave communication | PROFIBUS DP | — |
| IO controller-I-slave communication | PROFINET IO and PROFIBUS DP | IE/PB link |
| IO controller-IO controller communication for *S7 Distributed Safety* | PROFINET IO | PN/PN coupler |
| Master-master communication for *S7 Distributed Safety* | PROFIBUS DP | DP/DP coupler |

**Note**

"Shared Device" functionality is not supported for fail-safe I/O.

## Overview of safety-related communication via PROFIBUS DP

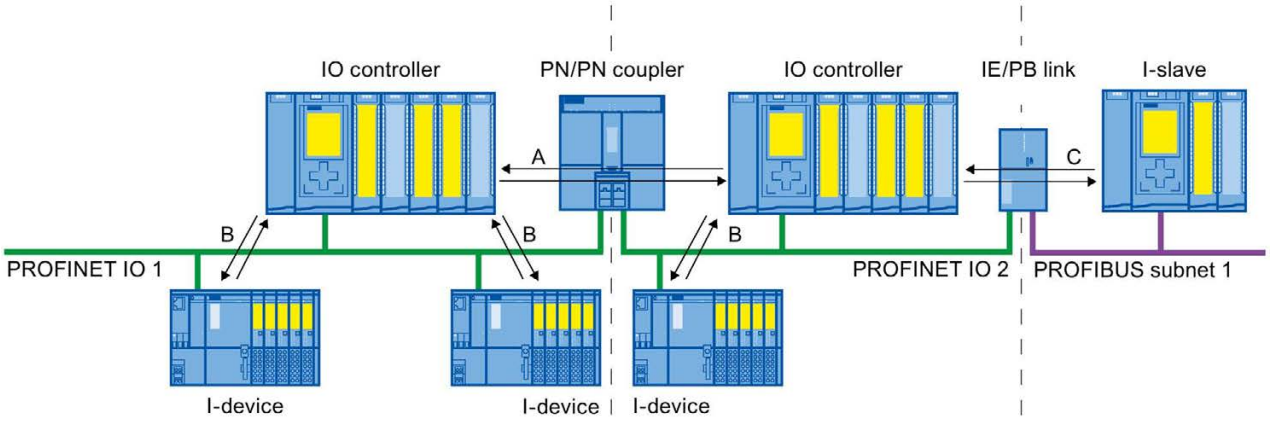
The figure below provides an overview of the options for safety-related communication via PROFIBUS DP in SIMATIC Safety F-systems with S7-1500 F-CPUs.




A safety-related master-master communication
B safety-related master-I-slave communication

## Overview of safety-related communication via PROFINET IO

The figure below provides an overview of the options for safety-related communication via PROFINET IO in SIMATIC Safety F-systems with S7-1500 F-CPUs.




A safety-related IO controller-IO controller communication
B safety-related IO controller-I-device communication
C safety-related IO controller-I-slave communication

### Safety-related CPU-CPU communication via PROFIBUS DP or PROFINET IO

In safety-related CPU-CPU communication, a fixed amount of data of the data type BOOL or INT (DINT as alternative) is transmitted fail-safe between the safety programs in F-CPUs of DP masters/I-slaves or IO controllers/I-devices.

The data are transferred using the SENDDP instruction for sending and the RCVDP instruction for receiving. The data are stored in configured transfer areas of the devices. The hardware identifier (HW identifier) defines the transfer areas configured.

### Safety-related CPU-CPU communication for *S7 Distributed Safety*

Safety-related communication is possible from F-CPUs in *SIMATIC Safety* to F-CPUs in *S7 Distributed Safety*.

## 9.2.2 Safety-related IO controller-IO controller communication

### 9.2.2.1 Configure safety-related IO controller-IO controller communication

#### Introduction

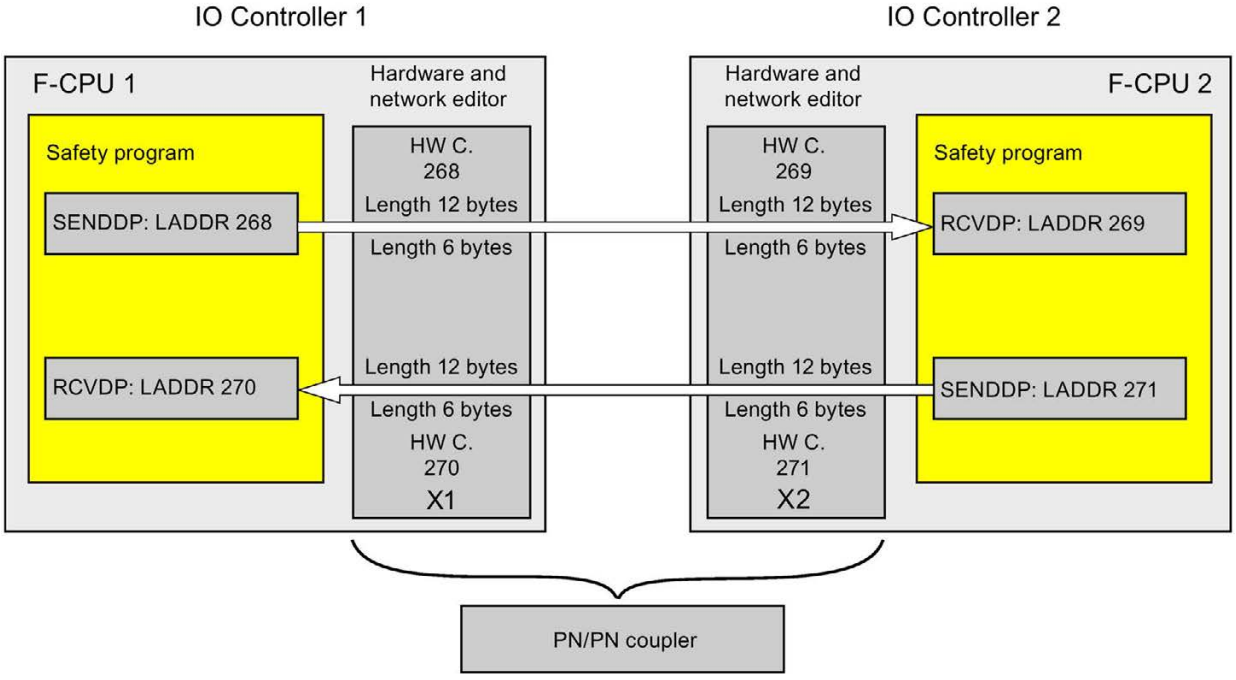
Safety-related communication between safety programs of the F-CPUs of IO controllers takes place over a PN/PN coupler that you set up between the F-CPUs.

**Note**

Deactivate the "Data validity display DIA" parameter in the properties for the PN/PN coupler in the *hardware and network editor*. This is the default setting. Otherwise, safety-related IO controller-IO controller communication is not possible.

#### Configuring transfer areas

You must configure one transfer area for output data and one transfer area for input data in the *hardware and network editor* for each safety-related communication connection between two F-CPUs in the PN/PN coupler. The figure below shows how both of the F-CPUs are able to send **and** receive data (bidirectional communication).



## Rules for defining transfer areas

**Data to be sent**:

A total of 12 bytes (consistent) is required for the transfer area for output data; 6 bytes (consistent) are required for the transfer area for input data.

**Data to be received**:

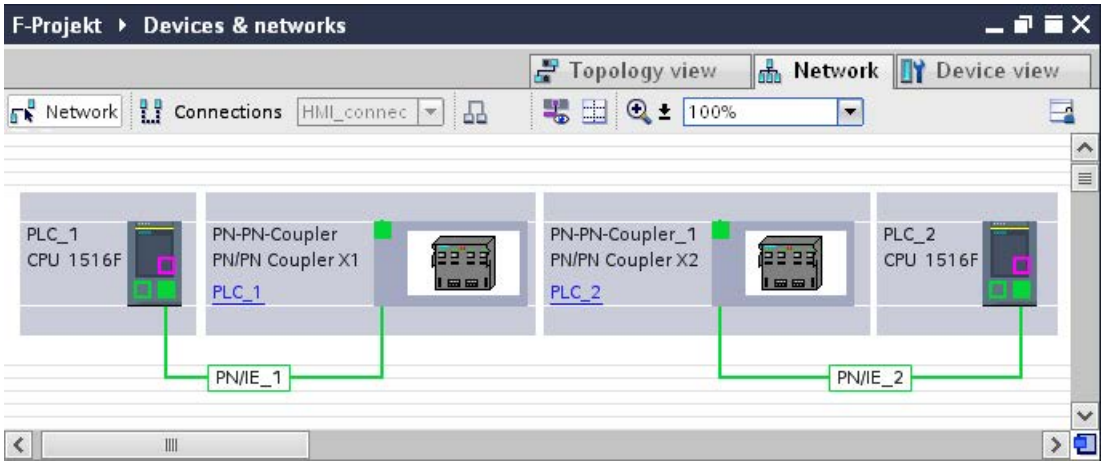
A total of 12 bytes (consistent) is required for the transfer area for input data; 6 bytes (consistent) are required for the transfer area for output data.

## Procedure for configuration

The procedure for configuring safety-related IO controller-IO controller communication is identical to that in the standard system.

Proceed as follows:

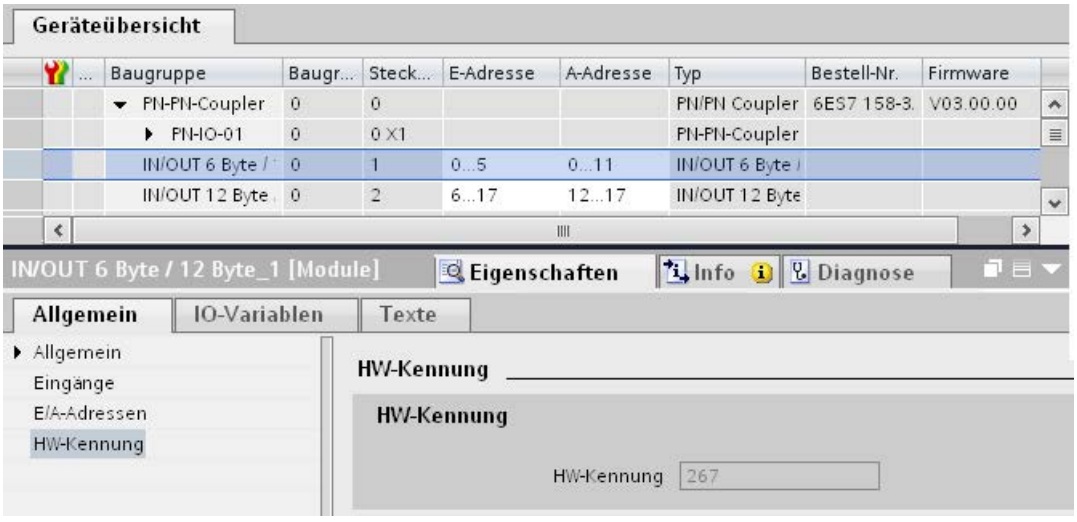
1. Insert two F-CPUS from the "Hardware catalog" task card into the project.
2. Switch to the network view of the *hardware and network editor*.
3. Insert a PN/PN coupler X1 and a PN/PN coupler X2 from "Other field devices\PROFINET IO\Gateway\Siemens AG\PN/PN Coupler" in the "Hardware catalog" task card.
4. Connect the PN interface of the F-CPU 1 with the PN interface of the PN/PN Coupler X1 and the PN interface of the F-CPU 2 with the PN interface of PN/PN Coupler X2.



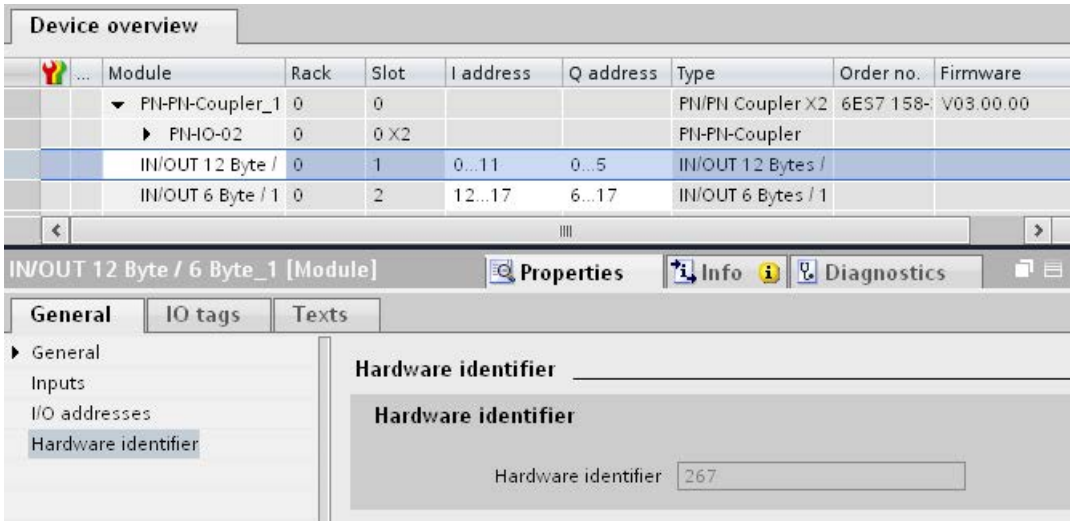
5. Switch to the device view of PN/PN coupler X1 for bidirectional communication connections i.e. where each F-CPU is both to send and to receive data. Select the following modules from "IN/OUT" in the "Hardware catalog" task card (with filter activated), and insert them in the "Device overview" tab:
   - One "IN/OUT 6 bytes / 12 bytes" module and
   - One "IN/OUT 12 bytes / 6 bytes" module

**Note**

The transfer areas are assigned on the basis of the hardware identifier which is automatically assigned to the modules and devices. You need the HW identifier to program the SENDDP and RCVDP blocks (LADDR input). A system constant is created in the corresponding F-CPU for each hardware identifier of the transfer area. You can assign these system constants symbolically to the SENDDP and RCVDP blocks.

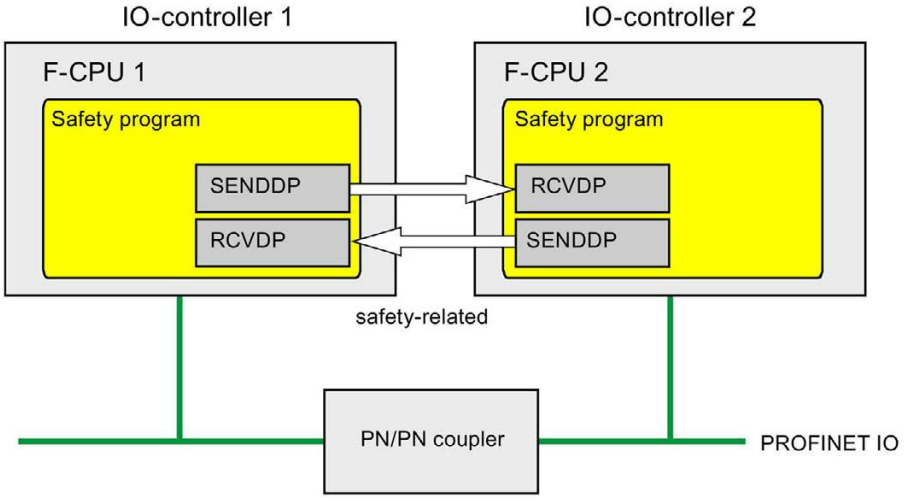


6. Select the following modules from "IN/OUT" in the device view of PN/PN coupler X2 and insert them in the "Device overview" tab:
   - One "IN/OUT 12 bytes / 6 bytes" module and
   - One "IN/OUT 6 bytes / 12 bytes" module



### 9.2.2.2 Safety-related IO controller-IO controller communication via SENDDP and RCVDP

#### Communication via SENDDP and RCVDP instructions




Safety-related communication between the F-CPUs of the IO controllers uses the SENDDP and RCVDP instructions for sending and receiving respectively. These can be used to perform a fail-safe transfer of a *fixed* amount of fail-safe data of the data type BOOL or INT (DINT as alternative).

You can find these instructions in the "Instructions" task card under "Communication". The RCVDP instruction **must** be called at the start of the main safety block. The SENDDP instruction **must** be called at the end of the main safety block.

Note that the send signals are not sent until after the SENDDP instruction call at the end of execution of the relevant F-runtime group.

A detailed description of the SENDDP and RCVDP instructions can be found in SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500) (Page 646).

### 9.2.2.3 Program safety-related IO controller-IO controller communication

#### Requirement for programming

The transfer areas for input and output data for the PN/PN coupler must be configured.

#### Programming procedure

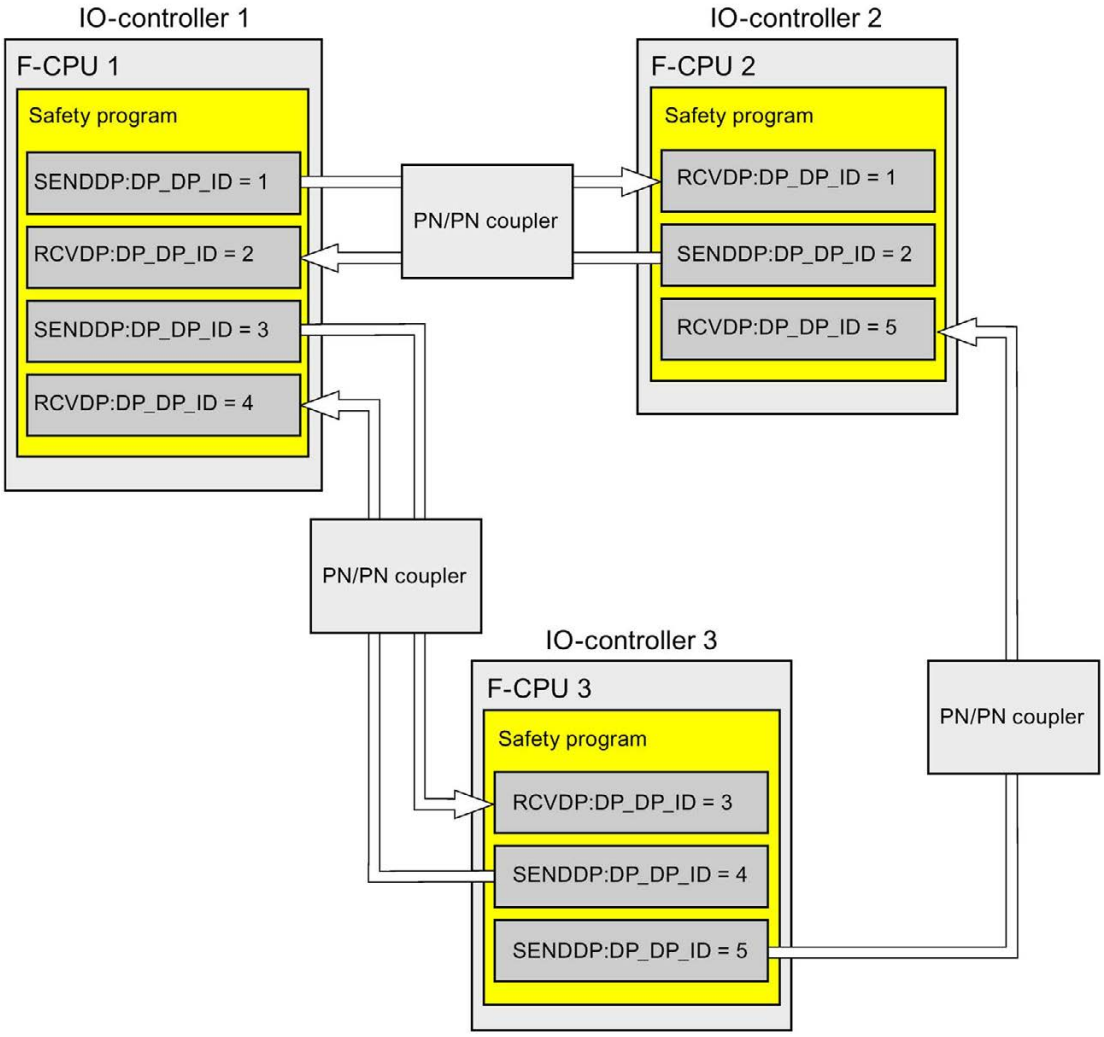
You program safety-related IO controller-IO controller communication as follows:

1. In the safety program from which data are to be sent, call the SENDDP instruction (Page 646) for sending at the end of the main safety block.
2. In the safety program in which data are to be received, call the RCVDP instruction (Page 646) for receiving at the start of the main safety block.
3. Assign the respective LADDR inputs HW identifiers (system constants in the default tag table) for the transfer areas for output and input data of the PN/PN coupler that are configured in the hardware and network editor.

   You must carry out this assignment for every communication connection for each of the F-CPUs involved.

4. Assign the value for the respective address relationship to the DP_DP_ID inputs. This establishes the communication relationship between the SENDDP instruction in one F-CPU and the RCVDP instruction in the other F-CPU: The associated instructions receive the same value for DP_DP_ID.

   The figure below contains an example of how to specify the address relationships for the inputs of the SENDDP and RCVDP instructions for five safety-related IO controller-IO-controller communication relationships.




**WARNING**

The value for each address relationship (DP_DP_ID input; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

5. Supply the SD_BO_xx and SD_I_xx inputs (SD_DI_00 as alternative) of SENDDP with the send signals. To cut down on intermediate signals when transferring parameters, you can write the value directly to the instance DB of SENDDP using fully qualified access (for example, "Name SENDDP_1".SD_BO_02) before calling SENDDP.
6. Supply the RD_BO_xx and RD_I_xx outputs (RD_DI_00 as alternative) of RCVDP with the signals that you want to process further in other program sections or use fully qualified access to read the received signals directly in the associated instance DB in the program sections to be processed further (e.g., "Name RCVDP_1".RD_BO_02).
7. If you want to send the data at the SD_DI_00 input instead of the data at the SD_I_00 and SD_I_01 inputs, supply the DINTMODE input (initial value = "FALSE") of SENDDP with TRUE.
8. Supply the SUBBO_xx and SUBI_xx or alternatively SUBDI_00 inputs of RCVDP with the fail-safe values that are to be output by RCVDP in place of the process data until communication is established for the first time after startup of the sending and receiving F-systems or in the event of an error in safety-related communication.
   - Specification of constant fail-safe values:

     For data of data type INT/DINT, you can enter constant fail-safe values directly as constants in the SUBI_xx or alternatively SUBDI_00 input (initial value = "0"). If you want to specify a constant fail-safe value "TRUE" for data of the data type BOOL, set TRUE for the SUBBO_xx input (initial value = "FALSE").
   - Specification of dynamic fail-safe values:

     If you want to specify dynamic fail-safe values, define a tag that you change dynamically through your safety program in an F-DB and specify this tag (fully qualified) in the SUBBO_xx or SUBI_xx or alternatively SUBDI_00 input.

     **⚠ WARNING**

     Note that your safety program for dynamically changing the tag for a dynamic fail-safe value can only be processed after the call of the RCVDP, because prior to the RCVDP call, there must not be any network and no more than one other RCVDP instruction in the main safety block. You must therefore assign appropriate start values for these tags to be output by RCVDP in the first cycle after a startup of the F-system. *(S017)*

9. Configure the TIMEOUT inputs of the RCVDP and SENDDP instructions with the required monitoring time.

**⚠ WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal level is pending for at least as long as the assigned monitoring time. *(S018)*

Information on calculating the monitoring times can be found in Monitoring and response times (Page 661).

10. Optional: Evaluate the ACK_REQ output of the RCVDP instruction, for example, in the standard user program or on the HMI system in order to query or to indicate whether user acknowledgment is required.
11. Supply the ACK_REI input of the RCVDP instruction with the acknowledgment signal for reintegration.
12. Optional: Evaluate the SUBS_ON output of the RCVDP or SENDDP instruction in order to query whether the RCVDP instruction is outputting the fail-safe values assigned in the SUBBO_xx and SUBI_xx or alternatively SUBDI_00 inputs.
13. Optional: Evaluate the ERROR output of the RCVDP or SENDDP instruction, for example, in the standard user program or on the HMI system in order to query or to indicate whether a communication error has occurred.
14. Optional: Evaluate the SENDMODE output of the RCVDP instruction in order to query whether the F-CPU with the associated SENDDP instruction is in disabled safety mode (Page 293).

#### 9.2.2.4 Safety-related IO controller-IO controller communication - Limits for data transfer

**Note**

If the data quantities to be transmitted exceed the capacity of the SENDDP / RCVDP correlated instructions, a second (or third) SENDDP / RCVDP call can be used. This requires configuration of an additional connection via the PN/PN coupler. Whether or not this is possible with one single PN/PN coupler depends on the capacity restrictions of the PN/PN coupler.

## 9.2.3 Safety-related master-master communication

### 9.2.3.1 Configure safety-related master-master communication

#### Introduction

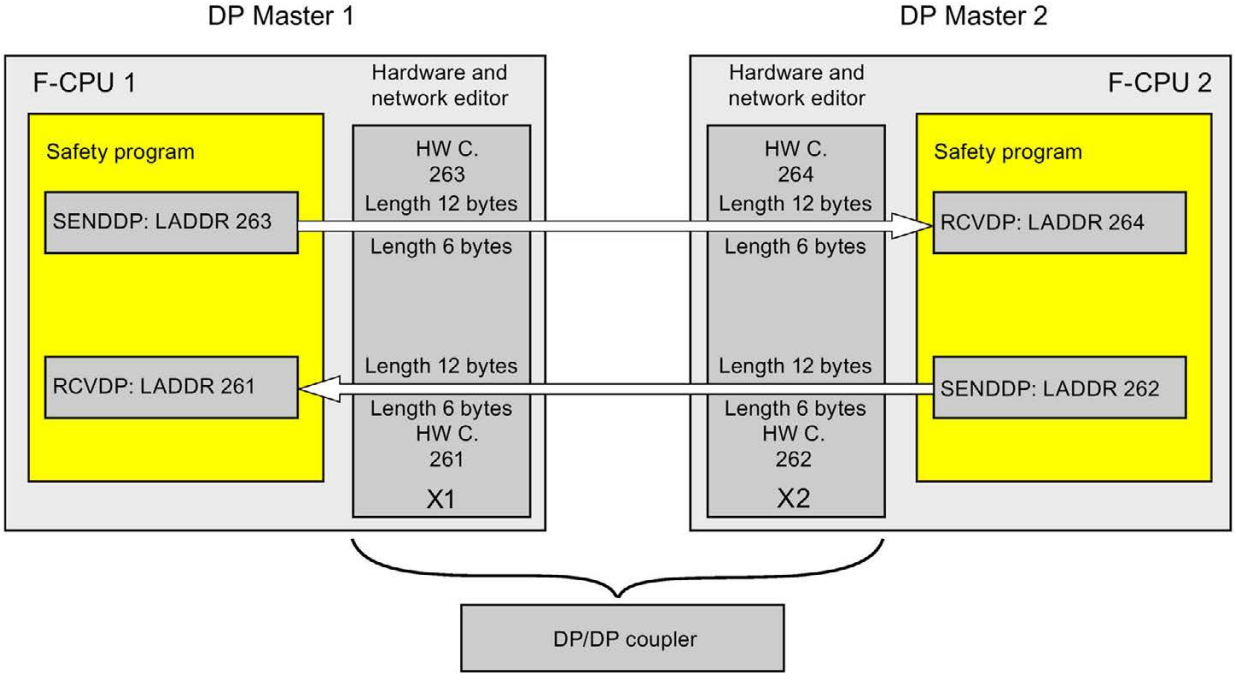
Safety-related communication between safety programs of the F-CPUs of DP masters takes place via a DP/DP coupler.

**Note**

Switch the data validity indicator "DIA" on the DIP switch of the DP/DP coupler to "OFF". Otherwise, safety-related CPU-CPU communication is not possible.

#### Configuring transfer areas

You must configure one transfer area for output data and one transfer area for input data in the *hardware and network editor* for each safety-related communication connection between two F-CPUs in the DP/DP coupler. The figure below shows how both of the F-CPUs are able to send and receive data (bidirectional communication).



## Rules for defining transfer areas

**Data to be sent**:

A total of 12 bytes (consistent) is required for the transfer area for output data; 6 bytes (consistent) are required for the transfer area for input data.

**Data to be received**:

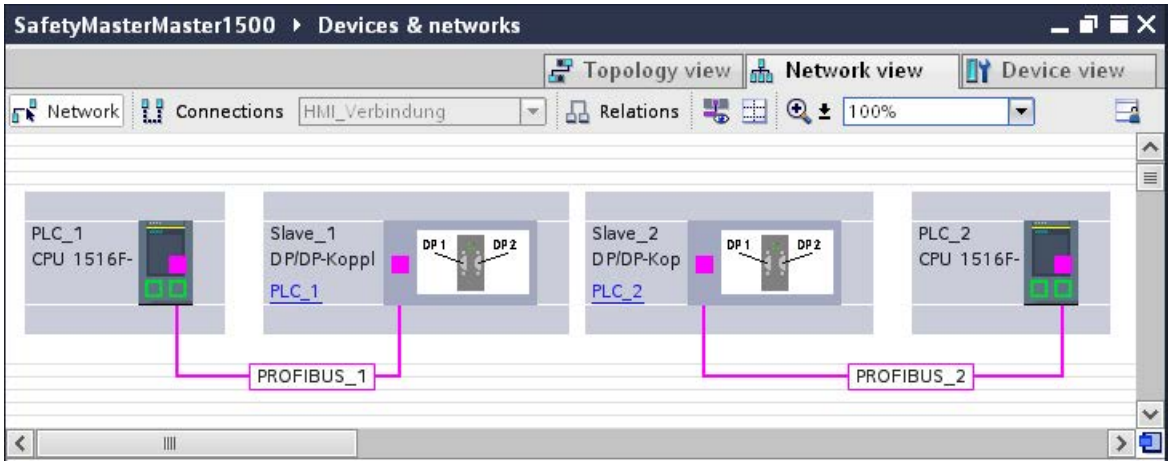
A total of 12 bytes (consistent) is required for the transfer area for input data; 6 bytes (consistent) are required for the transfer area for output data.

## Procedure for configuration

The procedure for configuring safety-related master-master communication is identical to that in the standard system.

Proceed as follows:

1. Insert two F-CPUS from the "Hardware catalog" task card into the project.
2. Switch to the network view of the *hardware and network editor*.
3. Select a DP/DP coupler from "Other field devices\PROFIBUS DP\Gateways\Siemens AG\DP/DP Coupler" in the "Hardware catalog" task card and insert it into the network view of the hardware and network editor.
4. Insert a second DP/DP coupler.
5. Connect a DP interface of F-CPU 1 to the DP interface of a DP/DP coupler and a DP interface of F-CPU 2 to the DP interface of the other DP/DP coupler.

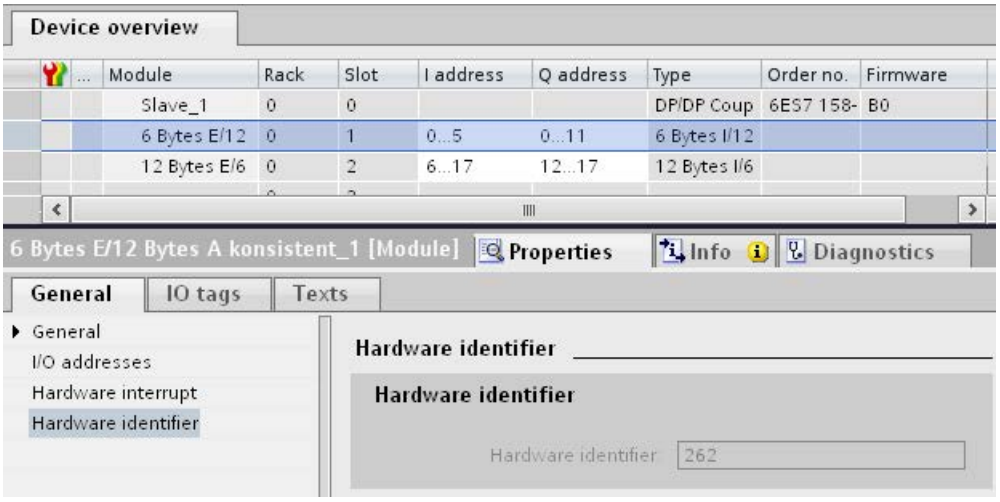



6. A free PROFIBUS address is assigned automatically in the properties of the DP/DP-coupler in the device view. You must set this address on the DP/DP coupler, either via the DIP switch on the device or in the configuration of the DP/DP coupler (see DP/DP Coupler (http://support.automation.siemens.com/WW/view/en/1179382) manual).

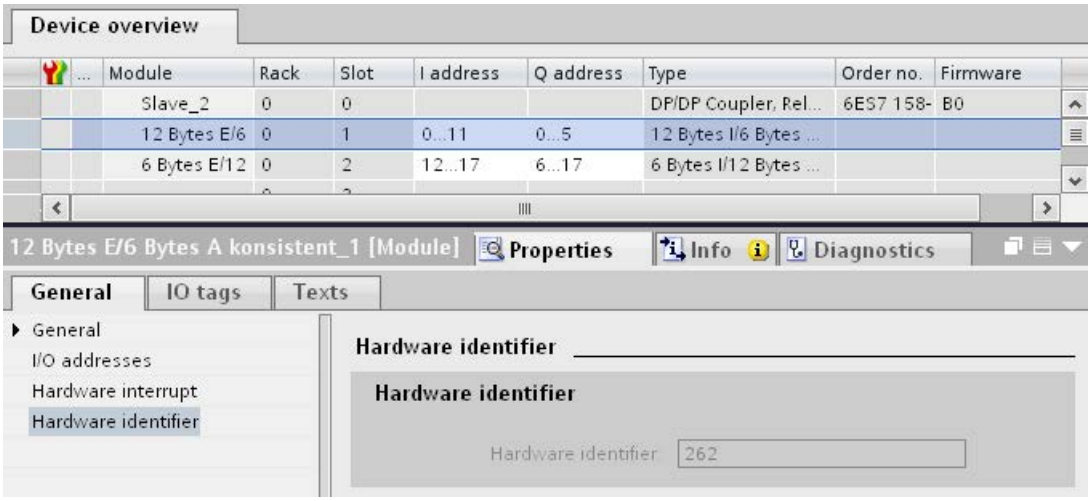
7. Switch to the device view of the DP/DP coupler PLC1 for bidirectional communication connections i.e. where each F-CPU should both send and receive data. Select the following modules from the "Hardware catalog" task card (with filter activated), and insert them in the "Device overview" tab of the DP/DP coupler:
   - One "6 bytes I/12 bytes Q consistent" module, and
   - One "12 bytes I/6 bytes Q consistent" module

**Note**

The transfer areas are assigned on the basis of the hardware identifier which is automatically assigned to the modules and devices. You need the HW identifier to program the SENDDP and RCVDP blocks (LADDR input). A system constant is created in the corresponding F-CPU for each hardware identifier of the transfer area. You can assign these system constants symbolically to the SENDDP and RCVDP blocks.

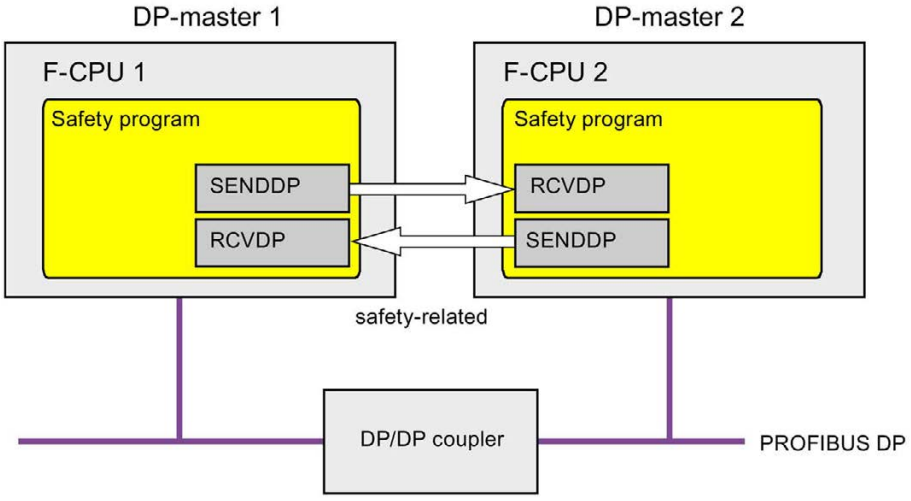


8. Select the following modules from the "Hardware catalog" task card (with filter activated) in the device view of DP/DP coupler PLC2, and insert them in the "Device overview" tab:
   - One "12 bytes I/6 bytes Q consistent" module, and
   - One "6 bytes I/12 bytes Q consistent" module



### 9.2.3.2 Safety-related master-master communication via SENDDP and RCVDP

#### Communication via SENDDP and RCVDP instructions




Safety-related communication between the F-CPUs of the DP masters uses the SENDDP and RCVDP instructions for sending and receiving respectively. These can be used to perform a fail-safe transfer of a *fixed* amount of fail-safe data of the data type BOOL or INT (DINT as alternative).

You can find these instructions in the "Instructions" task card under "Communication". The RCVDP instruction **must** be called at the start of the main safety block. The SENDDP instruction **must** be called at the end of the main safety block.

Note that the send signals are not sent until after the SENDDP instruction call at the end of execution of the relevant F-runtime group.

A detailed description of the SENDDP and RCVDP instructions can be found in SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500) (Page 646).

### 9.2.3.3 Program safety-related master-master communication

#### Requirement for programming

The address areas for input and output data for the DP/DP coupler must be configured.

#### Programming procedure

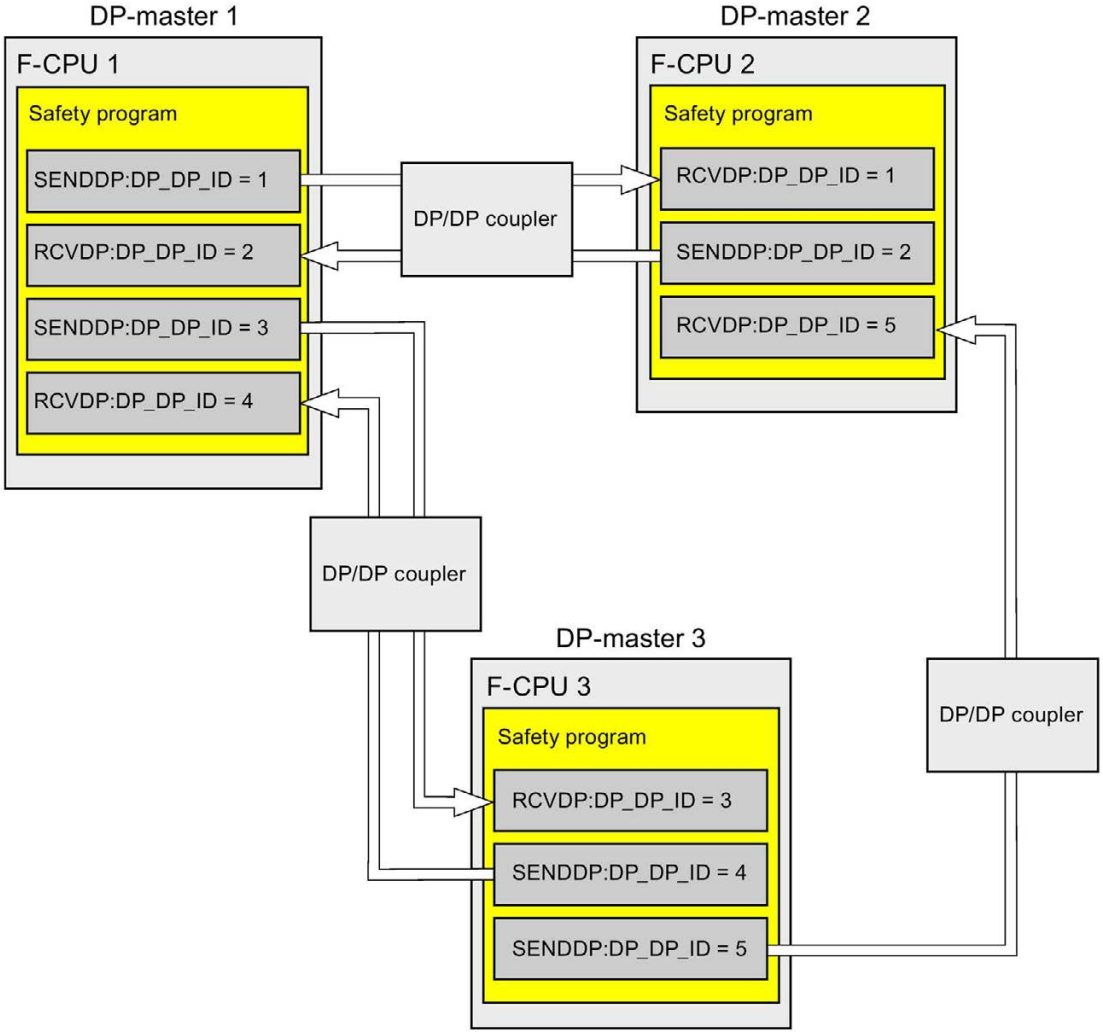
You program safety-related master-master communication as follows:

1. In the safety program from which data are to be sent, call the SENDDP instruction (Page 646) for sending at the end of the main safety block.
2. In the safety program from which data are to be received, call the RCVDP instruction (Page 646) for receiving at the start of the main safety block.
3. Assign the HW identifiers for the output and input data of the DP/DP coupler configured in the *hardware and network editor* (constant in the tag table) to the respective LADDR inputs.

   You must carry out this assignment for every communication connection for each of the F-CPUs involved.

4. Assign the value for the respective address relationship to the DP_DP_ID inputs. This establishes the communication relationship between the SENDDP instruction in one F-CPU and the RCVDP instruction in the other F-CPU: The associated instructions receive the same value for DP_DP_ID.

   The figure below contains an example of how to specify the address relationships at the inputs of SENDDP and RCVDP instructions for five safety-related master-master communications relationships.




**WARNING**

The value for each address relationship (DP_DP_ID input; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

5. Supply the SD_BO_xx and SD_I_xx inputs (SD_DI_00 as alternative) of SENDDP with the send signals. To cut down on intermediate signals when transferring parameters, you can write the value directly to the instance DB of SENDDP using fully qualified access (for example, "Name SENDDP_1".SD_BO_02) before calling SENDDP.
6. Supply the RD_BO_xx and RD_I_xx outputs (RD_DI_00 as alternative) of RCVDP with the signals that you want to process further in other program sections or use fully qualified access to read the received signals directly in the associated instance DB in the program sections to be processed further (e.g., "Name RCVDP_1".RD_BO_02).
7. If you want to send the data at the SD_DI_00 input instead of the data at the SD_I_00 and SD_I_01 inputs, supply the DINTMODE input (initial value = "FALSE") of SENDDP with TRUE.
8. Supply the SUBBO_xx and SUBI_xx or alternatively SUBDI_00 inputs of RCVDP with the fail-safe values that are to be output by RCVDP in place of the process data until communication is established for the first time after startup of the sending and receiving F-systems or in the event of an error in safety-related communication.
   - Specification of constant fail-safe values:

     For data of data type INT/DINT, you can enter constant fail-safe values directly as constants in the SUBI_xx or alternatively SUBDI_00 input (initial value = "0"). If you want to specify a constant fail-safe value "TRUE" for data of the data type BOOL, set TRUE for the SUBBO_xx input (initial value = "FALSE").
   - Specification of dynamic fail-safe values:

     If you want to specify dynamic fail-safe values, define a tag that you change dynamically through your safety program in an F-DB and specify this tag (fully qualified) in the SUBBO_xx or SUBI_xx or alternatively SUBDI_00 input.

     **⚠ WARNING**

     Note that your safety program for dynamically changing the tag for a dynamic fail-safe value can only be processed after the call of the RCVDP, because prior to the RCVDP call, there must not be any network and no more than one other RCVDP instruction in the main safety block. You must therefore assign appropriate start values for these tags to be output by RCVDP in the first cycle after a startup of the F-system. *(S017)*

9. Configure the TIMEOUT inputs of the RCVDP and SENDDP instructions with the required monitoring time.

**⚠ WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal level is pending for at least as long as the assigned monitoring time. *(S018)*

Information on calculating the monitoring times can be found in Monitoring and response times (Page 661).

10. Optional: Evaluate the ACK_REQ output of the RCVDP instruction, for example, in the standard user program or on the HMI system in order to query or to indicate whether user acknowledgment is required.
11. Supply the ACK_REI input of the RCVDP instruction with the acknowledgment signal for reintegration.
12. Optional: Evaluate the SUBS_ON output of the RCVDP or SENDDP instruction in order to query whether the RCVDP instruction is outputting the fail-safe values assigned in the SUBBO_xx and SUBI_xx or alternatively SUBDI_00 inputs.
13. Optional: Evaluate the ERROR output of the RCVDP or SENDDP instruction, for example, in the standard user program or on the HMI system in order to query or to indicate whether a communication error has occurred.
14. Optional: Evaluate the SENDMODE output of the RCVDP instruction in order to query whether the F-CPU with the associated SENDDP instruction is in disabled safety mode (Page 293).

#### 9.2.3.4 Safety-related master-master communication:Limits for data transfer

**Note**

If the data quantities to be transmitted exceed the capacity of the SENDDP / RCVDP correlated instructions, a second (or third) SENDDP / RCVDP call can be used. This requires configuration of an additional connection via the DP/DP coupler. Whether or not this is possible with one single DP/DP coupler depends on the capacity restrictions of the DP/DP coupler.

## 9.2.4 Safety-related IO controller-I-device communication

### 9.2.4.1 Configuring safety-related communication between IO controller and I-device

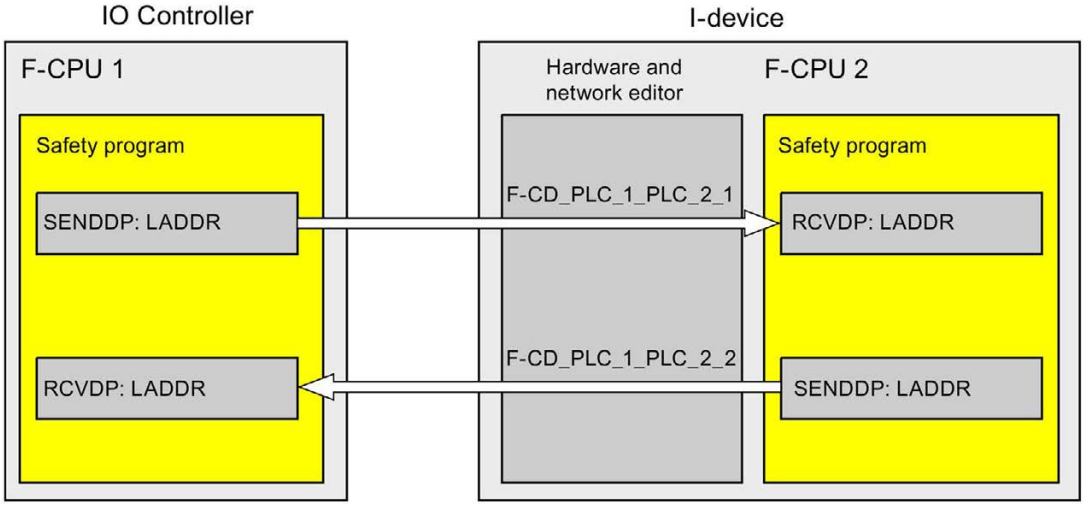
#### Introduction

Safety-related communication between the safety program of the F-CPU of an IO controller and the safety program(s) of the F-CPU(s) of one or more I-devices takes place via IO controller-I-device connections (F-CD) in PROFINET IO, in the same way as in standard systems.

You do not need any additional hardware for IO controller-I-device communication.

#### Configuring transfer areas

For every safety-related communication connection between two F-CPUs, you must configure transfer areas in the *hardware and network editor*. The figure below shows how both of the F-CPUs are able to send and receive data (bidirectional communication).




The transfer area is assigned a label when it is created to identify the communication relationship, for example "F-CD_PLC_2 PLC_1_1" for the first F-CD connection between IO-controller F-CPU 1 and I-device F-CPU 2.

When you create a transfer area, a system constant with the name of the transfer area is created in the F-CPU of the IO controller and in the F-CPU of the I-device. The system constant contains the hardware identifier of the transfer area for the corresponding F-CPU.

You assign the hardware identifier (system constant from the default tag table) of the transfer areas symbolically to the LADDR parameter of the SENDDP and RCVDP instructions in the safety programs.

### Procedure for configuration

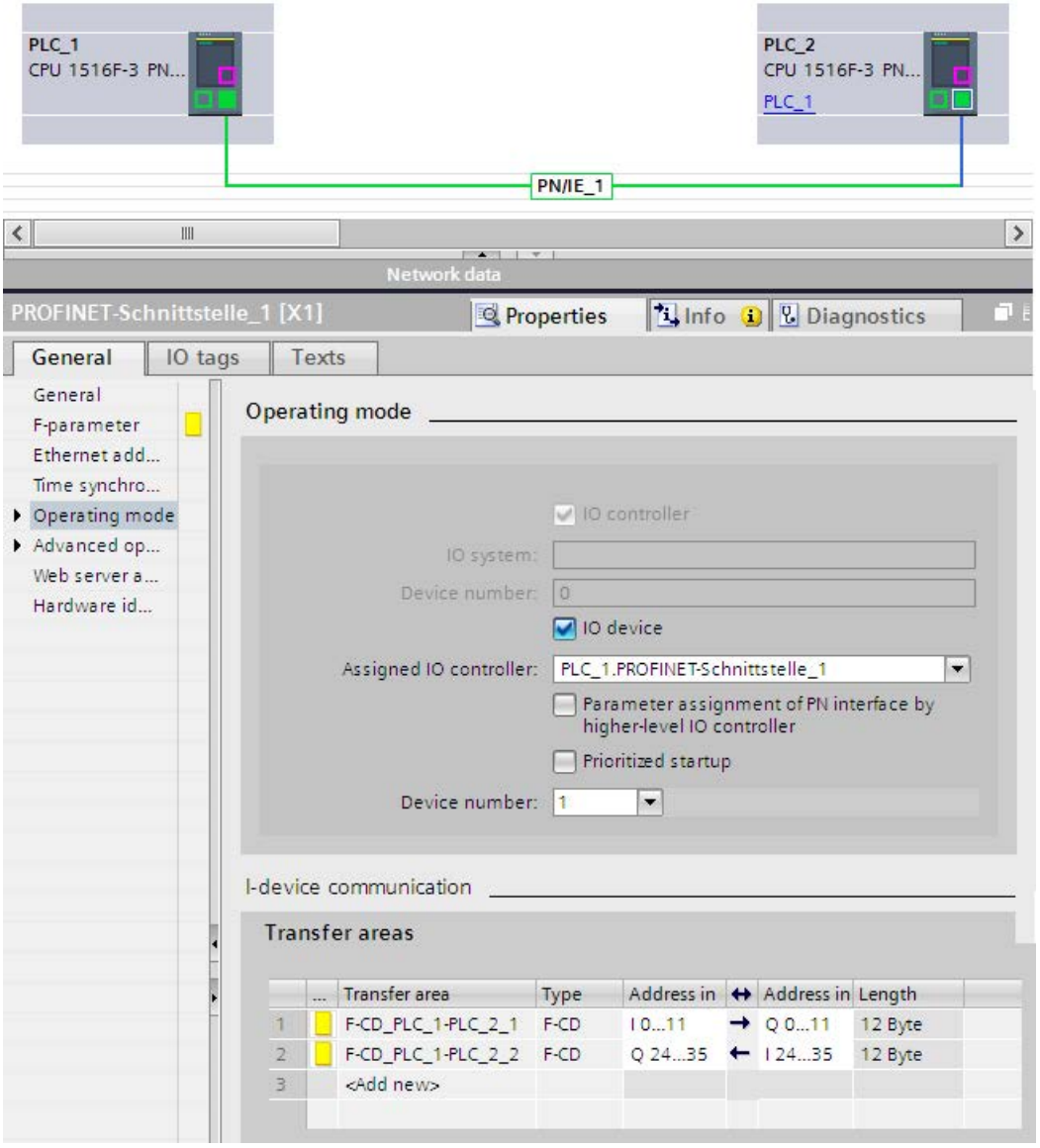
The procedure for configuring safety-related IO controller-I-device communication is identical to that in the standard system.

Proceed as follows:

1. Insert two F-CPUS from the "Hardware catalog" task card into the project.
2. Enable the "IO Device" mode for F-CPU 2 in the properties of its PN interface and assign this PN interface to a PN interface of F-CPU 1.
3. Select the PROFINET interface of F-CPU 2. Under "Transfer areas", create an F-CD connection (type "F-CD") for receiving from the IO controller (→). The F-CD connection is shown in yellow in the table and the address areas in the I-device and IO controller assigned are displayed.

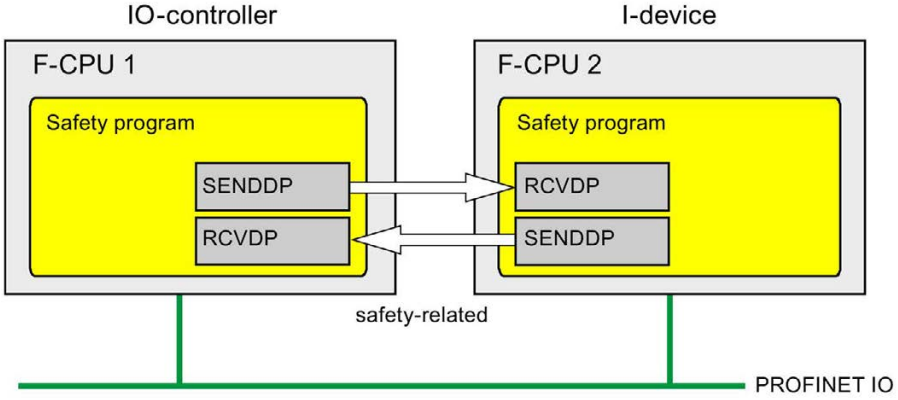
   In addition, an acknowledgment connection is created automatically for each F-CD connection. (see "Transfer area details").

4. Create an additional F-CD connection for sending to the IO controller.
5. In the transfer area you just created, click on the arrow to change the transfer direction to sending to the IO controller (←).



### 9.2.4.2 Safety-related IO controller-I-device communication via SENDDP and RCVDP

#### Communication via SENDDP and RCVDP instructions




Safety-related communication between the F-CPUs of the IO controller and an I-device makes use of the SENDDP and RCVDP instructions for sending and receiving, respectively. These can be used to perform a fail-safe transfer of a *fixed* amount of data of the data type BOOL or INT (DINT as alternative).

You can find these instructions in the "Instructions" task card under "Communication". The RCVDP instruction **must** be called at the start of the main safety block. The SENDDP instruction **must** be called at the end of the main safety block.

Note that the send signals are not sent until after the SENDDP instruction call at the end of execution of the relevant F-runtime group.

A detailed description of the SENDDP and RCVDP instructions can be found in SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500) (Page 646).

### 9.2.4.3 Programming safety-related IO controller I-device communication

#### Requirement for programming

The transfer areas must be configured.

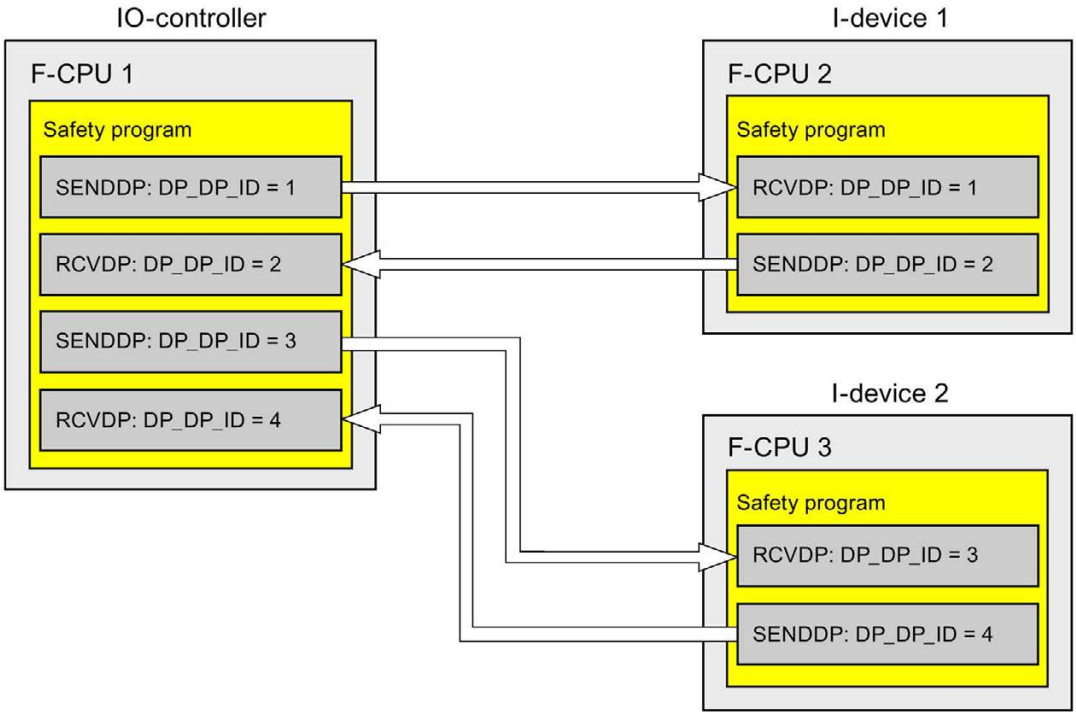
#### Programming procedure

The procedure for programming safety-related IO controller-I-device communication is the same as that for programming safety-related IO controller-IO controller communication (see Program safety-related IO controller-IO controller communication (Page 234)).

The assignment of the HW identifiers (system constants in the standard tag table) of the transfer areas to the LADDR parameter of the SENDDP/RCVDP instructions can be obtained from the following table:

| Instruction | HW identifier |
|---|---|
| SENDDP in the IO controller | Hardware identifier of the transfer area in the IO controller |
| RCVDP in the IO controller | Hardware identifier of the transfer area in the IO controller |
| SENDDP in the I-device | Hardware identifier of the transfer area in the I-device |
| RCVDP in the I-device | Hardware identifier of the transfer area in the I-device |

The figure below contains an example of how to specify the address relationships for the inputs of the SENDDP and RCVDP instructions for four safety-related IO controller-I-device communication relationships.



**WARNING**

The value for each address relationship (DP_DP_ID inputs; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

**WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal state is pending for at least as long as the assigned monitoring time. *(S018)*

Information on calculating the monitoring times can be found in Monitoring and response times (Page 661).

### 9.2.4.4 Safety-related IO-Controller-IO-Device communication - Limits for data transfer

## Limits for data transfer

If the amount of data to be transferred is greater than the capacity of related SENDDP/RCVDP instructions, you can use additional SENDDP/RCVDP instructions. Configure additional transfer areas for this purpose. Remember the maximum limit of 1440 bytes of input data or 1440 bytes of output data for transfer between an I-device and a IO controller.

The following table shows the amount of output and input data assigned in safety-related communication connections:

| Safety-related communication | Communication connection | Assigned input and output data | | | |
|---|---|---|---|---|---|
| | | In the IO controller | | In the I-device | |
| | | Output data | Input data | Output data | Input data |
| IO controller-I-Device | **Sending:**<br>I-Device 1 to IO controller | 6 bytes | 12 bytes | 12 bytes | 6 bytes |
| | **Receiving:**<br>I-Device 1 from IO controller | 12 bytes | 6 bytes | 6 bytes | 12 bytes |

Consider all additional configured safety-related and standard communication connections (transfer areas of type F-CD and CD) for the maximum limit of 1440 bytes of input data or 1440 bytes of output data for transfer between an I-device and an IO controller . In addition, data are assigned for internal purposes such that the maximum limit may be reached sooner.

When the limit is exceeded, a corresponding error message is displayed.

## 9.2.5 Safety-related master-I-slave communication

### 9.2.5.1 Configuring safety-related master-I-slave communication

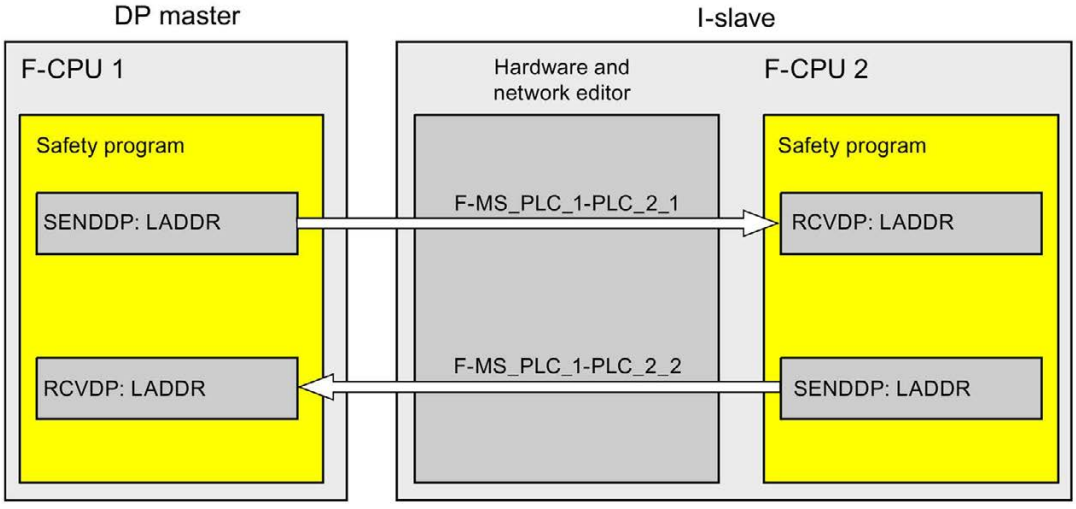
#### Introduction

Safety-related communication between the safety program of the F-CPU of a DP master and the safety program(s) of the F-CPU(s) of one or more I-slaves takes place over master-I-slave connections (**F-**MS), as in standard systems.

You do not need a DP/DP coupler for master-I-slave communication.

#### Configuring transfer areas

For every safety-related communication connection between two F-CPUs, you must configure transfer areas in the *hardware and network editor*. The figure below shows how both of the F-CPUs are able to send and receive data (bidirectional communication).




The transfer area is assigned a label when it is created to identify it as the communication relationship, for example "F-MS_PLC_2-PLC_1_1" for the first F-MS connection between DP master F-CPU 1 and I-slave F-CPU 2.

When you create a transfer area, a system constant with the name of the transfer area is created in the F-CPU of the DP master and in the F-CPU of the I-slave. The system constant contains the hardware identifier of the transfer area for the corresponding F-CPU.

You assign the hardware identifier (system constant from the default tag table) of the transfer areas symbolically to the LADDR parameter of the SENDDP and RCVDP instructions in the safety programs.

## Procedure for configuration

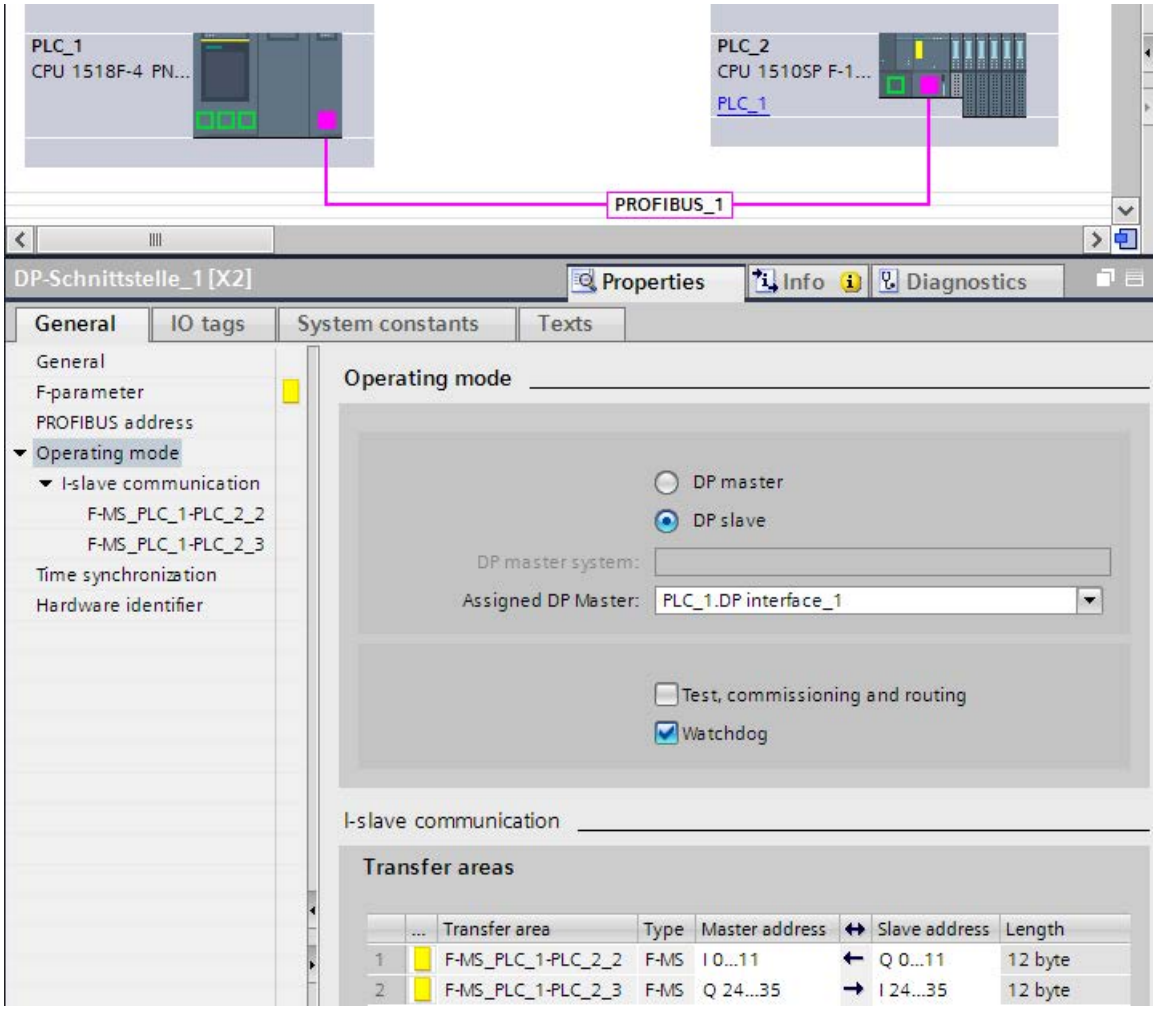
The procedure for configuring safety-related master-I-slave communication is identical to that in the standard system.

Proceed as follows:

1. Insert one F-CPU from the "Hardware catalog" task card as DP master in the project. If the F-CPU does not have an integrated PROFIBUS DP interface, insert e.g. an PROFIBUS-CM.
2. Insert a 1510SP F-1 PN CPU or 1512SP F-1 PN CPU as I-slave.
3. From the device view of the I-slave, insert a CM DP module.
4. Under properties, enable the "DP slave" mode (I-slave) for the CM DP module and assign this DP interface to a DP interface of F-CPU 1.
5. Select the PROFIBUS interface of F-CPU 2. Under "Transfer areas", you create an F-MS connection (type "F-MS") for sending to the DP master (←). The F-MS connection is shown in yellow in the table and the assigned transfer areas in the I-slave and DP master are displayed.

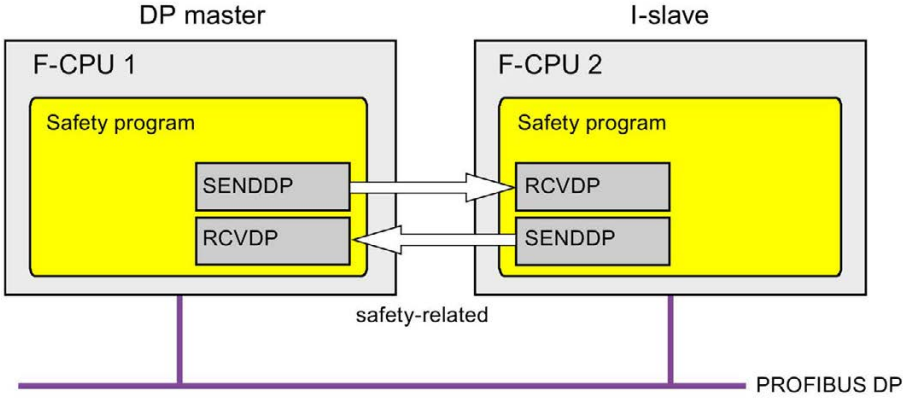
   In addition, an acknowledgment connection is created automatically for each F-MS connection. (see "Transfer area details").

6. Create an additional F-MS connection for receiving from the DP master.
7. In the transfer area you just created, click the arrow in order to change the transfer direction to receiving from DP master (→).



### 9.2.5.2 Safety-related master-I-slave communication via SENDDP and RCVDP

#### Communication via SENDDP and RCVDP instructions




Safety-related communication between the F-CPUs of the DP master and an I-slave makes use of the SENDDP and RCVDP instructions for sending and receiving, respectively. These can be used to perform a fail-safe transfer of a *fixed* amount of fail-safe data of the data type BOOL or INT (DINT as alternative).

You can find these instructions in the "Instructions" task card under "Communication". The RCVDP instruction **must** be called at the start of the main safety block. The SENDDP instruction **must** be called at the end of the main safety block.

Note that the send signals are not sent until after the SENDDP instruction call at the end of execution of the relevant F-runtime group.

A detailed description of the SENDDP and RCVDP instructions can be found in SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500) (Page 480).

### 9.2.5.3 Programming safety-related master-I-slave communication

#### Requirements

The transfer areas must be configured.

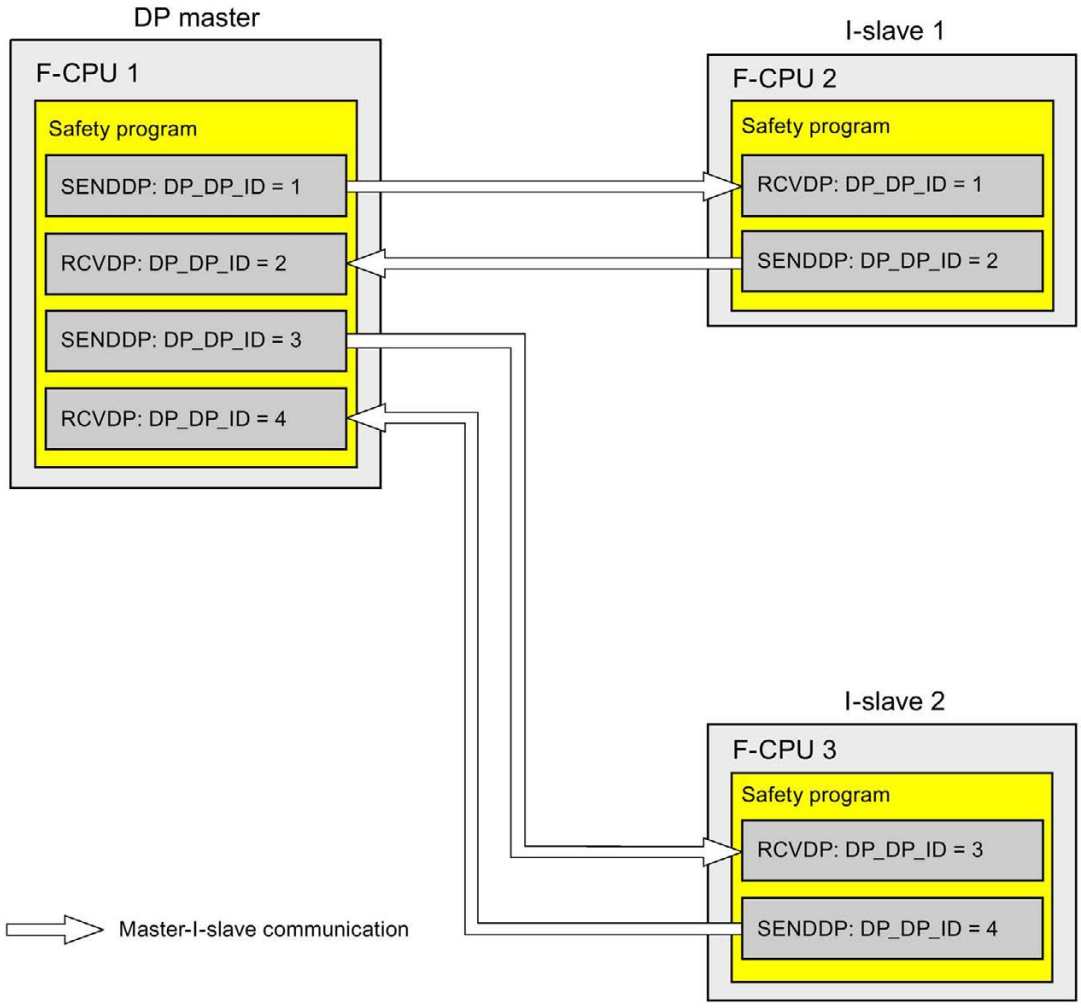
#### Programming procedure

The procedure for programming safety-related master-I-slave communication is the same as that for programming safety-related master-master communication (see Safety-related master-master communication (Page 238)).

The assignment of the HW identifiers of the transfer areas to the LADDR parameter of the SENDDP/RCVDP instructions can be obtained from the following table.

| Instruction | HW identifier |
|---|---|
| SENDDP in the DP master | Hardware identifier of the respective transfer area in the DP master |
| RCVDP in the DP master | Hardware identifier of the respective transfer area in the DP master |
| SENDDP in the I-slave | Hardware identifier of the transfer area in the I-slave |
| RCVDP in the I-slave | Hardware identifier of the transfer area in the I-slave |

The figure below contains an example of how to specify the address relationships at the inputs of SENDDP and RCVDP instructions for four safety-related master-I-slave communication relationships.




**WARNING**

The value for each address relationship (DP_DP_ID input; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

**⚠ WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal level is pending for at least as long as the assigned monitoring time. *(S018)*

Information on calculating the monitoring times can be found in Monitoring and response times (Page 661).

### 9.2.5.4 Limits for data transfer of safety-related master-I-slave communication

#### Limits for data transfer

If the amount of data to be transferred is greater than the capacity of related SENDDP/RCVDP instructions, you can use additional SENDDP/RCVDP instructions. Configure additional transfer areas for this purpose. Note the maximum limit of 244 bytes of input data or 244 bytes of output data for transfer between an I-slave and a DP master.

The following table shows the amount of output and input data assigned in safety-related communication connections:

| Safety-related communication | Communication connection | Assigned input and output data | | | |
|---|---|---|---|---|---|
| | | DP master | | I-slave | |
| | | Output data | Input data | Output data | Input data |
| Master-I-slave | **Sending:**<br>I-slave 1 to DP master | 6 bytes | 12 bytes | 12 bytes | 6 bytes |
| | **Receiving:**<br>I-slave 1 from DP master | 12 bytes | 6 bytes | 6 bytes | 12 bytes |

Consider all additional configured safety-related and standard communication connections (transfer areas of type F-MS-, F-DX-, F-DX-Mod., MS-, DX- and DX-Mod) for the maximum limit of 244 bytes of input data or 244 bytes of output data for transfer between an I-device and a DP master F-MS and MS). If the maximum limit of 244 bytes of input data or 244 bytes of output data is exceeded, you will receive a corresponding error message.

## 9.2.6 Safety-related IO controller-I-slave communication

### 9.2.6.1 Safety-related IO controller-I-slave communication

#### Introduction

Safety-related communication between the safety program of the F-CPU of an IO-controller and the safety program(s) of the F-CPU(s) of one or more I-slaves takes place over master-I-slave connections (**F-**MS), as in standard systems.

#### IE/PB link

For the safety-related IO-controller-I-slave communication, the IE/PB link is mandatory. Each of the two F-CPUs is linked to the IE/PB link by means of its PROFIBUS DP or PROFINET-interface.

**Note**

If you are using an IE/PB link, you must take this into account when configuring the F-specific monitoring times and when calculating the maximum response time of your F-system (see also Monitoring and response times (Page 661)).

Note that the Excel file for calculating response times (http://support.automation.siemens.com/WW/view/en/49368678/133100) for S7-300/400 F-CPUs does not support all conceivable configurations.

#### Reference

The information on safety-related master-I-slave communication in Safety-related master-I-slave communication (Page 254) also applies.

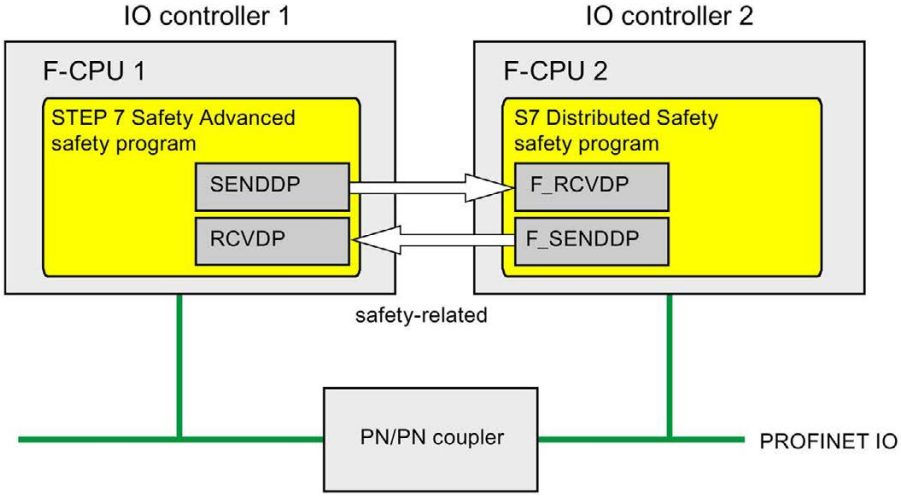
## 9.2.7 Safety-related communication with S7 F-systems

### 9.2.7.1 Introduction

Safety-related communication from F-CPUs in SIMATIC Safety to F-CPUs in S7 Distributed Safety F-systems is possible, via a PN/PN coupler or DP/DP coupler that you use between the two F-CPUs, as IO controller-IO controller communication or master-master communication.

### 9.2.7.2 Communication with S7 Distributed Safety via PN/PN coupler (IO controller-IO controller communication)

Communication functions between SENDDP/RCVDP instructions at the *STEP 7 Safety* end and F-application blocks F_SENDDP/F_RCVDP at the *S7 Distributed Safety* end:




#### Procedure at the *S7 Distributed Safety* end

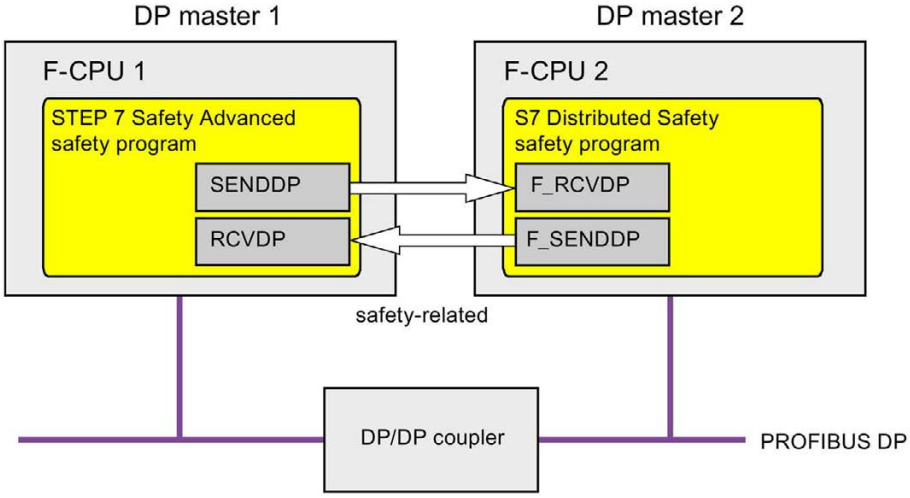
At the *S7 Distributed Safety* end, proceed as described in "Safety-related IO controller-IO controller communication" in the S7 Distributed Safety - Configuring and Programming (http://support.automation.siemens.com/WW/view/en/22099875) manual.

#### Procedure at the *STEP 7 Safety* end

At the *STEP 7 Safety* end, proceed as described in Safety-related IO controller-IO controller communication (Page 229).

### 9.2.7.3 Communication with S7 Distributed Safety via DP/DP coupler (master-master communication)

Communication functions between SENDDP/RCVDP instructions at the *STEP 7 Safety* end and F-application blocks F_SENDDP/F_RCVDP at the *S7 Distributed Safety* end:




#### Procedure at the *S7 Distributed Safety* end

At the *S7 Distributed Safety* end, proceed as described in "Safety-related master-master communication" in the S7 Distributed Safety - Configuring and Programming (http://support.automation.siemens.com/WW/view/en/22099875) manual.

#### Procedure at the *STEP 7 Safety* end

At the *STEP 7 Safety* end, proceed as described in Safety-related master-master communication (Page 238).

# 9.3 Configuring and programming communication between S7-300/400 and S7-1500 F-CPUs

## 9.3.1 Overview of communication

### Introduction

This section provides an overview of the options for safety-related communication between S7-300/400 and S7-1500 F-CPUs in SIMATIC Safety F-systems.

### Options for safety-related communication

| Safety-related communication | Particularities | On subnet | Additional hardware required |
|---|---|---|---|
| **Safety-related CPU-CPU communication:** | | | |
| Master-master communication | — | PROFIBUS DP | DP/DP coupler |
| Master-I-slave communication | — | PROFIBUS DP | — |
| IO controller-IO controller communication | — | PROFINET IO | PN/PN coupler |
| IO controller-I-device communication | — | PROFINET IO | — |

### Basic procedure for configuring and programming

Configure and program safety-related communication between S7-300/400 F-CPUs and S7-1500 F-CPUs as described in Configuring and programming communication (S7-300, S7-400) (Page 163) and Configuring and programming communication (S7-1500) (Page 226) for your application.

To program an S7-300/400 F-CPU, use the start addresses of the transfer areas. To program an S7-1500 F-CPU, use the HW identifiers of the transfer areas.

# Compiling and commissioning a safety program 10

## 10.1 Compiling the safety program

### Introduction

To compile a safety program, follow the same basic procedure as for compiling a standard user program. You can start at various points to accomplish this in *STEP 7*. The basics for compiling user programs can be found in the *Help on STEP 7*.

**Note**

Please note that following a safety-related change to the hardware configuration that not only this, but also the safety program has to be recompiled and downloaded. This also applies for changes to the F-I/O which are not used in the safety program.

We will show you the options for compiling the safety program below.

### Basics for compiling

| 1. Selection | 2. Via menu command/symbol… | 3. The following are compiled... | 4. Result: Safety program is... |
|---|---|---|---|
| **Select folder of the F-CPU in the project tree**<br>or<br>**Select F-CPU in network view**<br>or<br>**Select F-CPU in the device view**<br>or<br>**Select F-CPU in the topology view** | "Compile" shortcut menu: | | |
| | • "Hardware and software (only changes)" | Hardware configuration, standard and safety programs | Consistent |
| | • "Hardware (only changes)" | Hardware configuration | — |
| | • "Software (only changes)" | Changes in the standard and safety programs | Consistent |
| | • "Software (rebuild all blocks)" | Entire standard program and safety program | Consistent |
| | • "Software (reset memory reserve)" (S7-1200, S7-1500) | Entire standard program and safety program | Consistent |
| **Select Safety Administration Editor** | Menu "Edit > Compile" or "Compile" button | Entire standard program and safety program | Consistent |

| 1. Selection | 2. Via menu command/symbol… | 3. The following are compiled... | 4. Result: Safety program is... |
|---|---|---|---|
| **Select "Program blocks" folder in the project tree** | Menu "Edit > Compile" or "Compile" button | Changes in the standard and safety programs | Consistent |
| | "Compile" shortcut menu: | | |
| | • "Software" | Changes in the standard and safety programs | Consistent |
| | • "Software (rebuild all blocks)" | Entire standard program and safety program | Consistent |
| **Select user-created folder in the project tree** | Menu "Edit > Compile" or "Compile" button | All standard and F-blocks contained in the folder | Inconsistent |
| **User-created folder containing all F-blocks** | Menu "Edit > Compile" or "Compile" button | All standard blocks contained in the folder and complete safety program | Consistent |
| **(F-)blocks in "Program blocks" folder in the project tree** | Menu "Edit > Compile" or "Compile" button | Selected standard and F-blocks | Inconsistent |
| **Different (F-)blocks in "Program blocks" folder in the project tree** | | | |
| **Individual (F-)blocks in "Program blocks" folder in the project tree** | | | |
| **Downloading a safety program to an F-CPU or memory card** | Automatic compiling | Changes in the standard and safety programs | Consistent |

A consistency check is always performed, regardless of the selection. This consistency check extends across all selected blocks. If the consistency check does not detect any errors, the status of the compiled safety program is as specified in column 4 "Result: Safety program is..." of the table.

**Note**

The following applies for S7-300/400 F-CPUs:

If you want to compile an F-FB with know-how protection, open it before compiling.

## Result "Safety program is consistent"

After successful compilation of the safety program, a consistent safety program is always available in the "Program blocks" folder.

There may still be some F-blocks that are not called in an F-runtime group. These F-blocks are displayed in the "F-blocks" area of the *Safety Administration Editor*, marked with "No" in the "Used and compiled" column.

### Result "Safety program is inconsistent"

When the safety program is compiled with the result "Safety program is not consistent", only the selected F-blocks were compiled. Additionally required F-blocks and F-system blocks were not generated. The safety program in the "Program blocks" folder is not consistent and is, thus, not executable.

Use this procedure to test if modified F-blocks can be compiled.

### Reporting compiling errors

You can recognize whether or not the compilation was successful based on the message in the inspector window under "Info > Compile", error messages and warnings are output.

For information on the procedure you must follow to eliminate compiling errors, see "Eliminating compiling errors" in the *Help on STEP 7*.

### See also

Safety Administration Editor (Page 64)

# 10.2 Downloading the Safety Program

## Introduction

Once you have successfully compiled your safety program, you can download it to the F-CPU. To download a safety program, you follow essentially the same approach as for downloading a standard user program, via different starting points in *STEP 7*.

- In the "Load preview" dialog, enter data (e.g. password for the F-CPU) and set the requirements for downloading (e.g. that the F-CPU is switched to STOP mode before downloading).
- The "Load results" dialog shows the results after downloading.

We will show you the options for downloading the safety program later. For basic information on downloading, refer to the *Help on STEP 7*.

## Downloading a safety program to an F-CPU, if multiple F-CPUs are accessible

**⚠ WARNING**

If **multiple F-CPUs** can be accessed over a network (e.g. Industrial Ethernet) by **a programming device or PC**, you must take the following additional measures to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, such as a unique password for the F-CPU having the respective Ethernet address "PW_8".

Note the following:

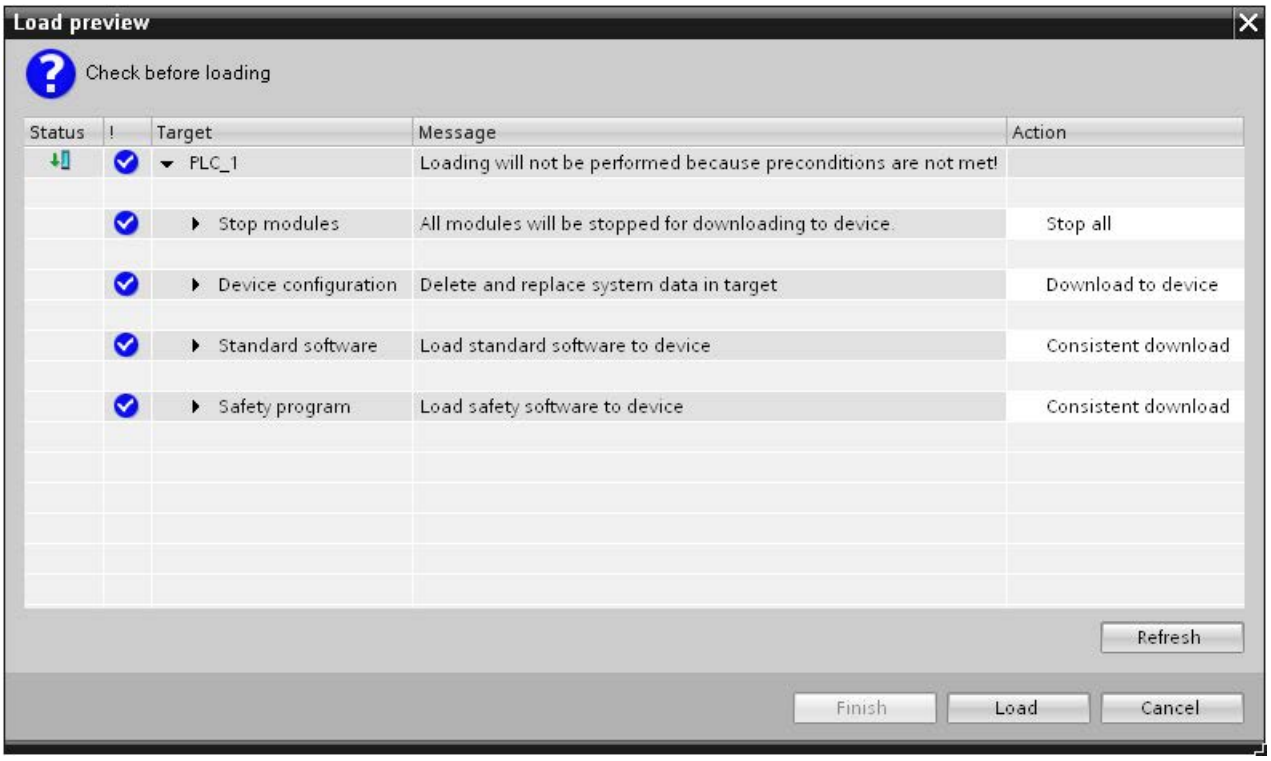
- A point-to-point connection must be used when assigning a password to an F-CPU for the first time (as when assigning an MPI address to an F-CPU for the first time).
- Before downloading a safety program to an F-CPU, you must first revoke an existing access permission for any other F-CPU.
- After activation of the access protection and before the transition to productive mode, you must download the safety program again to the F-CPU. *(S021)*

## Password prompt before downloading to an F-CPU

If you have assigned a protection level for the F-CPU (Page 82) (in the properties of the F-CPU in the "Protection" tab), the corresponding password is prompted in "Load preview" dialog. Without entry of password, only actions that are allowed without password are possible. As soon as the conditions for downloading are met, the "Load" button becomes active.

## "Load preview" dialog

For an F-CPU, the "Load preview" dialog also contains the section "Safety program".



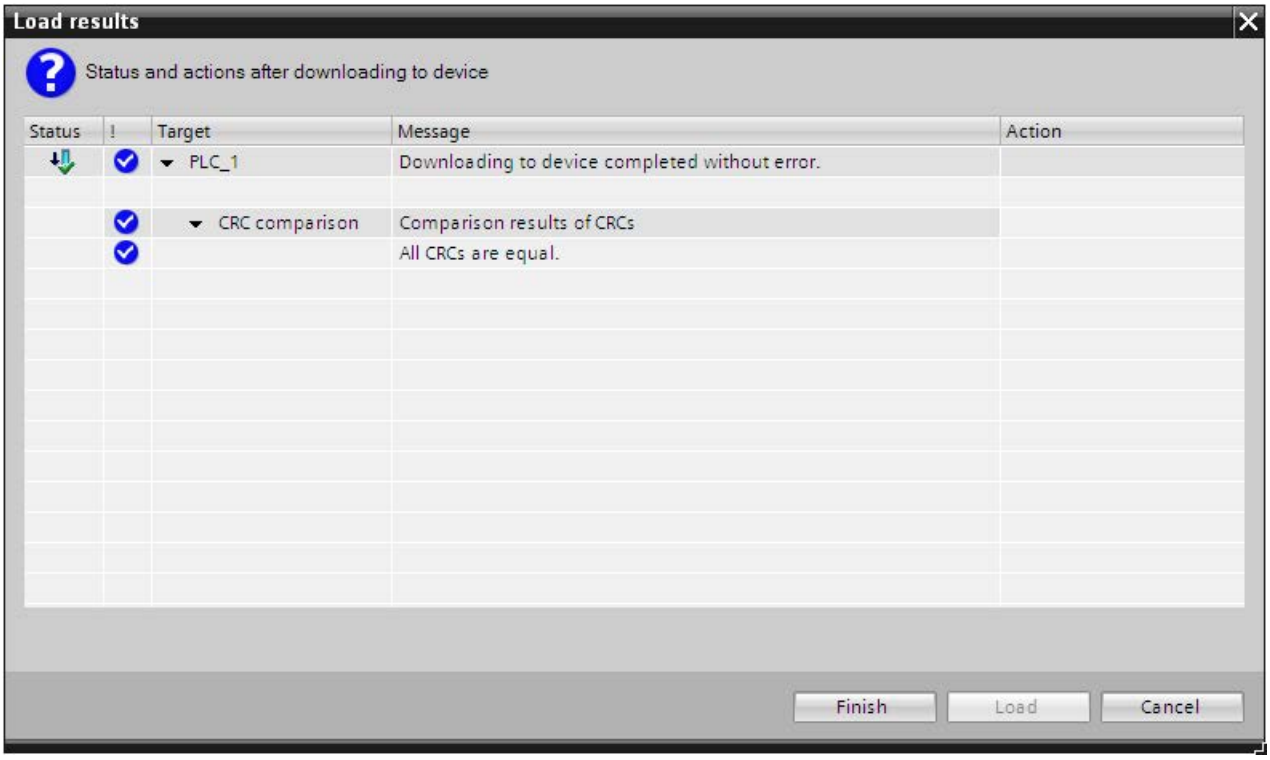

Make the following selection:

- In order to download a consistent safety program, select the "Consistent download" action under target "Safety program".
- You have the option of selectively downloading individual F-blocks to an S7-300/400 F-CPU. To download individual F-blocks selectively, select the "Download selection" action under the target "Safety program", and then select the required F-blocks. If necessary, you will be prompted to disable safety mode under "Disable safety mode". This setting is only suitable for the online test of individual F-blocks.
- (S7-300, S7-400) In order to download the safety program only, select the "Consistent download" action under target "Safety program" and the "Download selection" action under target"Standard software", and then select only the standard blocks that call the main safety block.
- Users who do not know the password for the F-CPU cannot download F-blocks. They have to select "No action" for the safety program.

For S7-1200/1500 F-CPUs, only the "Consistent download" value is possible as an action in the "Load preview" dialog. It is not possible to select separate loading of standard program or safety program. The complete user program is automatically consistently downloaded as soon as changes have been made in both the standard program and in the safety program. Please also note Data exchange between standard user program and safety program (Page 159).

## "Load results" Dialog

After downloading into the F-CPU, the dialog "Load results" is opened. This dialog shows you the status and the necessary actions after downloading.




Verify that the "All CRCs are identical" message appears in this dialog. On the basis of the CRCs, the system checks after the download operation to determine whether all F-blocks were correctly transferred to the F-CPU.

## Rules for downloading the safety program to an F-CPU

**Note**

You can perform the downloading of a consistent safety program only in STOP mode.

(S7-300, S7-400) If you are downloading F-blocks only, the blocks in which the main safety blocks are called (e.g., cyclic interrupt OB 35) are not downloaded. To do so, select the "Selection" option under "Standard software" in the preview dialog, and select the necessary blocks.

**Note**

If *STEP 7 Safety* detects an inconsistent safety program during startup of the F-CPU, startup of the F-CPU is prevented if the F-CPU supports this detection. (See product information for the respective S7-300/400 F-CPU. This is always supported with S7-1200/1500 F-CPUs). A corresponding diagnostic event is entered in the diagnostic buffer of the F-CPU:

If the F-CPU does not support this detection function, the F-CPU can go to STOP mode if an inconsistent safety program is executed in activated safety mode.

The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

When downloading the safety program, ensure that the "Consistent download" action is set for the "Safety program" selection in the "Load preview" dialog.

Verify that the "All CRCs are identical" message appears in the "Load results" dialog for S7-300/400 F-CPUs. The system uses CRCs after the download operation to check whether all F-blocks have been correctly transferred to the F-CPU. If not, repeat the download operation.

Inconsistent downloading is only possible in disabled safety mode.

## Downloading individual F-blocks (S7-300, S7-400)

You can download F-blocks and standard blocks simultaneously to the F-CPU via the project tree. However, as soon as F-blocks are to be downloaded, a check is carried out to determine whether or not the F-CPU is in STOP mode or disabled safety mode. If not, you have the option of switching to disabled safety mode or placing the F-CPU in STOP mode. We recommend that you download individual F-blocks in disabled safety mode because downloading to an F-CPU in STOP results in the F-CPU no longer being able to startup following the download.

If you want to download individual F-blocks to the F-CPU, for example, to test changes, make sure that you have not selected the folder "Program blocks" or the F-CPU in the project tree but only one of the blocks you want to download.

Only then will you be prompted in the "Load preview" dialog to disable safety mode once you have changed the option from "Consistent download" to "Download selection".

If you fail to observe this prompt, the blocks are downloaded without deactivation of the safety mode which will STOP the F-CPU.

You can also deactivate the safety mode explicitly in the *Safety Administration Editor* before you start the download.

Be aware that the consistency of the safety program in the F-CPU cannot be guaranteed when individual F-blocks are downloaded. For a consistent safety program, always download the entire safety program to the F-CPU.

### Rules for downloading individual F-blocks (S7-300, S7-400)

The following rules apply to downloading of individual F-blocks:

- Downloading is only possible in disabled safety mode or when the F-CPU is in STOP mode.
- F-blocks can only be downloaded to an F-CPU to which a safety program has already been downloaded.
- Only an offline safety program is permitted to be used as a source program.

Consequently, you have to download the entire safety program when initially downloading the safety program and after changing the password for the safety program.

**Note**

Downloading of individual F-blocks is only suitable for testing F-blocks. Prior to the transition to productive mode, you must download the safety program consistently to the F-CPU.

### Transferring a safety program with programming device/PC or memory card to an F-CPU

For information on transferring the safety program to the F-CPU with a programming device/PC or a memory card, refer to Function test of safety program and protection through program identification (Page 274).

### Uploading the safety program to a programming device or PC (S7-300, S7-400)

**Note**

In principle, it is possible to upload a safety program from the F-CPU to a programming device or PC. Note, however, that any symbols used in the safety program are deleted and cannot be recreated, since no symbol information is saved in the F-CPU. Symbols are available only if you are using an offline project.

A safety program uploaded from the F-CPU to a programming device/PC must not be downloaded again to the F-CPU.

### Uploading the safety program to a programming device or PC (S7-1200, S7-1500)

The "Load from device" function has not been released for S7-1200/1500 F-CPUs. This also applies to F-I/O in standard mode.

### Uploading to a PG/PC (S7-300, S7-400)

You upload a safety program from an F-CPU to a programming device/PC using menu command "Online> Upload device to PG/PC" or the "Load from device" button in the toolbar.

You can only upload the hardware configuration if F-capability is not enabled for the F-CPU. You can only upload F-I/O if these are operated in standard mode.

### Download in *S7-PLCSIM* (S7-300, S7-400, S7-1500)

You can load the safety program with *S7-PLCSIM* (hardware simulation) like in a real F-CPU. For this purpose, use the menu command "Online > Simulation > Start" to start *S7-PLCSIM*, same as in the standard system.

Note that the cycle time on a simulated CPU can differ from the real cycle time.

## 10.3 Safety program work memory requirements (S7-300, S7-400)

### Estimation

You can estimate the work memory requirement for the safety program as follows:

**Work memory required for the safety program**

32 KB for F-system blocks

+ 4.4 KB for safety-related communication between F-runtime groups
+ 4.5 x work memory requirement for all F-FB/F-FCs/main safety blocks
+ 4.5 x work memory requirement of all utilized instructions, which are shown in the "Instructions" task card with the block icon.
  (except SENDDP, RCVDP, SENDS7, and RCVS7)
+ Work memory requirement of the utilized SENDDP and RCVDP instructions
  (4.3 KB each)
+ Work memory requirement of the utilized SENDS7 and RCVS7 instructions
  (8.5 KB each)

**Work memory required for data**

5 x work memory requirement for all F-DBs (including F-communication DB, but excluding DB for F-runtime group communication) and I-DBs for main safety block/F-FB

+ 24 x work memory requirement for all DBs for F-runtime group communication
+ 2.3 x work memory requirement for all I-DBs of instructions (except SENDDP, RCVDP, SENDS7 and RCVS7)
+ Work memory requirement of all I-DBs of instructions SENDDP (0.2 KB), RCVDP (0.3 KB), SENDS7 (0.6 KB), and RCVS7 (1.0 KB)
+ 0.7 KB per F-FC
+ 0.7 KB per F-I/O (for F-I/O DBs, etc.)
+ 4.5 KB

### Block size of automatically generated F-blocks

Do not utilize the entire maximum size of an F-block, because the automatically generated F-blocks are larger and as a result, the maximum possible size may be exceeded in the F-CPU. If the block size is exceeded, this triggers a corresponding error message with information on which F-blocks are too large. These must be divided up, if necessary.

## 10.4 Function test of safety program and protection through program identification

### 10.4.1 Transferring the safety program to an S7-300/400 F-CPU with memory card inserted (flash card or SIMATIC Micro memory card)

#### F-CPUs with memory card inserted (flash card or SIMATIC Micro memory card)

The following warnings apply when the safety program is transferred from a programming device or PC to:

- F-CPUs with flash card inserted (e.g. CPU 416F-2)
- F-CPUs with SIMATIC Micro memory card

  (e.g. CPU 317F-2 DP, CPU 315F-2 PN/DP or IM 151-7 F-CPU)

**⚠ WARNING**

If the function test of the safety program is not carried out in the destination F-CPU, you must adhere to the following procedure when transferring the safety program to the F-CPU with a **programming device/PC** to ensure that the F-CPU does not contain an "old" safety program:

- Load the safety program on the F-CPU.
- Perform a program identification (i.e. check whether the collective F-signatures match online and offline).
- Perform a memory reset of the F-CPU using the mode selector or via the programming device/PC. Once the work memory has been deleted, the safety program is again transferred from the load memory (memory card, SIMATIC Micro memory card for F-CPUs 3xxF, ET 200S and ET 200pro IM-CPUs or flash card for F-CPUs 4xxF). *(S022)*

**WARNING**

If **multiple F-CPUs** can be accessed over a network (e.g. Industrial Ethernet) by **a programming device or PC**, you must take the following additional measures to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, such as a unique password for the F-CPU having the respective Ethernet address "PW_8".

Note the following:

- A point-to-point connection must be used when assigning a password to an F-CPU for the first time (as when assigning an MPI address to an F-CPU for the first time).
- Before downloading a safety program to an F-CPU, you must first revoke an existing access permission for any other F-CPU.
- After activation of the access protection and before the transition to productive mode, you must download the safety program again to the F-CPU. *(S021)*

## Rules for plugging removable media into the F-CPU

**WARNING**

You must limit access to the F-CPU to persons who are authorized to plug in removable media by using access protection.

You must ensure that the correct safety program is on the connected removable medium, either with online program identification or other appropriate measures (for example, unique ID of the removable medium). *(S025)*

## 10.4.2 Transferring the safety program to an S7-400 F-CPU without flash card inserted

The following warnings apply when the safety program is transferred from a programming device or PC to:

- F-CPUs without an inserted flash card (e.g. CPU 416F-2)

**⚠ WARNING**

If the function test of the safety program is not carried out in the destination F-CPU, you must adhere to the following procedure when transferring the safety program to the F-CPU with a **programming device/PC** to ensure that the F-CPU does not contain an "old" safety program:

- Perform a memory reset of the F-CPU using the **mode selector** or via the **programming device/PC**.
- Download the configuration and the safety program to the F-CPU.
- Perform a program identification (i.e. check whether the collective F-signatures match online and offline). *(S023)*

**⚠ WARNING**

If **multiple F-CPUs** can be accessed over a network (e.g. Industrial Ethernet) by **a programming device or PC**, you must take the following additional measures to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, such as a unique password for the F-CPU having the respective Ethernet address "PW_8".

Note the following:

- A point-to-point connection must be used when assigning a password to an F-CPU for the first time (as when assigning an MPI address to an F-CPU for the first time).
- Before downloading a safety program to an F-CPU, you must first revoke an existing access permission for any other F-CPU.
- After activation of the access protection and before the transition to productive mode, you must download the safety program again to the F-CPU. *(S021)*

## 10.4.3 Transferring the safety program to an S7-300/400 F-CPU with memory card

**Use of SIMATIC Micro memory card or flash card**

The following warning applies when you use:

- Flash card (e.g. for CPU 416F-2)
- SIMATIC Micro memory card (e.g. for CPU 317F-2 DP, CPU 315F-2 PN/DP or IM 151-7 F-CPU)

**⚠ WARNING**

If the function test of the safety program is not carried out in the destination F-CPU, you must adhere to the following procedure when transferring the safety program to the F-CPU with a memory card (SIMATIC Micro memory card or flash card) to ensure that the F-CPU does not contain an "old" safety program:

- Switch off the power to the F-CPU. For F-CPUs with battery backup (e.g. CPU 416F-2), remove any battery. (To make sure that the F-CPU is de-energized, wait for the buffer time of the power supply you are using or, if this is not known, remove the F-CPU.)
- Remove the memory card (SIMATIC Micro memory card or flash card) with the old safety program from the F-CPU.
- Plug the memory card (SIMATIC Micro memory card or flash card) with the new safety program into the F-CPU.
- Switch on the F-CPU again. For F-CPUs with battery backup (e.g. CPU 416F-2), reinsert the battery, if one was removed.

You need to ensure that the memory card inserted (SIMATIC Micro memory card or flash card) contains the correct safety program. You can do so using program identification or other measures, such as a unique identifier on the memory card (SIMATIC Micro memory card or flash card).

When loading a safety program on a memory card (**SIMATIC Micro memory card** or flash card), you must adhere to the following procedure:

- Load the safety program on the memory card (SIMATIC Micro memory card or flash card).
- Perform a program identification (in other words, check whether the collective F-signatures offline and on the memory card, SIMATIC Micro memory card or flash card match).
- Label the memory card (SIMATIC Micro memory card or flash card) appropriately.

The procedure outlined must be ensured through organizational measures. *(S026)*

### Rules for plugging removable media into the F-CPU

**⚠ WARNING**

You must limit access to the F-CPU to persons who are authorized to plug in removable media by using access protection.

You must ensure that the correct safety program is on the connected removable medium, either with online program identification or other appropriate measures (for example, unique ID of the removable medium). *(S025)*

## 10.4.4 Transferring the safety program to a WinAC RTX F

**⚠ WARNING**

In order to guarantee that there is no "old" safety program in the F-controller, you must adhere to the following procedure when transferring the safety program to the F controller with a programming device/PC:

1. Perform a memory reset of WinAC RTX F (see Windows Automation Center RTX WinAC RTX (F) 2010 (http://support.automation.siemens.com/WW/view/en/43715176) manual).
2. Download the configuration to the WinAC RTX F (see Windows Automation Center RTX WinAC RTX (F) 2010 (http://support.automation.siemens.com/WW/view/en/43715176) manual).
3. Download the safety program (Page 268) to the WinAC RTX F.

   **If the function test of the safety program does not take place in the destination F-controller, you must also follow points 4 and 5:**
4. Perform a program identification (i.e. check whether the collective F-signatures match online and offline).
5. Perform the F-system startup.

Between the online program identification and the startup of the F-system, the WinAC RTX F must not be closed (for example, as a result of NETWORK OFF/NETWORK ON or booting). *(S024)*

**WARNING**

If **multiple F-CPUs** can be accessed over a network (e.g. Ind. Ethernet) by **a programming device or PC**, you must also take the following additional measures to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, such as a unique password for the F-CPUs having the respective Ethernet address "PW_8".

Note the following:

- A point-to-point connection must be used when assigning a password to an F-CPU for the first time (as when assigning an MPI address to an F-CPU for the first time).
- Before downloading a safety program to an F-CPU, you must first revoke existing access permission for any other F-CPU.
- After activation of the access protection and before the transition to productive mode, you must download the safety program again to the F-CPU. *(S021)*

### Rules for plugging removable media (for example SIMATIC Micro memory cards, flash cards and hard disks) into the F-CPU

**WARNING**

You must limit access to the F-CPU to persons who are authorized to plug in removable media by using access protection.

You must ensure that the correct safety program is on the connected removable medium, either with online program identification or other appropriate measures (for example, unique ID of the removable medium). *(S025)*

## 10.4.5 Transferring the safety program to an S7-1200/1500 F-CPU

**WARNING**

If the function test of the safety program is not carried out in the destination F-CPU, you must adhere to the following procedure when transferring the safety program to the F-CPU with a **programming device/PC** to ensure that the F-CPU does not contain an "old" safety program:

- Load the safety program on the F-CPU.
- Perform a program identification (i.e. check whether the collective F-signatures match online and offline). *(S042)*

**WARNING**

If **multiple F-CPUs** can be accessed over a network (e.g. Industrial Ethernet) by **a programming device or PC**, you must take the following additional measures to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, such as a unique password for the F-CPU having the respective Ethernet address "PW_8".

Note the following:

- A point-to-point connection must be used when assigning a password to an F-CPU for the first time (as when assigning an MPI address to an F-CPU for the first time).
- Before downloading a safety program to an F-CPU, you must first revoke an existing access permission for any other F-CPU.
- After activation of the access protection and before the transition to productive mode, you must download the safety program again to the F-CPU. *(S021)*

### Rules for plugging removable media into the F-CPU

**WARNING**

You must limit access to the F-CPU to persons who are authorized to plug in removable media by using access protection.

You must ensure that the correct safety program is on the connected removable medium, either with online program identification or other appropriate measures (for example, unique ID of the removable medium). *(S025)*

### Using a SIMATIC memory card

**⚠ WARNING**

If the function test of the safety program is not performed in the destination F-CPU, you need to ensure that the SIMATIC memory card inserted has the correct safety program. You can do so through a program identification or other measures, such as a unique identifier on the SIMATIC memory card.

When downloading a safety program to a SIMATIC memory card, you must adhere to the following procedure:

- Load the safety program on the SIMATIC memory card.
- Perform a program identification (i.e. check to determine whether the collective F-signatures offline and on the SIMATIC memory card match).
- Label the SIMATIC memory card accordingly.

The procedure outlined must be ensured through organizational measures. *(S043)*

## 10.4.6 Restoring a backup of the safety program to an S7-300/1500 F-CPU

You have the option of backing up an F-CPU in the same way as a standard CPU. You can find information on backing up a CPU in the *online help on STEP 7* under "Creating a backup of an S7 CPU".

Note the following warning when using an F-CPU:

**⚠ WARNING**

After restoring a safety program for an F-CPU, you must connect with the F-CPU using *STEP 7* and read out the collective F-signature from the F-CPU load memory using the SAE and compare it with your expected result. You can also check the result using the F-CPU display. *(S055)*

## 10.4.7 Transferring the safety program to an S7-1200 F-CPU with inserted program card

**⚠ WARNING**

You must observe the following procedure to ensure that there is no "old" safety program in the internal load memory of the F-CPU when you insert a program card into an S7-1200 F-CPU:

1. If the internal load memory of the F-CPU has already been deleted (for example, when the F-CPU has already been operated with a program card as external load memory), **the STOP/RUN LED (orange) and the maintenance LED must flash during startup for 3 seconds on an F-CPU without SIMATIC Memory Card**. You can skip steps 3 in this case.
2. Insert the program card into the F-CPU.

   If the F-CPU is in RUN, it will change to STOP. The maintenance LED on the F-CPU is flashing to indicate that the program card is being evaluated or that the internal load memory must be deleted.
3. Use one of the following methods to delete the internal load memory:
   - Turn the F-CPU off and back on.
   - Switch the F-CPU from STOP to RUN.
   - Execute the "Memory reset" (MRES) function.

   After restart and deletion of the internal load memory, **the STOP/RUN LED (orange) and the maintenance LED must be flashing**. The internal load memory of the F-CPU has been deleted in this case and does no longer store an "old" safety program.
4. Use one of the following methods to evaluate the program card:
   - Turn the F-CPU off and back on.
   - Switch the F-CPU from STOP to RUN.
   - Execute the "Memory reset" (MRES) function.

   The F-CPU restarts and evaluates the program card.

   The F-CPU then enters the startup mode (RUN or STOP) that has been set up for the F-CPU. *(S061)*

**⚠ WARNING**

If you use a programming device/PC to transfer F-blocks to an S7-1200 F-CPU with inserted program card (external load memory), you must ensure that the transfer takes place to the external load memory. This can be accomplished, for example, by the following measures:

- Check to see if the program card is inserted correctly.
- Insert a program card whose memory size is different than the size of the internal load memory. Check in the project tree under "Online & Diagnostics > Diagnostics > Memory" if the memory size displayed for the load memory matches the memory size of the program card. *(S058)*

**Note**

For an S7-1200 F-CPU without inserted SIMATIC Memory Card and deleted internal load memory, the status LEDs have the status described in the table below.

| Description | STOP/RUN Orange/Green | ERROR Red | MAINT Orange |
|---|---|---|---|
| Internal load memory deleted and SIMATIC Memory Card not inserted. | Flashing (orange)<br>(for 3 seconds during startup) | Off | Flashing<br>(for 3 seconds during startup) |

### 10.4.8 Transferring the safety program from the internal load memory to an empty SIMATIC Memory Card of an S7-1200 F-CPU

**⚠ WARNING**

To ensure that the safety program is transferred from the internal load memory of the F-CPU to the SIMATIC Memory Card when plugging an empty SIMATIC Memory Card into an S7-1200 F-CPU and that the internal load memory of the F-CPU is deleted afterward, you must observe the following procedure:

1. Make sure that you are using an empty SIMATIC Memory Card, for example, by checking in the Windows Explorer that the "SIMATIC.S7S" folder and the "S7_JOB.S7S" file are deleted.
2. Insert the empty SIMATIC Memory Card into the F-CPU.

   If the F-CPU is in RUN, it will change to STOP. The maintenance LED on the F-CPU is flashing to indicate that the program can be copied from the internal load memory to the SIMATIC Memory Card and that the internal load memory is deleted afterward.
3. Use one of these methods to trigger copying from the internal load memory to the SIMATIC Memory Card and subsequent deletion of the internal load memory:
   - Turn the F-CPU off and back on.
   - Switch the F-CPU from STOP to RUN.
   - Execute the "Memory reset" (MRES) function.

   After restart and copying of the program from the internal load memory to the SIMATIC Memory Card and subsequent deletion of the internal load memory, **the STOP/RUN LED (orange) and the maintenance LED must be flashing**. The internal load memory of the F-CPU has been deleted in this case and does no longer store the safety program. The SIMATIC Memory Card is now a program card.
4. Use one of the following methods to evaluate the program card:
   - Turn the F-CPU off and back on.
   - Switch the F-CPU from STOP to RUN.
   - Execute the "Memory reset" (MRES) function.

   The F-CPU restarts and evaluates the program card.

   The F-CPU then enters the startup mode (RUN or STOP) that has been set up for the F-CPU. *(S057)*

## 10.4.9 Transferring the safety program to an S7-1200 F-CPU using a transfer card

**⚠ WARNING**

You must observe the following procedure to ensure that there is no "old" safety program in the internal load memory when you transfer a safety program to an S7-1200 F-CPU using a transfer card:

1. If the internal load memory of the F-CPU has already been deleted, **the STOP/RUN LED (orange) and the maintenance LED must flash for 3 seconds during startup on an F-CPU without SIMATIC Memory Card**. You can skip step 3 in this case.
2. Insert the transfer card into the F-CPU.

   If the F-CPU is in RUN, it will change to STOP. The maintenance LED on the F-CPU is flashing to indicate that the transfer card is being evaluated or that the internal load memory must be deleted.
3. Use one of the following methods to delete the internal load memory:
   - Turn the F-CPU off and back on.
   - Switch the F-CPU from STOP to RUN.
   - Execute the "Memory reset" (MRES) function.

   After restart and deletion of the internal load memory, **the STOP/RUN LED (orange) and the maintenance LED must be flashing**. The internal load memory of the F-CPU has been deleted in this case and does no longer store an "old" safety program.
4. Use one of the following methods to evaluate the transfer card (transfer from the transfer card to the internal load memory):
   - Turn the F-CPU off and back on.
   - Switch the F-CPU from STOP to RUN.
   - Execute the "Memory reset" (MRES) function.

   After restart and evaluation of the SIMATIC Memory Card, the F-CPU copies the safety program to the internal load memory of the F-CPU. Once the copy process is complete, the maintenance LED on the F-CPU is flashing to indicate that you can remove the transfer card.
5. Remove the transfer card from the F-CPU.
6. Use one of the following methods to evaluate the safety program transferred to the internal load memory:
   - Turn the F-CPU off and back on.
   - Switch the F-CPU from STOP to RUN.
   - Execute the "Memory reset" (MRES) function.

   The F-CPU then enters the startup mode (RUN or STOP) that has been set up for the F-CPU. *(S059)*

### 10.4.10 Updating the safety program on an S7-1200 F-CPU using a transfer card

**⚠ WARNING**

If you run an update of the safety program on an S7-1200 F-CPU with the help of a transfer card, you must ensure that the transfer to the internal load memory took place correctly by means of a subsequent program identification. *(S060)*

## 10.5 Comparing Safety Programs

### Compare safety programs as in standard

You can use the *comparison editor* in *STEP 7* for offline-online or offline-offline comparison of safety programs. The procedure is the same as for standard user programs. The contents of F-blocks are also compared for the comparison of safety programs. As a result, an offline-offline comparison can also be used for an acceptance of changes (Page 319).

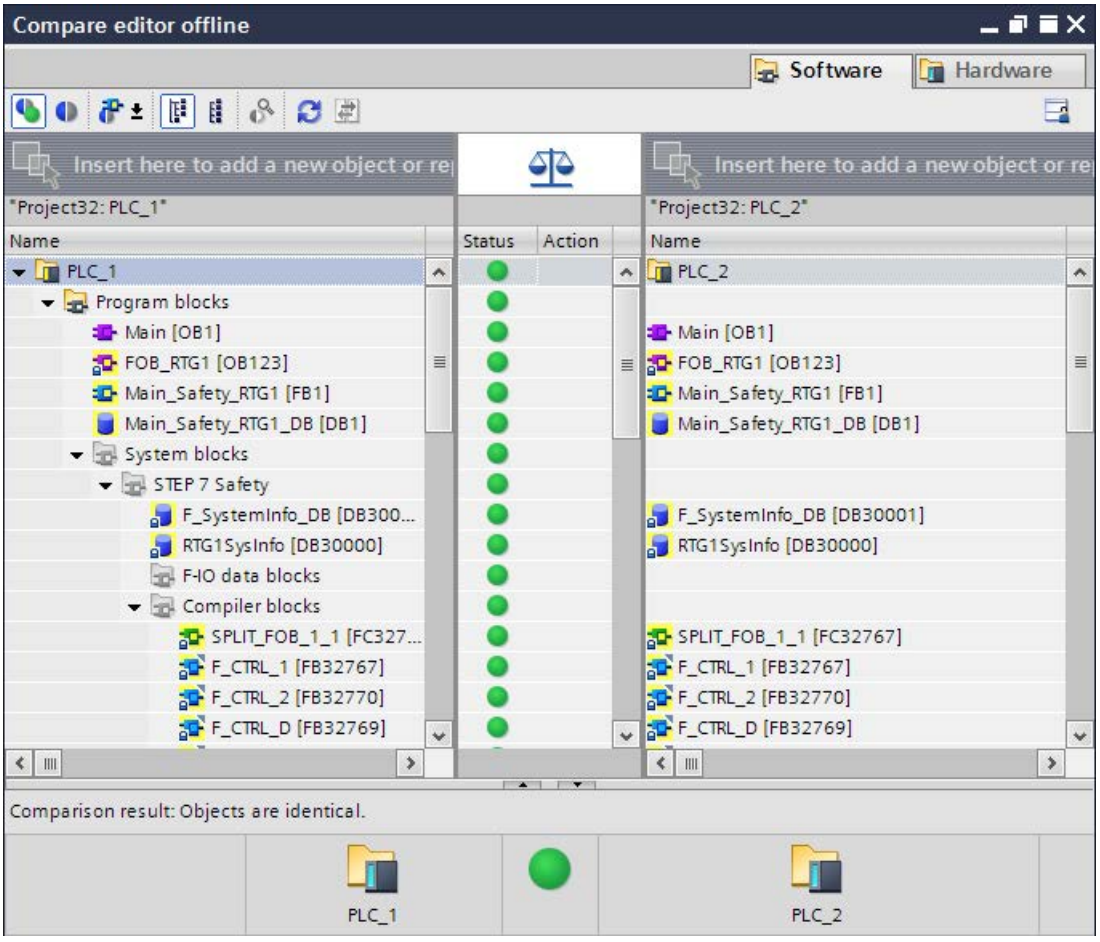
**Note**

During the offline-online comparison, the comparison statuses may occasionally differ between the *comparison editor* and status display in the project tree or *Safety Administration Editor*. The decisive status is the result of the comparison in the *comparison editor*, since this is the only comparison that takes into account the contents of the F-blocks.

## Comparison result for safety programs

The representation of the comparison result corresponds to the representation of *STEP 7*.

If you click the "Program blocks" folder on the left of the comparison editor, you can see the collective F-signature of the safety program displayed under "Comparison result". You also receive information about whether the safety program is consistent.




If you click on an F-block, you can see the respective signatures and interface signatures in addition to the standard information.

**Note**

If you interrupt the connection to the F-CPU during the online/offline comparison, the comparison result will be incorrect.

The meaning of the symbols in the "Status" column is explained below:

Table 10- 1 Offline-online comparison: Status of the comparison and causes

| Symbol | Meaning for safety program | Possible causes |
|---|---|---|
|  | The offline and online versions of the object are identical. | — |
|  | The offline and online versions of the object are different. | The relevant differences are displayed in the row above the detailed comparison. |
|  | Object exists only offline | The block only exists offline. |
|  | Object exists only offline | The block exists only offline. |

Table 10- 2 Offline-offline comparison: Status of the comparison and causes

| Symbol | Meaning for safety program | Possible causes |
|---|---|---|
|  | Both offline versions of the object are identical. | — |
|  | Both offline versions of the object are different | The relevant differences are displayed in the row above the detailed comparison. |
|  | Object available only in safety program 1 | — |
|  | Object available only in safety program 2 | — |

## Comparison filter options

You can use filters in the *comparison editor* to limit the comparison result to the following block groups:

- Compare only F-blocks
- Compare only F-blocks relevant for certification
- Compare only standard blocks

You also have the *STEP 7* filter options "Show only objects with differences" and "Show identical and different objects".

For comparison of safety programs, F-blocks in the "System blocks" folder are also relevant.

## Classification of displayed changes

Regardless of whether you carried out an offline/online or offline/offline-comparison, the following changes could account for the indicated changes to the automatically generated F-blocks:

- Change in the maximum cycle time of F-runtime group and warn cycle time of F-runtime group
- Change in F-parameters of the F-CPU
- Modified version of F-system blocks (S7-1200/1500: Displayed as change of the "F_SystemInfo_DB" block).
- (S7-300/400) Change in the F-runtime group communication, for example, change in the number of a DB for F-runtime group communication
- Change in main safety block, F-FB, F-FC, F-DB
- Change of the hardware configuration for the F-I/O addressed in the safety program

It is possible that a block is displayed as changed, but no changes are displayed in the detailed comparison of the block content. This is not a display problem but means that changes in the hardware configuration or the tag table, for example, have an effect on this block. Test this block in case of doubt.

## Printing result of comparison

The comparison result can be printed via "Project > Print" in the menu bar or the print button in the toolbar. Select "Print objects/area" "All" and "Properties" "All".

# 10.6 Printing project data

## Printing

You can print all important project data of the hardware configuration of the F-I/O and safety program. You obtain a "safety summary" that, alongside the documentation, serves as a basis for testing the correctness of the individual components of the system. Correctness is a prerequisite for system acceptance.

The collective F-signature specifications in the footer of the printout ensure that the printout is explicitly associated with a safety program.

## Procedure for creating a safety summary

To create a safety summary, follow these steps:

1. In the project tree, select the *Safety Administration Editor* of the F-CPU whose safety summary you want to create.
2. Select "Print" in the shortcut menu or "Project > Print" in the menu bar or the print button in the toolbar.

   In the displayed dialog, you can make layout settings for the printout and specify the scope of the printout (all/subset), among other things.
3. Select the "All" option, if the F-blocks and F-compliant PLC data types are to be shown in the printout. This is necessary, for example, to document the program code for the acceptance(see Acceptance of system (Page 308)). Select the "Compact" option to exclude the source code from the printout.
4. Click the "Print" button.

As a result, you receive the safety summary for the F-CPU.

## Safety summary

The safety summary provides documentation of the safety program and provides support for the acceptance of the system.

### Contents of the summary in overview

The topics that are considered in the summary are summarized in the following:

- General information on program identification, software versions, access protection, settings (from the "Settings" work area of the *Safety Administration Editor*)
- System library elements used in safety program (from the "Instructions" task card and F-system blocks) along with their versions
- Information about the F-runtime groups (F-monitoring time, cycle time warning limit, max. cycle time, F-blocks and their names)
- Listing of the F-blocks within the "Program blocks" folder (name, function, associated F-runtime group, signature)
- (S7-1200, S7-1500) Listing of F-compliant PLC data types
- Data from the standard user program that are evaluated in the safety program
- Parameters of safety-related CPU-CPU communication
- (S7-300, S7-400) Absolute addresses and names of the F-shared DB tags that can be accessed from the standard user program
- Information on hardware (used F-I/O, CPU version, addresses)
- Information on the printout (print date, number of pages)

### Footer of the printouts

On the basis of the footer of the printout, you can find out:

- whether you printed out the "correct" project (specification of the project name and file path)
- whether the printout is consistent and belongs to the same safety program and the same version (the same collective F-signature in the footer of every page means that the printout belongs to the safety program with this collective F-signature).

The footer is added to the source code of the F-blocks only if the "All" option was selected for the safety summary.

If F-blocks are printed by other means, the footer is omitted, and you can no longer easily identify whether the block printout belongs to the current safety program version.

### Printing a migrated project

You can only print a safety summary for a project migrated from *S7 Distributed Safety* V5.4 SP5 if the project was compiled with *STEP 7 Safety Advanced* and the new program structure for safety programs (main safety block) has therefore been applied. Otherwise, the printout is not possible and you will receive a corresponding error message.

We recommend that you print out your project in *S7 Distributed Safety V5.4 SP5* before the migration.

# 10.7 Testing the safety program

## 10.7.1 Overview of Testing the Safety Program

### Complete function test or test of changes

After creating a safety program, you must carry out a complete function test in accordance with your automation task.

For changes made to a safety program that has already undergone a complete function test, only the changes need be tested.

### Monitoring

In general, all read-only test functions (such as tag monitoring) are also available for safety programs and in safety mode.

### Modifying

Modifying safety program data and write accesses to the safety program are possible only in a limited way and in disabled safety mode.

### Simulation with *S7-PLCSIM* (S7-300, S7-400, S7-1500)

You can test the safety program using *S7-PLCSIM*. You use *S7-PLCSIM* in the same way as for standard user programs.

You start the simulation with *S7-PLCSIM* using menu item "Online > Simulation > Start".

## 10.7.2 Disabling safety mode

### Introduction

The safety program generally runs in the F-CPU in safety mode. This means that all fault control measures are activated. The safety program cannot be modified during operation (in RUN mode) in safety mode. You must disable safety mode of the safety program to, for example, modify tags in the safety program in RUN mode. Safety mode remains deactivated until the F-CPU is next switched from STOP to RUN mode.

### Rules for disabling safety mode

**⚠ WARNING**

Because changes to the safety program can be made in RUN mode when safety mode is deactivated, you must take the following into account:

- Disabling safety mode is intended for test purposes, commissioning, etc. Whenever safety mode is deactivated, the safety of the system must be ensured by other organizational measures, such as monitored operation, manual safety shutdown, and access restrictions to certain areas.
- Disabling of safety mode must be displayed.

  Use the MODE tag in the F-shared DB ("F_GLOBDB".MODE) for S7-300/400 F-CPUs or in the F-runtime group information DB (e.g RTG1SysInfo.F_SYSINFO.MODE) for S7-1200/1500 F-CPUs, which you can evaluate to read the operating mode (1 = Disabled safety mode). This means not only is the disabled safety mode displayed on the programming device or PC in the dialog box for disabling safety mode, but it can also be indicated by means of an indicator light controlled by the standard user program or a message to an HMI system generated by evaluating the above-mentioned "Disabled safety mode" variable in the F-shared DB.
- It must be possible to verify that safety mode has been disabled. A log is required, if possible by recording messages to the HMI system, but if necessary, through organizational measures. In addition, it is recommended that disabling of safety mode be indicated on the HMI system.
- Safety mode is disabled F-CPU-wide. In spite of this, you must take the following into account for safety-related CPU-CPU communication: If the F-CPU with the SENDDP or SENDS7 instruction is in disabled safety mode, you can no longer assume that the data sent by this F-CPU are generated safely. You must then implement organizational measures such as operation monitoring and manual safety shutdown to ensure safety in those portions of the system that are affected by the sent data. Alternatively, you must output fail-safe values instead of the received data in the F-CPU with the instruction RCVDP or RCVS7 (Page 646) or SENDMODE by evaluating SENDMODE. *(S027)*

### Procedure for disabling safety mode

To disable safety mode, follow these steps:

1. Open the *Safety Administration Editor* of the corresponding F-CPU.
2. Open the work area "General (Page 67)" in the area navigation.
3. Check to see whether the safety mode status is displayed as activated.

   If so, continue with the next step; if not, stop the process, because safety mode is already disabled or cannot be disabled.
4. Click the "Disable safety mode" button.
5. Enter the password for the online safety program.

   If you enter the correct password, another prompt will appear, which also contains the collective F-signature in the F-CPU. Check to see whether this is the collective F-signature you expected. If there is a match, acknowledge the dialog.

   Safety mode is then disabled.

   If the password is not valid, safety mode is not deactivated and remains active.

   (S7-300, S7-400) When individual F-blocks are downloaded, the condition "Disable safety mode" is listed automatically in the "Load preview" dialog. For this reason, it is not necessary to explicitly disable safety mode before every F-block download.

**Note**

If the collective F-signature or the passwords do not agree for the safety program online and offline, this means:

- The offline safety program was modified after the last downloading, or
- An incorrect F-CPU was addressed. Check the latter based on the online collective F-signature.

### Enabling safety mode

**Note**

To enable safety mode, the F-CPU must be switched from STOP to RUN mode.

Switching the F-CPU from STOP to RUN mode always enables safety mode, even if the safety program has been modified or is not consistent. The MODE tag in the F-shared DB for S7-300/400 F-CPUs or F-runtime group information DB is set to "0" for S7-1200/1500 F-CPUs.

If you have changed your safety program, but have not recompiled and downloaded it, the F-CPU can revert to STOP mode.

### Evaluating safety mode/disabled safety mode

If you wish to evaluate safety mode/disabled safety mode in the safety program, you can evaluate the "MODE" tag in the F-shared DB for S7-300/400 F-CPUs or F-runtime group information DB for S7-1200/1500 F-CPUs (1 = Disabled safety mode). You use fully qualified access to access this tag (e.g. "F_GLOBDB".MODE or RTG1SysInfo.MODE).

You can use this evaluation, for example, to passivate F-I/O when the safety program is in disabled safety mode. To do so, assign the "MODE" tag in the F-shared DB or F-runtime group information DB to all "PASS_ON" tags in the F-I/O DBs of the F-I/O that you wish to passivate.

**⚠ WARNING**

When the safety program is in disabled safety mode, the "MODE" tag in the F-shared DB or F-runtime group information DB is also evaluated in disabled safety mode.

Even if the F-I/O are passivated in disabled safety mode as a result of evaluation of the "MODE" tag, system safety must be ensured in disabled safety mode through other organizational measures, such as operation monitoring and manual safety shutdown. *(S028)*

### See also

F-shared DB (S7-300, S7-400) (Page 114)

F-runtime group information DB (S7-1200, S7-1500) (Page 115)

## 10.7.3 Testing the safety program

### Introduction

In disabled safety mode, certain fault control measures of the safety program are deactivated to enable online changes to be made to the safety program in RUN mode. In this way, safety program data can be changed using standard tools from *STEP 7* (watch table, monitoring in the *program editor*).

### Changing the safety program data by modifying tags

In addition to data in the standard user program, which can always be modified, you can modify the following data of a safety program in disabled safety mode:

- Process image of the F-I/O (channel values and value status (S7-1200, S7-1500))
- F-DBs (except DB for F-runtime group communication), instance DBs of F-FBs
- F-I/O DBs (for permitted signals, see F-I/O DB (Page 129))

**Note**

F-I/O can only be modified in RUN mode of the F-CPU. You must allocate a separate row in the watch table for each channel value and value status (S7-1200, S7-1500) to be modified; this means, for example, that digital channels of data type BOOL cannot be modified on a byte-by-byte or word-by-word basis.

You can modify a maximum of 5 inputs/outputs from a watch table for S7-300/400 F-CPUs. You can use more than one watch table.

You cannot modify a configured F-I/O from which neither a channel value or a value status (S7-1200, S7-1500), nor any tag from the associated F-I/O DB has been used. In your safety program, you should therefore always use at least one tag from the associated F-I/O DB or at least one channel value or value status (S7-1200, S7-1500) of the F-I/O to be modified.

As a trigger point, you must set "Begin scan cycle" or "End scan cycle", either "permanently" or "once". However, note that regardless of the trigger point setting, requests to modify inputs (PII) of F-I/O always become effective before the main safety block is executed and requests to modify outputs (PIQ) always become effective after execution of the main safety block.

For inputs (PII), modify requests take priority over fail-safe value output, while for outputs (PIQ), fail-safe value output takes priority over modify requests. For outputs (channels) that are not activated in the properties for the F-I/O, modify requests affect the PIQ only, and not the F-I/O.

**Note**

The following applies for S7-1200/1500 F-CPUs:

To avoid invalid combinations of channel value and status value:

- The value status is set by the F-system automatically to 1 when setting a channel value to a value <> fail-safe value 0
- The fail-safe value 0 is automatically output when setting the value status to 0 for the associated channel value

**⚠ WARNING**

Constant modify requests for F-I/O can continue to remain active in the following cases:

- The connection between the programming device or PC and F-CPU is interrupted (e.g. because the bus cable is unplugged)
- if the watch table no longer responds

To delete constant modify requests in such cases, you must

- Switch the relevant F-CPU from STOP to RUN when the connection to the programming device or PC is interrupted

  or
- Perform a memory reset on the relevant F-CPU. *(S029)*

### Wiring test using tag table

You can carry out a wiring test for an input by changing an input signal and verifying whether or not the new value arrives in the PII.

You can carry out a wiring test for an output by changing the output with the Modify function and verifying whether the required actuator responds.

For the wiring test (both for an input and output), note that a safety program must be running on the F-CPU in which at least one channel of the F-I/O to be modified or one tag from the associated F-I/O DB has been used.

For F-I/O that can also be operated as standard I/O (e.g., S7-300 fail-safe signal modules), you can also carry out the wiring test for outputs using the Modify function in STOP mode by operating the F-I/O as standard I/O rather than in safety mode. When doing so, you must comply with the other rules for testing.

**Note**

A modify function controlled by the F-system is only possible if the *STEP 7 Safety* optional package is installed. Modification of tags by means of an HMI system as well as modification without an installed *STEP 7 Safety* optional package can cause the F-CPU to go into STOP.

The test functions are selected using the standard tools from *STEP 7* (*program editor*, watch table). An attempt to modify a safety program in safety mode is rejected with a corresponding error message, or a dialog box for disabling safety mode is provided. In certain circumstances, a modify request can cause the F-CPU to go to STOP mode.

### Opening and changing F-blocks

Opening of an F-block online in the F-CPU is only possible with write protection in the *program editor*; that is, you cannot change an F-block directly in the F-CPU, even if safety mode is deactivated. After a successful password prompt, the F-block switches automatically to offline mode while opening (block icon is labeled accordingly in the project tree) and can be changed by you.

You must then compile the F-block and download it to the F-CPU. To do so, follow the procedure as outlined in "Downloading the Safety Program (Page 268)".

### Downloading F-blocks for testing purposes (S7-300, S7-400)

To execute a download of F-blocks for testing purposes, only the F-blocks to be tested may be selected and "Stop module" has to be set to "No action" in the "Load preview" dialog.

### Modifying values in F-DBs

Values in F-DBs can only be modified online in the F-CPU. If the value is also to be changed offline, you must do this by editing the start value offline and compiling the safety program.

## Additional rules for testing

- Forcing is not possible for F-I/O.
- Setting breakpoints in the standard user program will cause the following errors in the safety program:
  - Expiration of F-cycle time monitoring
  - Error during communication with the F-I/O
  - Error during safety-related CPU-CPU communication
  - Internal CPU faults

  If you nevertheless want to use breakpoints for testing, you must first disable safety mode (Page 293). This will result in the following errors:
  - Error during communication with the F-I/O
  - Error during safety-related CPU-CPU communication
- Changes in the configuration of F-I/O or safety-related CPU-CPU communication can only be tested after the hardware configuration has been saved and downloaded, and after the safety program has been compiled and downloaded to the F-CPU.

## Procedure for monitoring and modifying the safety program

Monitor or modify the required F-data and/or F-I/O from an open watch table or from the *program editor* (program status) (for the procedure, see *Help on STEP 7*, "Testing user program").

1. For modifying, deactivate the safety mode (Page 293) in the automatically shown dialog.
2. Terminate existing modify requests after testing is complete before activating safety mode.

If the safety program does not behave as you wish during testing, you have the option of modifying the safety program in RUN mode (Page 302) and immediately continuing testing until the safety program behaves according to your requirements.

## 10.7.4 Testing the safety program with S7-PLCSIM (S7-300, S7-400, S7-1500)

You can test your safety program analog to your standard program on a simulated CPU with *S7-PLCSIM* and without the need for hardware.

You use *S7-PLCSIM* for SIMATIC Safety F-systems as you would for S7 standard systems. Note the following special features:

### Safety mode/disabled safety mode

We recommend that you test your safety program in safety mode to detect whether the F-CPU goes into STOP as early as in the test phase of your safety program in *S7-PLCSIM* as a result of, for example that the results of instructions were outside the permitted range for the data type.

The following simulations can be run in *S7-PLCSIM*, just as on an actual F-CPU, in disabled safety mode only.

- Modifying tags in F-DBs and F-I/O DBs.
- (S7-1500) Modifying inputs (value status) for F-output I/O (see below)

The CPU can go to STOP mode in *S7-PLCSIM* if this is disregarded. The cause of the diagnostic event is entered in the diagnostic buffer of the CPU.

(S7-1500) To prevent unintentional modification of tags in F-DBs and F-I/O DBs in safety mode, we recommend that you do not select the "Activate/deactivate modification of non-inputs" button in *S7-PLCSIM*.

### Monitoring the F-cycle time

The F-system monitors the F-cycle time of the F-runtime group in safety mode. Due to different cycle times of the CPU in *S7-PLCSIM* and the real F-CPU, monitoring of the F-cycle time can be triggered on the CPU in *S7-PLCSIM* and the CPU goes into STOP mode. A corresponding diagnostic event is entered in the diagnostic buffer of the CPU.

In this case you must increase the "Maximum cycle time of the F-runtime group" for all affected F-runtime groups in the SAE (for example, to the maximum time that can be set) during simulation in *S7-PLCSIM*.

## Input simulation of F-I/O

Modification of inputs (channel values) in *S7-PLCSIM:*

You modify inputs (channel values) of F-I/O as you would inputs (channel values) of standard I/O in *S7-PLCSIM*.

Modification of inputs (value status) in *S7-PLCSIM:*

(S7-1500) By modifying inputs (value status) of F-I/O you can simulate the incoming and outgoing F-I/O/channel faults. Keep in mind the following notes/restrictions:

- To realistically simulate the behavior of the F-I/O, you must note the connection between channel value and value status on the real F-I/O. The combination value status = 0 and channel value <> fail-safe value (0) is invalid and can result in the simulation deviating from the behavior of the real F-CPU.
- During the transition from "STOP" to "RUN" of the CPU in *S7-PLCSIM*, all F-I/O inputs (value status) are initialized with 1. This means you can start with the modification of inputs (channel values) without simulation of the inputs (value status).
- The modification of inputs (value status) in *S7-PLCSIM* does not have an effect on the tags QBAD and PASS_OUT in the F-I/O DB. Note that with real F-I/O QBAD and PASS_OUT can be 1 as soon as the value status is 0 for at least once channel of the F-I/O. (see tags of the F-I/O DB: PASS_OUT/QBAD/QBAD_I_xx/QBAD_O_xx and value status (Page 135)).
- For F-I/O configured with "Behavior after channel fault" = "Passivation of the complete F-I/O", use the tag PASS_ON in the F-I/O DB for simulation of the passivation of the complete F-I/O for F-I/O / channel faults. If you only passivate individual inputs (channel value including value status) for the simulation, the behavior of the simulation will deviate from the real F-CPU.
- You can also use the PASS_ON tag in the F-I/O DB for F-I/O without value status to simulate the passivation of the entire F-I/O in case of F-I/O or channel faults.
- You must modify the inputs (channel values) to $7FFF_H$ (for overflow) or $8000_H$ (for underflow) to simulate an F-I/O/channel fault of the SM 336; AI 6 x 13Bit or the SM 336; F-AI 6 x 0/4...20 mA HART with configuration "Behavior after channel fault" = "Passivate channel".
- Inputs (value status) for F-output I/O may only be modified in disabled safety mode. The CPU can go to STOP mode in *S7-PLCSIM* if this is disregarded. The cause of the diagnostic event is entered in the diagnostic buffer of the CPU.
- For F-I/O which does not support the RIOforFA-Safety profile, you must run a user acknowledgment with a positive edge at the ACK_REI tag of the F-I/O DB as with a real F-I/O for reintegration after the value status has changed from 0 to 1 or when the channel value has changed from $7FFF_H$/$8000_H$ to unequal $7FFF_H$/$8000_H$ (see above) when ACK_NEC = 1 of the F-I/O DB. Reintegration takes place automatically in all other cases possibly deviating from the real F-I/O.

### Update times

Keep in mind that the status of the inputs (channel values or value status (S7-1500)) which you are monitoring in the SIM table in *S7-PLCSIM* is only identical to the status being processed in the F-program if there is not passivation of the associated F-I/O.

With passivation of the F-I/O, the F-program operates with fail-safe values (channel value and value status (S7-1500) =0).

### Instructions for communication between F-CPUs

You cannot simulate communication between F-CPUs with the SENDDP and RCVDP instructions in *S7-PLCSIM*. You can, however, use the SENDDP and RCVDP instructions together with *S7-PLCSIM*.

During simulation in *S7-PLCSIM*, the RCVDP instruction outputs the fail-safe values pending at its inputs SUBBO_xx and SUBI_xx or alternatively SUBDI_00. The SENDDP and RCVDP instructions signal this with 1 at output SUBS_ON.

(S7300/400) You cannot simulate communication between F-CPUs with the SENDS7 and RCVS7 instructions in *S7-PLCSIM*. You can, however, use the SENDS7 and RCVS7 instructions together with *S7-PLCSIM*.

During simulation in *S7-PLCSIM*, the RCVS7 instruction outputs the initial values specified in the communication DB as fail-safe values. The SENDS7 and RCVS7 instructions signal this with 1 at output SUBS_ON.

### Inconsistent safety program (S7-1500)

If the CPU goes into STOP in *S7-PLCSIM* with the diagnostic entry "Safety program: inconsistent", the CPU is not initialized correctly in *S7-PLCSIM* yet. Perform a memory reset of the CPU in *S7-PLCSIM* and download the program once again to the CPU in *S7-PLCSIM*.

## 10.7.5 Changing the safety program in RUN mode (S7-300, S7-400)

### Introduction

Changes to the safety program during operation (in RUN mode) can only be made in disabled safety mode (Page 293). You make changes to F-blocks offline in the *program editor* in the same way as for a standard program. F-blocks cannot be changed online.

**Note**

If you do **not** want to make changes to the safety program during operation, see Creating F-blocks in FBD/LAD (Page 118).

### Procedure for changing the safety program in RUN mode

To change the safety program, follow these steps:

1. Change the main safety block or F-FB and its associated instance DB, F-FC, or F-DB in the *Program editor*.
2. Download the changed F-block(s) to the F-CPU (for procedure, see Downloading the Safety Program (Page 268)). The entire program is then automatically compiled.
3. If safety mode is active, the "Load preview" dialog will prompt you to deactivate it and to enter the password for the safety program.

**Note**

The following applies for S7-300/400 F-CPUs:

When downloading in disabled safety mode, you can only download the fail-safe blocks created by you (main safety blocks, F-FB, F-FC, or F-DB), F-application blocks, or standard blocks and their associated instance DBs. If you download automatically added F-blocks (F-SBs or automatically generated F-blocks and associated instance DBs, F-shared DB), the F-CPU can go to STOP mode or safety mode can be activated.

Therefore, always select individual blocks only when downloading in disabled safety mode.

### Sequence for downloading changes

Changes in the safety program in RUN mode when safety mode is deactivated can cause e.g., the status of an actuator to change as a result of program changes.

After changes, start by downloading the safety program and then the function of the standard user program monitored by the safety program.

### Restrictions on safety-related CPU-CPU communication

During operation (in RUN mode), you cannot establish new safety-related CPU-CPU communication by means of new SENDDP/RCVDP or SENDS7/RCVS7 instructions.

To establish new safety-related CPU-CPU communication you must always download the relevant safety program consistently to the F-CPU while in STOP mode after inserting a new SENDDP, SENDS7, RCVDP, or RCVS7 instruction.

### Restrictions for F-runtime group communication (S7-300, S7-400)

You cannot make any changes to the safety-related communication between F-runtime groups in RUN mode. This means that you cannot assign, delete, or change any DBs for F-runtime group communication of an F-runtime group.

Following changes in the F-runtime group communication, you must always download the safety program consistently to the F-CPU while in STOP mode.

### Restrictions for F-I/O access (S7-300, S7-400)

If during operation (in RUN mode), you insert an F-I/O access to an F-I/O of which no single channel value or tag from the associated F-I/O DB has yet been used in the safety program, the F-I/O access only becomes effective when the safety program is downloaded consistently to the F-CPU.

### Changes in the standard user program (S7-300, S7-400)

You can download changes in the standard user program when the F-CPU is in RUN mode, regardless of whether safety mode is enabled or disabled.

**⚠ WARNING**

(S7-300, S7-400) In safety mode, access with the CPU password must not be authorized during changes to the standard user program as this would allow changes to the safety program. To rule out this possibility, you must configure **protection level "Write protection for fail-safe blocks"** and configure a password for the F-CPU. If only **one person** is authorized to change the standard user program **and** the safety program, the protection level "Write protection" or "Read/write protection" should be configured so that other persons have only limited access or no access at all to the entire user program (standard and safety programs). (See also Access protection (Page 76)) *(S001)*

### Changes in the standard user program (S7-1200, S7-1500)

You can download changes in the standard user program when the F-CPU is in RUN mode, regardless of whether safety mode is enabled or disabled.

**⚠ WARNING**

(S7-1200, S7-1500) In safety mode, the safety program must be password-protected. You therefore need to configure at least the protection level "Full access (no protection)". This protection level only allows full access to the standard user program, not to F-blocks.

If you select a higher protection level, for example to protect the standard user program, you must assign an additional password for "Full access (no protection)". *(S041)*

### Procedure for applying changes to the safety program (S7-300, S7-400)

If you download individual F-blocks to the F-CPU during operation (in RUN mode), the F-system blocks (F-SBs) and the automatically generated F-blocks are neither updated nor downloaded, resulting in an inconsistent safety program in the F-CPU. Use the following procedure to apply changes to the safety program:

1. Download the safety program consistently to the F-CPU, and activate safety mode by switching the F-CPU from STOP to RUN mode (for procedure, see Downloading the Safety Program (Page 268)).
2. Follow the steps described in Acceptance of Changes (Page 319).

## 10.7.6 Deleting the safety program

### Deleting individual F-blocks

To delete an F-block, follow the same procedure as in *STEP 7*.

### Deleting an F-runtime group

See Deleting an F-runtime group (Page 116)

(S7-300, S7-400) Remove all calls that you have used to call the safety program (Main_Safety).

### Deleting the entire safety program for F-CPUs *with* plugged memory card (SIMATIC Micro memory card)

To delete an entire safety program, proceed as follows:

1. Delete all F-blocks (shown with yellow symbol) in the project tree.
2. Select the F-CPU in the *hardware and network editor* and clear the "F-capability activated" option in the properties of the F-CPU.
3. Compile the hardware configuration.

   The offline project no longer contains a safety program.
4. Format the SIMATIC memory card online in the F-CPU as usual in the standard program.
5. Select the menu command "Project> SIMATIC Card Reader> Show SIMATIC Card Reader" in the menu bar.
6. Open the "SIMATIC Card Reader" folder. You can now access the memory Card and can delete the safety program.

   You can then download the offline standard user program to the F-CPU.

### Deleting the entire safety program for F-CPUs *with* plugged memory card (SIMATIC Micro memory card or flash card)

To delete an entire safety program, proceed as follows:

1. Delete all F-blocks (shown with yellow symbol) in the project tree.
2. Select the F-CPU in the *hardware and network editor* and clear the "F-capability activated" option in the properties of the F-CPU.
3. Clear the cyclic interrupt OB 3x for each F-runtime group.
4. Compile the hardware configuration.

   The offline project no longer contains a safety program.
5. To delete a safety program on a Memory Card (SIMATIC Micro Memory Card or Flash Card), insert the Memory Card (SIMATIC Micro Memory Card or Flash Card) in the programming device or PC or in a SIMATIC USB prommer.
6. Select the menu command "Project> SIMATIC Card Reader> Show SIMATIC Card Reader" in the menu bar.
7. Open the "SIMATIC Card Reader" folder. You can now access the memory Card and can delete the safety program.

   You can then download the offline standard user program to the F-CPU.

### Deleting the entire safety program for F-CPUs *without* plugged flash card

To delete an entire safety program, proceed as follows:

1. Delete all F-blocks (shown with yellow symbol) in the project tree.
2. Select the F-CPU in the *hardware and network editor* and clear the "F-capability activated" option in the properties of the F-CPU.
3. Compile the hardware configuration.

   The offline project no longer contains a safety program.

By performing a memory reset on the F-CPU (in the "Online tools" task card of the F-CPU), you can delete the safety program.

## 10.8 F-change history

Enable the logging of changes to the safety program by using the option "Enable F-change history" in the *Safety Administration Editor*. The F-change history behaves like the standard change history.

An F-change history is created for each F-CPU in the project navigation under "Common data/logs".

The following is logged in the F-change history:

- collective F-signature
- User name
- Compile time stamp
- Download of the safety program with time stamp
- Compiled F-blocks with signature and time stamp

The F-change history can contain a maximum of 5000 entries per F-CPU. When the 5000 entries are exceeded, a new F-change history is created using the name pattern "F-change history <CPU name> YYYY-MM-DD hh:mm:ss".

| NOTICE |
| --- |
| The connection between the F-CPU and the associated F-change history is made through the name of the F-change history.<br>Therefore, do not rename the F-CPU and the F-change history. If you rename the F-CPU or the F-change history, a new F-change history with the changed name is started. |

# System Acceptance 11

## 11.1 Overview of System Acceptance

### Introduction

When a system undergoes an acceptance, all standards relevant to the specific application must be met. This also applies to systems that are not "subject to acceptance". For the acceptance, you must consider the requirements in the Certification Report (http://support.automation.siemens.com/WW/view/en/49368678/134200).

As a general rule, the acceptance of an F-System is performed by an independent expert.

Observe all warnings in this manual.

**WARNING**

The F-CPU and F-I/O must be configured in the hardware and network editor of the TIA Portal as described in this documentation. F-blocks must be created with the program editor of the TIA Portal as described in this documentation. For system acceptance, you have to use the safety summary created according to this documentation. Any other procedures are not permitted. *(S056)*

### Proof of correctness of components

In order for a system acceptance to be granted, you must recognize and document the correctness of the individual components. For documentation of the component properties, you must create a safety summary. First compile the safety program with "Compile > Software (rebuild all blocks)"

The following characteristics must be covered:

- Correctness of the safety program including hardware configuration (including testing) (Page 309)
- Completeness of the safety summary (Page 310)
- Compliance of the system library elements used in the safety program with the TÜV certificate (Page 311)
- Completeness and correctness of the hardware configuration (Page 312)
- Correctness of the communication configuration (Page 316)
- Consistency of the online safety program (Page 317)
- Other characteristics (Page 318) such as software version, use of data from the standard user program

After the acceptance, you should archive all relevant documents and also the project data so as to make the accepted project available as a reference for a subsequent acceptance.

### Safety summary

The safety summary (Page 290) is the project documentation required for acceptance of the system.

## 11.2 Correctness of the safety program including hardware configuration (including testing)

### Verification/function test

Already during the creation, you will test (Page 292) your safety program and the associated hardware configuration. You must carry out tests with regard to the specification of your safety functions and document them before you seek acceptance for the system.

To allow you to perform a code review of your safety program and document the accepted program code, the source code of all F-blocks is printed as a part of the safety summary (Page 290), provided you have selected the "All" property for the printout.

The correct function of the safety program must be guaranteed by complete function tests before it may be used productively. You should archive the test reports along with the safety summary and the acceptance documents.

Times, for example monitoring times (Page 661) and delay times, are extremely hard to verify with functional tests (Page 274). You should check these times selectively to determine whether they are dimensioned correctly, for example, on the basis of the safety summary.

Some of these times are itemized specially in the safety summary, for example, the F-monitoring time (for communication between F-CPU and F-I/O) and the monitoring time of the safety-related CPU-CPU communication (parameter TIMEOUT). For the approximate determination of these monitoring times, the Excel file for response time calculation is available on the Internet (http://support.automation.siemens.com/WW/view/en/49368678/133100).

### Consistency of the safety program

Check in the "General information" section of the safety summary to determine whether the safety program was recognized as "consistent".

This is the case for S7-300/400 F-CPUs if the following signatures are also identical:

- Collective F-signature ("General information" section, "Collective F-signature")
- "Signature of F-blocks with F-attribute" ("General information" section, "Current compilation")

Consistency of the safety program is required for the acceptance. If the signatures are not identical, you should recompile the safety program and regenerate the safety summary.

### Password protection

Verify that a password was assigned for both the safety program and the F-CPU, if other organizational measures for access protection of the safety program were not taken.

For information about this, refer to section "General information" under "Access protection".

### Additional notes

If necessary, review the "Notes" in section "General information", which provides additional notes on the safety program that must be observed or errors that must be eliminated.

## 11.3 Completeness of the safety summary

### Introduction

If your safety program status including hardware configuration is ready for acceptance, you must carry out and document additional checks on the basis of the safety summary. The safety summary must be complete and belong to the safety program undergoing acceptance.

### Procedure for creation of safety summary

To generate the safety summary, follow the procedure described in Printing project data (Page 290).

In so doing, use the "All" property in order to include the source code of your F-blocks in the printout.

### Checking the safety summary for completeness

If you want to use an existing summary, whose completeness is not exactly known, you must check to determine whether the same collective F-signature is contained in the footer on all pages of the printout. This allows you to prove that all printed sheets belong to the same project.

In section "Supplementary information", you can find the number of pages in the safety summary, among other things. With this, you can prove that all pages of the safety summary are printed.

If you created the safety summary with the "All" option, the source code of all F-blocks will also be printed. The printout of this source code also contains the footer to enable you to easily assign the source code to a particular safety summary.

### Association with the safety program

In the "General information" section of the safety summary, check whether the collective F-signature corresponds to the collective F-signature of the safety program to be accepted in the work area of the Safety Administration Editor under "*General*". If they are not the same, then the summary and safety program do not match.

# 11.4 Compliance of the system library elements used in the safety program with the TÜV certificate

## Introduction

The *STEP 7 Safety* optional package contains LAD/FBD instructions and F-system blocks that have been created and tested by SIEMENS and certified by TÜV for use in programming your safety program.

To allow you to check whether the instructions and F-system blocks used correspond to the TÜV certificate and with the versions you have created, these instructions are listed in the safety summary under "System library elements used in the safety program" along with their versions, signatures and initial value signatures (S7-300, S7-400) or versions (S7-1200, S7-1500).

## Procedure

To check, download the current Annex 1 of the report for the TÜV certificate "SIMATIC Safety" from the Internet (http://support.automation.siemens.com/WW/view/en/49368678/134200).

**⚠ WARNING**

- The versions of the system library elements listed in the safety summary must match the versions in Annex 1.
- The versions of the system library elements listed in the safety summary must match the safety requirements of your application.

  Keep in mind possible differences in functionality of different versions, see chapters "Settings" area (Page 71) and Instructions - LAD (Page 329) or Instructions - FBD (Page 494).
- (S7-300, S7-400) The signatures and initial value signatures of the system library elements listed in the safety summary must match the signatures and initial value signatures in Annex 1. *(S054)*

In case of discrepancies, recheck whether you have the correct versions.

# 11.5 Completeness and correctness of the hardware configuration

## Introduction

The hardware configuration is an essential component of the project to be accepted. With the configuration of the hardware, you have set properties that can influence the safety of signals. You must document these settings in detail with the safety summary to prove that you fulfill the safety requirements for your application.

The section "Hardware configuration of F-I/O" is available in the safety summary for this. This section consists of several tables:

- A table with information about the F-CPU and the ranges of the F-destination addresses used and the "Basis for PROFIsafe addresses" parameter of the F-CPU.
- An overview table with the F-I/O used.
- A table for each F-I/O with detailed specifications, for example, the configured parameter values.

**Note**

Note that you will find F-I/O that you address via safety-related I-slave-slave communication in the safety summary of the I-slave and not in the safety summary of the assigned DP master.

## Procedure for checking that the hardware configuration is complete

Check whether the safety summary contains the complete configured F-I/O.

## Procedure for checking the correctness of the hardware configuration

To check the hardware configuration for correctness, proceed as follows:

1. Check in the "F-hardware configuration" section to verify the uniqueness of the PROFIsafe addresses.

   Check in the section "F-hardware configuration", if the "Basis for PROFIsafe addresses" parameter of the individual F-CPUs differs.

   Please see PROFIsafe addresses for F-I/O of PROFIsafe address type 1 (Page 52) and PROFIsafe addresses for F-I/O of PROFIsafe address type 2 (Page 54).

   Please note the following for F-I/Os of PROFIsafe address type 1:

   **⚠ WARNING**

   F-I/Os of PROFIsafe address type 1 are uniquely addressed by their F-destination address (e.g. with the switch setting on the address switch).

   The following rules ensure the uniqueness of the F-destination addresses.

   The F-destination address (and therefore also the switch setting on the address switch) of the F-I/O must be unique network-wide* and CPU-wide** (system-wide) **for the entire** F-I/O. The F-I/O of PROFIsafe address type 2 must also be considered. *(S051)*

   * A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

   ** "CPU-wide" means all F-I/Os assigned to an F-CPU: central F-I/O of this F-CPU as well as F-I/Os for which the F-CPU is DP master/IO controller. An F-I/O that is addressed using I-slave-slave communication is assigned to the F-CPU of the I-slave and not to the F-CPU of the DP master / IO controller.

   Please note the following for F-I/Os of PROFIsafe address type 2:

   **⚠ WARNING**

   F-I/O of PROFIsafe address type 2 is uniquely addressed using a combination of F-source address ("Basis for PROFIsafe addresses of the assigned F-CPU" parameter) and F-destination address.

   The combination of F-source address and F-destination address for each F-I/O must be unique network-wide* and CPU-wide** (system-wide). In addition, the F-destination address may not be occupied by F-I/O of PROFIsafe address type 1.

   To ensure that addresses are unique across F-CPUs for supported configurations (Page 50), you need to ensure that the "Basis for PROFIsafe addresses" parameter of all F-CPUs is unique network-wide*. This is achieved through different settings for the "Basis for PROFIsafe addresses" parameter of the F-CPUs. *(S052)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

** "CPU-wide" means all F-I/Os assigned to an F-CPU: central F-I/O of this F-CPU as well as F-I/Os for which the F-CPU is DP master/IO controller. An F-I/O that is addressed using I-slave-slave communication is assigned to the F-CPU of the I-slave and not to the F-CPU of the DP master / IO controller.

**WARNING**

Check the documentation for your fail-safe GSD based DP slaves / GSD based I/O devices to find out the valid PROFIsafe address type. If you do not find the necessary information, assume PROFIsafe address type 1. Proceed as described under PROFIsafe addresses for F-I/O of PROFIsafe address type 1 (Page 52) or Configurations supported by the SIMATIC Safety F-system (Page 50).

Set the PROFIsafe source address for fail-safe GSD based DP slaves / fail-safe GSD based I/O devices according to the manufacturer's specifications. If the PROFIsafe source address needs to correspond to the "Basis for PROFIsafe addresses" parameter of the F-CPU (PROFIsafe address type 2), you will find the latter in the "Properties" tab of the F-CPU. In this case, also check in the safety summary that the value of the F-CPU for the "Basis of PROFIsafe addresses" parameter matches the value of the PROFIsafe source address of the fail-safe GSD based DP slave/fail-safe GSD based I/O device. *(S053)*

2. Check the safety-related parameters (including F-monitoring time or F_WD_Time) of all configured F-I/O.

   You can find these parameters in the "Hardware configuration of F-I/O" section in the detailed tables for the F-I/O.

   The table consists of two parts:

   – Left part is for parameters that relate to the F-I/O themselves and their access function in the safety program ("Module data")
   – Right part is for the parameters of the individual channels ("Channel parameters")

   These parameters must be set as prescribed by the safety requirements of your application.

   When using fail-safe GSD based DP slaves/GSD based I/O devices, note the relevant documents for the possible additional safety-related (technical) parameters.

**Note**

F-I/O that are to be assigned the same safety-related parameters (except for PROFIsafe addresses) can be copied during configuration. Except for the PROFIsafe addresses, you no longer have to check the safety-related parameters individually. It is sufficient to compare the "Signature of F-parameters (without addresses)" in the "Hardware configuration of the F-I/O" section in the overview table. This also applies to fail-safe GSD based DP slaves/GSD based I/O devices without i-parameters. For GSD based DP slaves / GSD based I/O devices with i-parameters, it may be that "F-parameter signature (w/o addresses)" does not match, even though all safety-related parameters, except for the PROFIsafe addresses, do match. In this case, you need to compare all safety-related parameters.

3. Check whether the MLFBs of the F-I/O in the safety summary correspond to the MLFBs of the actual F-I/O in the system. If the MLFBs do not correspond, the F-I/O in the system must be spare-part compatible with the F-I/O listed in the safety summary.
4. For non-supported configuration, please see Configurations supported by the SIMATIC Safety F-system (Page 50).

**WARNING**

Please note the following with configurations that are not included in supported configurations:

- Make sure that the F-I/O appears in the safety summary and that an F-I/O DB has been created for it. Otherwise you cannot use the F-I/O in this configuration. (Contact Customer Support.)
- For F-I/Os in the PROFINET IO environment**, you must check the PROFIsafe operating mode parameter (F_Par_Version) against the safety summary to make sure that it is correct. V2 mode must be set in the PROFINET IO environment. F-I/O which only support V1 mode may not be used in the PROFINET IO environment.
- You must ensure that PROFIsafe address assignment is unique CPU-wide* and network-wide***:
  - Use the safety summary to check that the F-source address corresponds to the "Basis for PROFIsafe addresses" parameter of the F-CPU for F-I/O of PROFIsafe address type 2.
  - For F-I/O of PROFIsafe address type 1 or if you cannot set the F-source address in accordance with the F-CPU, you will have to ensure the uniqueness of the PROFIsafe address solely by assigning a unique F-destination address.

  You must check the uniqueness of the F-destination address individually for each F-I/O based on the safety summary in a configuration that is not supported. *(S050)*

* "CPU-wide" means all F-I/Os assigned to an F-CPU: central F-I/O of this F-CPU as well as F-I/Os for which the F-CPU is DP master/IO controller. An F-I/O that is addressed using I-slave-slave communication is assigned to the F-CPU of the I-slave and not to the F-CPU of the DP master / IO controller.

** The F-I/O is located in the "PROFINET IO environment" if at least part of safety-related communication with the F-CPU takes place via PROFINET IO. If the F-I/O is connected via I-slave-slave communication, also keep in mind the communication line to the DP master/IO controller.

*** A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

## 11.6 Correctness of the communication configuration

### Introduction

Safety-related communication is based on the mechanisms of the standard communication of *STEP 7*. To detect errors in standard communication, there must be additional protection for safety-related communication connections between F-CPUs. For the acceptance, you must document the restrictions (uniqueness) resulting from the protection.

For this purpose, the "Parameters for safety-related CPU-CPU-communication" section is available in the safety summary. In this section, there are up to two tables (for communication via PROFIBUS DP or PROFINET IO and for communication via S7 connections).

### Procedure for testing for correctness of the communication configuration

To check the communication configuration for correctness, proceed as follows:

- In the "Safety-related CPU-CPU communication via PROFIBUS DP or PROFINET IO" table, check to determine whether you have assigned a unique DP_DP_ID parameter throughout the network for all safety-related communication connections (e.g., master-master, master-I-slave and IO controller-IO controller communication).

  This means that sender and receiver of the relevant communication connection must have the same values for DP_DP_ID. All other communication connections in the entire network must not have these values.

- In the "Safety-related CPU-CPU communication via S7 connections" table, check to determine whether you have assigned a unique R_ID parameter throughout the network for all safety-related communication via S7 connections.

## 11.7 Consistency of the online safety program

Once you have checked all properties of the offline safety program you must ensure that the safety program is identical on the F-CPU on which it is supposed to be run.

The signatures must match the checked signature from the safety summary online and offline. To do so, open the *Safety Administration Editor* and go to online view.

In the "General" area, check the information for

- Program signature online
- Program signature offline
- Information on whether safety programs are consistent online and offline.
- Version check view

Use the output information to check which situation you are dealing with and, if necessary, execute the recommended measure:

| collective F-signatures online/offline | Consistency online to offline | Statement | Measure |
|---|---|---|---|
|  | (n/a) | The safety programs are different. | Acceptance of the online safety program required |
|  |  | The safety programs are identical but different versions of F-blocks are used. | The safety program must be downloaded to the F-CPU for the latest versions to become effective. |
|  |  | The safety programs are identical. | None |

Keep in mind that only a change comparison will provide reliable information as to whether the safety programs are identical. The display of signatures is used for quick identification of changes.

# 11.8 Other characteristics

## Introduction

In addition, you must check a few more characteristics that are also relevant for the acceptance of the project.

## Validity check for data transfer from the standard program to the safety program

Check to determine whether a validity check was programmed for all data transferred from the standard user program to the safety program. For this purpose, the "Data from the standard user program" section lists all signals that you are using in the safety program.

## Checking the program version

Check whether the version of *STEP 7 Safety* used to create the summary (in the footer of the printout) is as least as high as the version used to compile the safety program. The latter version can be found in the "General information" section of the safety summary under "Used Versions". Both versions must be listed in Annex 1 of the TÜV certificate.

## Ability to disable safety mode

Make sure that safety mode cannot be disabled. For information about this, refer to section "General information" under "Safety program settings". This setting ensures that the safety mode of the safety program cannot be disabled inadvertently.

## Access protection

Check in the "General information" section under "Protection" to determine whether the setting for access protection is permitted. Note the following warning.

Otherwise, the project must not be accepted, because the safety program in the F-CPU is not protected against unauthorized accesses.

**⚠ WARNING**

(S7-300, S7-400) In safety mode, access with the CPU password must not be authorized during changes to the standard user program as this would allow changes to the safety program. To rule out this possibility, you must configure **protection level "Write protection for fail-safe blocks"** and configure a password for the F-CPU. If only **one person** is authorized to change the standard user program **and** the safety program, the protection level "Write protection" or "Read/write protection" should be configured so that other persons have only limited access or no access at all to the entire user program (standard and safety programs). (See also Access protection (Page 76).) *(S001)*

**WARNING**

(S7-1200, S7-1500) In safety mode, the safety program must be password-protected. You therefore need to configure at least the protection level "Full access (no protection)". This protection level only allows full access to the standard user program, not to F-blocks.

If you select a higher protection level, for example to protect the standard user program, you must assign an additional password for "Full access (no protection)". *(S041)*

### See also

Data Transfer from Standard User Program to Safety Program (Page 161)

## 11.9 Acceptance of Changes

### Introduction

In general, you can adopt the same approach for the acceptance of changes as the initial acceptance (see Overview of System Acceptance (Page 308)).

To avoid the acceptance of the entire system in case of negligible changes, *STEP 7 Safety* helps you to identify those parts of your safety program that have changed.

For an acceptance of changes, it is sufficient to check the following:

- Checking the changed or newly added F-blocks.
- Checking the changed or newly added instructions and F-system blocks.
- Checking the safety-related parameters of the changed or newly added F-I/O.

You then perform a function test of the changed or newly added F-blocks/F-I/O.

**Note**

Acceptance of changes is not possible after CPU migration.

### Detection of changes in the safety program

To detect changes in the safety program, proceed as follows:

1. Perform an offline-offline comparison between the changed safety program and the accepted safety program (see Comparing Safety Programs (Page 286)). Use filter setting "Compare only F-blocks relevant for certification". This limits the output of the comparison to exactly those F-blocks that must be considered for the acceptance of changes.

   The status of the comparison enables you to identify which F-blocks were changed.

### Detection of safety-related changes in the configuration of the F-I/O

There are two possible ways of detecting safety-related changes in the configured F-I/O:

- Comparison in the comparison editor
- Comparison based on two safety summaries

### Comparison in the comparison editor

If you have saved the accepted project, you can also perform an offline-offline comparison of the changed project and the accepted project to detect changes in the configuration of the F-I/O (see Comparing Safety Programs (Page 286)).

1. Navigate in the comparison result to the "System blocks > STEP 7 Safety > F-I/O DBs" folder. All data blocks listed in this folder are F-I/O-DBs and are each assigned to an F-I/O.
2. The names of the F-I/O DBs (Page 138) are automatically assigned. They specify to which F-I/O they refer. If you have changed the names of the F-I/O DBs, you can find their numbers in the module data of the safety summary.

   If the F-I/O-DBs in the comparison result are identical, this means that the safety-related configuration (standard parameters may have changed) of the assigned F-I/O was not changed (assuming that your current safety program is consistent and compiled). In this way, you can quickly identify which F-I/O have changed.
3. If you have found changed F-I/O, you can check the changed parameters in the safety summary as described above. In so doing, always check the DB number to ensure that the DB name did not lead you to the wrong F-I/O.

### Comparison based on two safety summaries

In the overview table of the utilized F-I/O in the "Hardware configuration of F-I/O" section of the safety summary, compare the parameter CRCs of all GSD based F-I/O/DP slaves/GSD based I/O devices with those in the safety summary of the accepted project.

If the "Parameter signature (without addresses)" is different for an F-I/O, this indicates the existence of a safety-related change in the configuration of this F-I/O, e.g. including PROFIsafe addresses.

If this information is identical, only the PROFIsafe addresses were changed. In this case, you do not have to check the other safety-related parameters of the F-I/O individually.

Make sure that the uniqueness of the PROFIsafe addresses of all configured F-I/O will continue to be ensured. This means:

- The F-destination addresses with F-I/O of PROFIsafe address type 1.
- The F-destination addresses and the F-source address with F-I/O of PROFIsafe address type 2.

See also
PROFIsafe addresses for F-I/O of PROFIsafe address type 1 (Page 52)

PROFIsafe addresses for F-I/O of PROFIsafe address type 2 (Page 54)

Peculiarities when configuring fail-safe GSD based DP slaves and fail-safe GSD based I/O devices (Page 61)

### Detecting changes of the start addresses of F-I/O

You can detect changes in the start addresses of F-I/O in the output of the program comparison, as well. Note that the names of the F-I/O DBs contain the address in this case, provided you have not changed the names manually. In order to detect changes, it is necessary to distinguish between the following two cases:

- You have not changed the name of the F-I/O DB; the name represents the same F-I/O in both program versions.

  If you change the start address of an F-I/O, this also changes the name of the F-I/O DB. If you then compare the program with its previous version, the newly named F-I/O does not exist in the old program version and the formerly named F-I/O does not exist in the new program version. The comparison log contains two lines in which the names are different. Both I-DBs represent the module whose start address has changed. Carry out a comparison for this module using the module signatures and check the fail-safe parameters, if applicable.

- You have changed the name of the F-I/O DB.

  In this case, the F-I/O DB appears in the comparison output as a changed F-I/O, just the same as if other module parameters have changed.

# Operation and Maintenance 12

## 12.1 Notes on Safety Mode of the Safety Program

### Introduction

Pay attention to the following important notes on safety mode of the safety program.

### Use of simulation devices/simulation programs

**⚠ WARNING**

**Use of simulation devices/simulation programs**

If you operate simulation devices or simulation programs that generate safety message frames, e.g., based on PROFIsafe, and make them available to the SIMATIC Safety F-system via the bus system (such as PROFIBUS DP or PROFINET IO), you must ensure the safety of the F-system using organizational measures, such as operational monitoring and manual safety shutdown.

If you use the S7-PLCSIM (Page 292) to simulate safety programs, these measures are not necessary because S7-PLCSIM cannot establish an online connection to a real component.

Note, for example, that a protocol analyzer is not permitted to perform any function that reproduces recorded message frame sequences with correct time behavior. *(S030)*

### STOP via programming device or PC, mode selector, or communication function

**⚠ WARNING**

**STOP via programming device/PC, mode selector, or communication function**

Switching from STOP to RUN mode using a programming device/PC interface, mode selector, or communication function is not locked. For example, only one keystroke is necessary to switch from STOP to RUN mode on a programming device or PC interface. For this reason, a STOP that you have set by means of a programming device or PC, mode selector, or communication function must not be regarded as a safety condition.

You must therefore program startup protection (see Programming startup protection (Page 121)). *(S031)*

## Switch F-CPU to STOP mode with the "STP" Instruction

**WARNING**

**Switching F-CPU to STOP mode with the "STP" Instruction**

A STOP state initiated by the "STP" instruction can be canceled very easily (and unintentionally) from the programming device or PC. For this reason, a STOP initiated by the STP instruction is not a safety-related STOP.

You must therefore program startup protection (see Programming startup protection (Page 121)). *(S032)*

## CRC error in safety-related communication

**Note**

**CRC error in safety-related communication**

If you observe that an F-CPU requests manual acknowledgement of a CRC error more than once within the space of 100 hours, and this occurs repeatedly, check whether the PROFINET or PROFIBUS installation guidelines have been followed.

There is a CRC error if:

- The ACK_REQ tag of the F-I/O DB is set and the DIAG tag of the F-I/O DB (bit 2 or bit 6) indicates CRC errors
  or
- A CRC error is entered in the diagnostic buffer of the F-CPU

In this case, the failure probability values (PFD/PFH) for safety-related communication no longer apply.

Information on installation guidelines for PROFINET and PROFIBUS can be found in:

- PROFIBUS Installation Guidelines (www.profibus.com/PBInstallationGuide)
- PROFIBUS Interconnection Technology (http://www.profibus.com/nc/downloads/downloads/profibus-interconnection-technology/display/)
- PROFINET Installation Guide (www.profibus.com/PNInstallationGuide)
- PROFINET Cabling and Interconnection Technology (http://www.profibus.com/nc/downloads/downloads/profinet-cabling-and-interconnection-technology/display/)
- PROFIsafe Environment Requirements (www.profibus.com/PROFIsafeRequirements)

If your review indicates that the configuration guidelines for PROFIBUS and PROFINET have been met, contact Technical Support.

# 12.2 Replacing Software and Hardware Components

## Replacement of software components

When replacing software components on your programming device or PC (e.g. with a new version of *STEP 7)*, you must adhere to the information regarding upward and downward compatibility in the documentation and readme files for these products.

## Replacement of hardware components

Hardware components for SIMATIC Safety (F-CPU, F-I/O, batteries, etc.) are replaced in the same way as in standard automation systems.

## Removing and inserting F-I/O during operation

It is possible to remove and insert F-I/O during operation, as with standard F-I/O. However, be aware that replacing an F-I/O module during operation can cause a communication error in the F-CPU.

You must acknowledge the communication error in your safety program in the ACK_REI tag of the F-I/O DB (Page 132) or, alternatively, by using the "ACK_GL (Page 407)" instruction. Without an acknowledgment, the F-I/O will remain passivated.

## CPU firmware update

Check of the CPU operating system for F-approval: When using a new CPU operating system (firmware update), you must check to see if the CPU operating system you are using is approved for use in an F-system.

The minimum CPU operating system versions with guaranteed F-capability are specified in the appendix of the Certificate. This information and any notes on the new CPU operating system must be taken into account.

## Firmware update for interface module

When using a new operating system for an interface module, e.g. IM 151-1 HIGH FEATURE ET 200S (firmware update), you must observe the following:

If you have selected the "Activate firmware after update" option for the firmware update (see *Help on STEP 7*, "Online & Diagnostics"), the IM will be automatically reset following a successful download operation and will then run on the new operating system. Note that the firmware update for interface modules during operation generates a communication error in the F-CPU.

You must acknowledge the communication error in your safety program in the ACK_REI tag of the F-I/O DB (Page 132) or, alternatively, by using the "ACK_GL (Page 407)" instruction. Without an acknowledgment, the F-I/O will remain passivated.

## Preventive maintenance (proof test)

The probability values for the certified F-system components guarantee a proof-test interval of 20 years for ordinary configurations.

Proof test for complex electronic components generally means replacement with new, unused components.

## PFD and PFH values for S7-300/400 F-CPUs and F-I/O

You will find a list of the failure probability values (PFD and PFH values) for components that can be used in SIMATIC Safety on the Internet (http://support.automation.siemens.com/WW/view/en/49368678/133300).

## PFD and PFH values for S7-1200/1500 F-CPUs

Below are the probability of failure values ($PFD_{avg}$, PFH values) for S7-1200 F-CPUs with a service life of 20 years and an MTTR of 100 hours:

| **Operation in low demand mode**<br>low demand mode<br>**According to IEC 61508:2010:**<br>**$PFD_{avg}$ = Average probability of a dangerous failure on demand** | **Operation in high demand or continuous mode**<br>high demand/continuous mode<br>**According to IEC 61508:2010:**<br>**PFH = Average frequency of a dangerous failure [$h^{-1}$]** |
|---|---|
| < < 2E-05 | < 1E-09 |

## PFD, PFH values for safety-related communication

Below you will find the failure probability values ($PFD_{avg}$, PFH values) for safety-related communication:

| **Operation in low demand mode**<br>low demand mode<br>**According to IEC 61508:2010:**<br>**$PFD_{avg}$ = Average probability of a dangerous failure on demand** | **Operation in high demand or continuous mode**<br>high demand/continuous mode<br>**According to IEC 61508:2010:**<br>**PFH = Average frequency of a dangerous failure [$h^{-1}$]** |
|---|---|
| < 1E-05* | < 1E-09* |

*** Note on S7-300/400 F-CPUs:**

The PFH value is valid under the assumption that a maximum of 100 F-I/Os are involved in a safety function. If you use more than 100 F-I/Os, you have to also add 4E-12 per F-I/O for the safety function.
The $PFD_{avg}$ value is valid for a service life of 20 years and under the assumption, that a maximum of 25 F-I/Os are involved in a safety function. If more than 25 F-I/OS are used, you need to add 3.5E-7 per F-I/O for this safety function.

# 12.3 Guide to diagnostics (S7-300, S7-400)

## Introduction

Here you find a compilation of diagnostic capabilities that can be evaluated for your system when an error occurs. Most of the diagnostic capabilities are the same as those in standard automation systems. The sequence of steps represents a recommendation.

## Steps for evaluating diagnostic capabilities

The following table shows the steps you take to evaluate diagnostic capabilities.

| Step | Procedure | Reference |
|---|---|---|
| 1 | **Evaluate LEDs on the hardware (F-CPU, F-I/O):**<br>• BUSF LED on the F-CPU: Flashes when a communication error occurs on PROFIBUS DP/PROFINET IO;<br>On if a programming error occurs when OB85 and OB121 are programmed (e.g. instance DB is not loaded)<br>• STOP LED on the F-CPU: illuminates when the F-CPU is in STOP mode<br>• Fault LEDs on the F-I/O: e.g. SF-LED (group error LED) on if any fault occurs in the individual F-I/O | *Manuals for F-CPU and F-I/O* |
| 2 | **Evaluate diagnostic buffer of the modules:**<br>You read the diagnostic buffer of a module (F-CPU, F-I/O, CP) in its online and diagnostic view in the "Diagnostic buffer" group under the "Online & Diagnostics folder. | *Help on STEP 7 and manuals for the F-CPU and F-I/O* |
| 3 | **Evaluate stacks of the F-CPU:**<br>when the F-CPU is in STOP mode, read the following successively:<br>• Block stack: Check whether STOP mode of the F-CPU was triggered by an F-block of the safety program<br>• Interruption stack<br>• Local data stack | *Help on STEP 7* |
| 4 | **Evaluate diagnostic tag of the F-I/O DB using testing and commissioning functions, by means of an operator control and monitoring system, or in the standard user program:**<br>Evaluate the DIAG tag in the F-I/O DB | F-I/O access (Page 122) |

| Step | Procedure | Reference |
|---|---|---|
| 5 | **Evaluate diagnostic outputs of the instance DBs of instructions using testing and commissioning functions, using an operator control and monitoring system, or in the standard user program:**<br>• For MUTING, EV1oo2DI, TWO_H_EN, MUT_P, ESTOP1, FDBBACK, SFDOOR, evaluate in the assigned instance:<br>– Output DIAG<br>• Evaluate the following for SENDDP or RCVDP in the assigned instance DB:<br>– Output RET_DPRD/RET_DPWR<br>– Output DIAG<br>• Evaluate the following for SENDS7 or RCVS7 in the assigned instance DB:<br>– Output STAT_RCV<br>– Output STAT_SND<br>– Output DIAG | Instructions (Page 494) |

### Tip on RET_DPRD/RET_DPWR

The diagnostic information of the RET_DPRD/RET_DPWR parameters correspond to the diagnostic information of the RETVAL parameter of the "DPRD_DAT" and "DPWR_DAT" instructions. You can find the description in the help on *STEP 7* for the "DPRD_DAT" and "DPWR_DAT" instructions.

### Tip: STAT_RCV and STAT_SND

The diagnostic information of the parameter STAT_RCV corresponds to the diagnostic information of the parameter STATUS of the instruction "URCV". The diagnostic information of the parameter STAT_SND corresponds to the diagnostic information of the parameter STATUS of the instruction "USEND". You can find the description in the help on *STEP 7* for the instruction "UCRV" or "USEND" .

## 12.4 Guide to diagnostics (S7-1500)

Detailed information on diagnostics for an S7-1500 F-CPU can be found in the Diagnostics (http://support.automation.siemens.com/WW/view/en/59192926) function manual.

## 12.5 Guide to diagnostics (S7-1200)

Detailed information on diagnostics for an S7-1200 F-CPU can be found in the S7-1200 User Manual Fail-Safety (http://support.automation.siemens.com/WW/view/en/34612486/133300).

# STEP 7 Safety V13 SP 1 instructions 13

## 13.1 Overview of instructions

### Overview of instructions for the safety program

When programming an F-block, you can find all instructions available for programming an F-block in LAD or FBD with the configured F-CPU in the "Instructions" task card.

In addition to the instructions that are familiar to you from programming a standard block, there are also special safety functions, e.g., for two-hand monitoring, discrepancy analysis, muting, emergency STOP, safety door monitoring, and feedback monitoring.

All instructions for LAD and FBD are explained in the following.

### Note the following

**Note**

Preconnection of enable input EN or evaluation of enable output ENO is not possible.

# 13.2 Instructions - LAD

## 13.2.1 General

### 13.2.1.1 New network (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Requirement

An F-block is open.

#### Procedure

To insert a new network, follow these steps:

1. Select the network after which you want to insert a new network.
2. Select the "Insert network" command in the shortcut menu.

**Note**

If you insert an element into the last empty network of the F-block in an LAD program, a new empty network is automatically inserted below it.

#### Result

A new empty network is inserted into the F-block.

### 13.2.1.2 Empty box (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Requirement

A network is available.

#### Procedure

To insert an LAD instruction into a network using an empty box, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to "Basic instructions > General > Empty box".
3. Use a drag-and-drop operation to move the "Empty box" element to the desired place in the network.
4. Hover the cursor over the yellow triangle in the top right corner of the empty box.

   A drop-down list is displayed.
5. Select the required instruction from the drop-down list.

If the instruction acts as a function block (FB) within the system, the "Call options" dialog opens. In this dialog, you can create an instance data block for the function block, either as a single instance or, if necessary, multi-instance, in which data of the inserted instruction are stored. Once it is created, the new instance data block can be found in the "Program resources" folder in the project tree under "Program blocks > System blocks". If you have selected "multi-instance", you can find it in the block interface in the "Static" section.

#### Result

The empty box is changed to the appropriate instruction. Placeholders are inserted for the parameters.

### 13.2.1.3 Open branch (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

Use branches to program parallel connections with the Ladder Logic (LAD) programming language. Branches are inserted into the main current path. You can insert several contacts into the branch, thereby creating a parallel connection from series connections. You can program complex ladder diagrams in this way.

#### Requirement

- A network is available.
- The network contains elements.

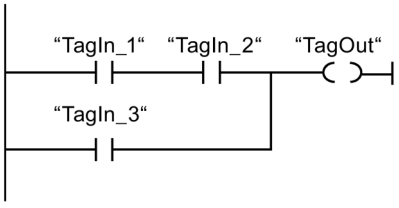
### Procedure

To insert a new branch in a network, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to "Basic instructions > General > Open branch".
3. Use a drag-and-drop operation to move the element to the desired place in the network.
4. If you want to connect the new branch directly to the power rail, drag the element to the power rail.

### Example

The following figure provides an example of how to use branches:




## 13.2.1.4 Close branch (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

### Description

Branches must be closed again at suitable places. When a branch is closed, any necessary empty elements are added. If necessary, branches will be arranged so that they do not cross each other.

### Requirement

A branch is available.

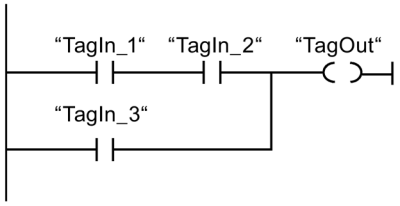
### Procedure

To close an open branch, follow these steps:

1. Select the open branch.
2. Press and hold down the left mouse button.
3. A dashed line will appear as soon as the cursor is moved.
4. Drag the dashed line to a suitable place on the network. Permissible connections are indicated by green lines.
5. Release the left mouse button.

### Example

The figure below provides an example of how to use branches:




## 13.2.2 Bit logic operations

### 13.2.2.1 ---| |---: Normally open contact (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

### Description

The activation of the normally open contact depends on the signal state of the associated operand. If the operand has signal state "1," the normally open contact is closed. Power flows from the left power rail through the normally open contact into the right power rail and the signal state at the output of the instruction is set to "1".

If the operand has signal state "0," the normally open contact is not activated. The power flow to the right power rail is interrupted and the signal state at the output of the instruction is reset to "0".

Two or more normally open contacts are linked bit-by-bit by AND when connected in series. With a series connection, power flows when all contacts are closed.

The normally open contacts are linked by OR when connected in parallel. With a parallel connection, power flows when one of the contacts is closed.

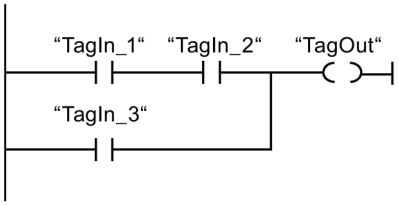
### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Input | BOOL | Operand whose signal state is queried. |

## Example

The following example shows how the instruction works:




Operand "TagOut" is set when one of the following conditions is fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state at operand "TagIn_3" is "1".

### 13.2.2.2 ---| / |---: Normally closed contact (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

The activation of the normally closed contact depends on the signal state of the associated operand. If the operand has signal state "1", the normally closed contact is opened and the signal state at the output of the instruction is reset to "0".

If the operand has signal state "0", the normally closed contact is not activated and the signal state at the output of the instruction is set to "1".

Two or more normally closed contacts are linked bit-by-bit by AND when connected in series. With a series connection, power flows when all contacts are closed.

The normally closed contacts are linked by OR when connected in parallel. With a parallel connection, power flows when one of the contacts is closed.

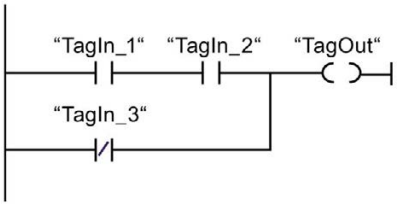
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Input | BOOL | Operand whose signal state is queried. |

### Example

The following example shows how the instruction works:




Operand "TagOut" is set when one of the following conditions is fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state at operand "TagIn_3" is "0".

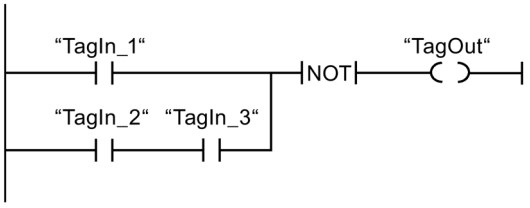
## 13.2.2.3 --|NOT|--: Invert RLO (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

### Description

You can use the "Invert RLO" instruction to invert the signal state of the result of logic operation (RLO). When the signal state is "1" at the input of the instruction, the output of the instruction has the signal state "0". When the signal state is "0" at the input of the instruction, the output has the signal state "1".

### Example

The following example shows how the instruction works:




Operand "TagOut" is reset when one of the following conditions is fulfilled:

- Operand "TagIn_1" has signal state "1".
- Operands "TagIn_2" and "TagIn_3" have signal state "1".

### 13.2.2.4 ---( )---: Assignment (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Assignment" instruction to set the bit of a specified operand. When the result of logic operation (RLO) at the input of the coil is "1," the specified operand is set to signal state "1". When the signal state is "0" at the input of the coil, the bit of the specified operand is reset to "0".

The instruction does not influence the RLO. The RLO at the input of the coil is sent immediately to the output.

The "Assignment" instruction can be placed at any position in the network.

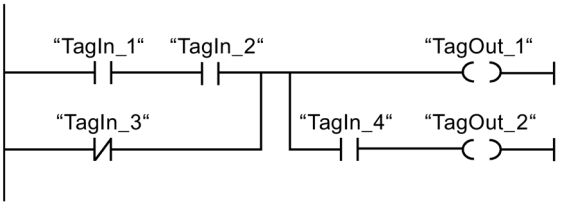
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Output | BOOL | Operand to which the RLO is assigned. |

## Example

The following example shows how the instruction works:




Operand "TagOut_1" is set when one of the following conditions is fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state at operand "TagIn_3" is "0".

Operand "TagOut_2" is set when one of the following conditions is fulfilled:

- Operands "TagIn_1" and "TagIn_2" as well as operand "TagIn_4" have signal state "1".
- The signal state of operand "TagIn_3" is "0" and the signal state of operand "TagIn_4" is "1".

### 13.2.2.5 ---( R )---: Reset output (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Reset output" instruction to reset the signal state of a specified operand to "0".

If power flows to the coil (RLO is "1"), the specified operand is set to "0". If the result of logic operation at the input of the coil is "0" (no signal flow to the coil), the signal state of the specified operand remains unchanged.

The instruction does not influence the RLO. The RLO at the input of the coil is sent directly to the output of the coil.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the operands of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

If the operand area "local data (temp)" is used for the operands of the instruction, the local data bit used must be initialized beforehand.

**Note**

You cannot use the "process image of the inputs", "process image of the outputs" from standard I/O and "standard DB" and "bit memory" operand areas for the operands of the instruction.

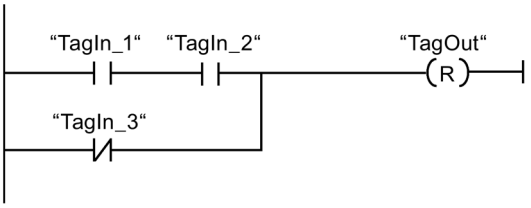
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Output | BOOL | Operand that is reset when RLO = "1". |

## Example

The following example shows how the instruction works:




Operand "TagOut" is reset when one of the following conditions is fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state of operand "TagIn_3" is "0".

### 13.2.2.6 ---( S )---: Set output (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Set output" instruction to set the signal state of a specified operand to "1".

If power flows to the coil (RLO is "1"), the specified operand is set to "1". If the result of logic operation at the input of the coil is "0" (no signal flow to the coil), the signal state of the specified operand remains unchanged.

The instruction does not influence the RLO. The RLO at the input of the coil is sent directly to the output of the coil.

**Note**

The instruction is not executed if it is applied to an output of an F-I/O that is passivated (e.g., during startup of the F-system). Therefore, it is preferable to access outputs of the F-I/O using only the "Assignment" instruction.

An F-I/O output is passivated if QBAD or QBAD_O_xx = 1 or value status = 0 is set in the corresponding F-I/O DB.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the operands of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

If the operand area "local data (temp)" is used for the operands of the instruction, the local data bit used must be initialized beforehand.

**Note**

You cannot use the "process image of the inputs", "process image of the outputs" from standard I/O and "standard DB" and "bit memory" operand areas for the operands of the instruction.

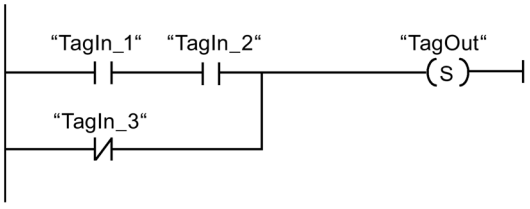
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Output | BOOL | Operand that is set when RLO = "1". |

## Example

The following example shows how the instruction works:




Operand "TagOut" is set when one of the following conditions is fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state of operand "TagIn_3" is "0".

### 13.2.2.7 SR: Set/reset flip-flop (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Set/reset flip-flop" instruction to set or reset the bit of the specified operand based on the signal state of inputs S and R1. If the signal state at input S is "1" and the signal state at input R1 is "0", the specified operand is set to "1". If the signal state at input S is "0" and the signal state at input R1 is "1", the specified operand is reset to "0".

Input R1 takes priority over input S. If the signal state is "1" at the two inputs S and R1, the signal state of the specified operand is reset to "0".

The instruction is not executed if the signal state at the two inputs S and R1 is "0". The signal state of the operand then remains unchanged.

The current signal state of the operand is transferred to output Q and can be queried there.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the operands of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the operands of the instruction.

If the operand area "local data (temp)" is used for the operands of the instruction, the local data bit used must be initialized beforehand.

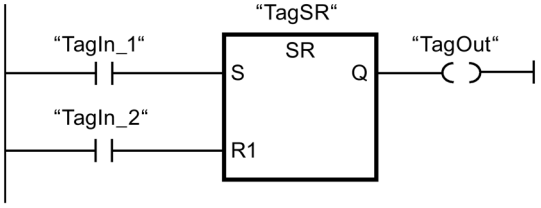
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| S | Input | BOOL | Enable setting |
| R1 | Input | BOOL | Enable resetting |
| <Operand> | Output | BOOL | Operand that is set or reset. |
| Q | Output | BOOL | Signal state of the operand |

## Example

The following example shows how the instruction works:




Operands "TagSR" and "TagOut" are set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- Operand "TagIn_2" has signal state "0".

Operands "TagSR" and "TagOut" are reset when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "0" and operand "TagIn_2" has signal state "1".
- Both operands "TagIn_1" and "TagIn_2" have signal state "1".

### 13.2.2.8 RS: Reset/set flip-flop (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Reset/set flip-flop" instruction to reset or set the bit of the specified operand based on the signal state of inputs R and S1. When the signal state is "1" at input R and "0" at input S1, the specified operand is reset to "0". When the signal state is "0" at input R and "1" at input S1, the specified operand is set to "1".

Input S1 takes priority over input R. If the signal state is "1" at the two inputs R and S1, the signal state of the specified operand is set to "1".

The instruction is not executed if the signal state at the two inputs R and S1 is "0". The signal state of the operand then remains unchanged.

The current signal state of the operand is transferred to output Q and can be queried there.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the operands of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the operands of the instruction.

If the operand area "local data (temp)" is used for the operands of the instruction, the local data bit used must be initialized beforehand.

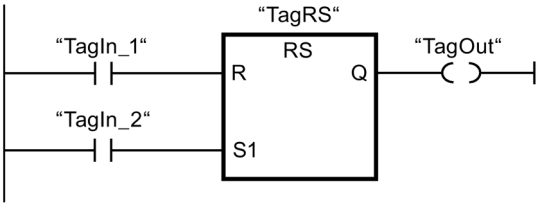
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| R | Input | BOOL | Enable resetting |
| S1 | Input | BOOL | Enable setting |
| <Operand> | Output | BOOL | Operand that is reset or set. |
| Q | Output | BOOL | Signal state of the operand |

## Example

The following example shows how the instruction works:




Operands "TagRS" and "TagOut" are reset when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- Operand "TagIn_2" has signal state "0".

Operands "TagRS" and "TagOut" are set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "0" and operand "TagIn_2" has signal state "1".
- Operands "TagIn_1" and "TagIn_2" have signal state "1".

### 13.2.2.9 --|P|--: Scan operand for positive signal edge (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Scan operand for positive signal edge" instruction to determine if there is a change from "0" to "1" in the signal state of a specified operand (<Operand1>). The instruction compares the current signal state of <Operand1> with the signal state of the previous query saved in <Operand2>. If the instruction detects a change in the result of logic operation from "0" to "1", there is a positive, rising edge.

If a rising edge is detected, the output of the instruction has signal state "1". In all other cases, the signal state at the output of the instruction is "0".

Enter the operand to be queried (<Operand1>) in the operand placeholder above the instruction. Enter the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

**Note**

The address of the edge memory bit must not be used more than once in the program, otherwise the edge memory bit would be overwritten. This would influence edge evaluation and the result would no longer be unequivocal.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the edge memory bit <Operand2> of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the edge memory bit <Operand2> of the instruction.

If operand area "local data (temp)" is used for the edge memory bit <Operand2> of the instruction, the local data bit used must be initialized beforehand.

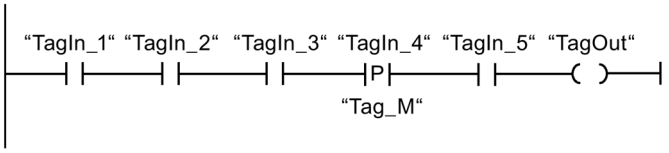
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | BOOL | Signal to be queried |
| <Operand2> | InOut | BOOL | Edge memory bit in which the signal state of the previous query is saved. |

## Example

The following example shows how the instruction works:




Operand "TagOut" is set when the following conditions are fulfilled:

- Operands "TagIn_1", "TagIn_2", and "TagIn_3" have signal state "1".
- There is a rising edge at input "TagIn_4". The signal state of the previous query is saved at edge memory bit "Tag_M".
- The signal state of operand "TagIn_5" is "1".

### 13.2.2.10 --|N|--: Scan operand for negative signal edge (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Scan operand for negative signal edge" instruction to determine if there is a change form "1" to "0" in the signal state of a specified operand. The instruction compares the current signal state of <Operand1> with the signal state of the previous query saved in <Operand2>. If the instruction detects a change in the result of logic operation from "1" to "0", there is a negative, falling edge.

If a falling edge is detected, the output of the instruction has signal state "1". In all other cases, the signal state at the output of the instruction is "0".

Enter the operand to be queried (<Operand1>) in the operand placeholder above the instruction. Enter the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

**Note**

The address of the edge memory bit must not be used more than once in the program, otherwise the edge memory bit would be overwritten. This would influence edge evaluation and the result would no longer be unequivocal.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the edge memory bit <Operand2> of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the edge memory bit <Operand2> of the instruction.

If operand area "local data (temp)" is used for the edge memory bit <Operand2> of the instruction, the local data bit used must be initialized beforehand.

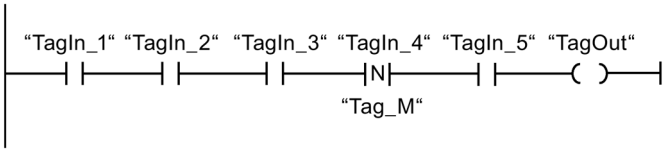
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | BOOL | Signal to be queried |
| <Operand2> | InOut | BOOL | Edge memory bit in which the signal state of the previous query is saved. |

## Example

The following example shows how the instruction works:




Operand "TagOut" is set when the following conditions are fulfilled:

- Operands "TagIn_1", "TagIn_2", and "TagIn_3" have signal state "1".
- There is a falling edge at operand "TagIn_4". The signal state of the previous query is saved at edge memory bit "Tag_M".
- The signal state of operand "TagIn_5" is "1".

### 13.2.2.11 P_TRIG: Scan RLO for positive signal edge (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Scan RLO for positive signal edge" instruction to query a change in the signal state of the result of logic operation from "0" to "1". The instruction compares the current signal state of the result of logic operation (RLO) with the signal state of the previous query, which is saved in the edge bit memory (<Operand>). If the instruction detects a change in the RLO from "0" to "1", there is a positive, rising edge.

If a rising edge is detected, the output of the instruction has signal state "1". In all other cases, the signal state at the output of the instruction is "0".

**Note**

The address of the edge memory bit must not be used more than once in the program, otherwise the edge memory bit would be overwritten. This would influence edge evaluation and the result would no longer be unequivocal.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the edge memory bit <Operand> of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the edge memory bit <Operand> of the instruction.

If operand area "local data (temp)" is used for the edge memory bit <Operand> of the instruction, the local data bit used must be initialized beforehand.

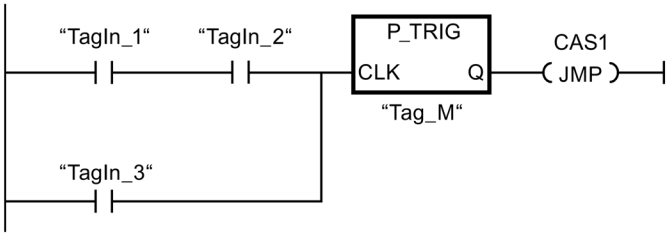
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CLK | Input | BOOL | Current RLO |
| <Operand> | InOut | BOOL | Edge memory bit in which the RLO of the previous query is saved. |
| Q | Output | BOOL | Result of edge evaluation |

### Example

The following example shows how the instruction works:




The RLO from the previous bit logic operation is saved in edge memory bit "Tag_M". If a change in the RLO signal state from "0" to "1" is detected, the program jumps to jump label CAS1.

## 13.2.2.12 N_TRIG: Scan RLO for negative signal edge (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

### Description

You can use the "Scan RLO for negative signal edge" instruction to query a change in the signal state of the result of logic operation from "1" to "0". The instruction compares the current signal state of the result of logic operation with the signal state from the previous query, which is saved in the edge memory bit (<Operand>). If the instruction detects a change in the RLO from "1" to "0", there is a negative, falling edge.

If a falling edge is detected, the output of the instruction has signal state "1". In all other cases, the signal state at the output of the instruction is "0".

**Note**

The address of the edge memory bit must not be used more than once in the program, otherwise the edge memory bit would be overwritten. This would influence edge evaluation and the result would no longer be unequivocal.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the edge memory bit <Operand> of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the edge memory bit <Operand> of the instruction.

If operand area "local data (temp)" is used for the edge memory bit <Operand> of the instruction, the local data bit used must be initialized beforehand.

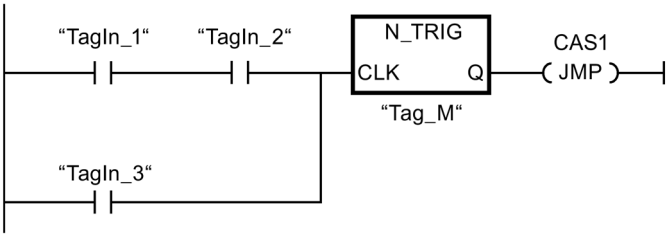
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CLK | Input | BOOL | Current RLO |
| <Operand> | InOut | BOOL | Edge memory bit in which the RLO of the previous query is saved. |
| Q | Output | BOOL | Result of edge evaluation |

## Example

The following example shows how the instruction works:




The RLO of the previous bit logic operation is saved in edge bit memory "Tag_M". If a change in the RLO signal state from "1" to "0" is detected, the program jumps to jump label CAS1.

## 13.2.3 Safety functions

### 13.2.3.1 ESTOP1: Emergency STOP up to stop category 1 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

This instruction implements an emergency STOP shutdown with acknowledgment for Stop Categories 0 and 1.

Enable signal Q is reset to 0, as soon as input E_STOP takes a signal state of 0 (Stop category 0). Enable signal Q_DELAY is reset to 0 after the time delay set at input TIME_DEL (Stop Category 1).

Enable signal Q is reset to 1 not before input E_STOP takes a signal state of 1 and an acknowledgment occurs. The acknowledgment for the enable takes place according to the parameter assignment at input ACK_NEC:

- If ACK_NEC = 0, the acknowledgment is automatic.
- If ACK_NEC = 1, you must use a rising edge at input ACK for acknowledging the enable.

Output ACK_REQ is used to signal that a user acknowledgment is required at input ACK for the acknowledgment. The instruction sets output ACK_REQ to 1, as soon as input E_STOP = 1.

Following an acknowledgment, the instruction resets ACK_REQ to 0.

Every call of the "Emergency STOP up to Stop Category 1" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., ESTOP1_DB_1) or a multi-instance (e.g., ESTOP1_Instance_1) for the "Emergency STOP up to Stop Category 1" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

The ACK_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S033)*

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

**Note:** Only one emergency STOP signal (E_STOP) can be evaluated for the instruction. Discrepancy monitoring of the two NC contacts (when two channels are involved) in accordance with Categories 3 and 4 as defined in ISO 13849-1:2006 / EN ISO 13849-1:2008 is performed with suitable configuration (type of sensor interconnection: 2-channel equivalent) directly through the F-I/O with inputs. In order to keep the discrepancy time from influencing the response time, during the configuration of discrepancy behavior, you must configure "Supply value 0".

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| E_STOP | Input | BOOL | Emergency STOP |
| ACK_NEC | Input | BOOL | 1=Acknowledgment necessary |
| ACK | Input | BOOL | 1=Acknowledgment |
| TIME_DEL | Input | TIME | Time delay |
| Q | Output | BOOL | 1=Enable |
| Q_DELAY | Output | BOOL | Enable is OFF delayed |
| ACK_REQ | Output | BOOL | 1=Acknowledgment necessary |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | Version 1.0 requires that the F_TOF block with the number FB 186 is available in the project tree in the "Program blocks/System blocks/STEP 7 Safety" folder.<br>When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically. If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. You will then avoid number conflicts. |
| 1.1 | x | — | -— | These versions are functionally identical to version V1.0, but do not require the F_TOF block to have a particular number. |
| 1.2 | x | — | x | |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Startup characteristics

After an F-system startup, when ACK_NEC = 1, you must acknowledge the instruction using a rising edge at input ACK.

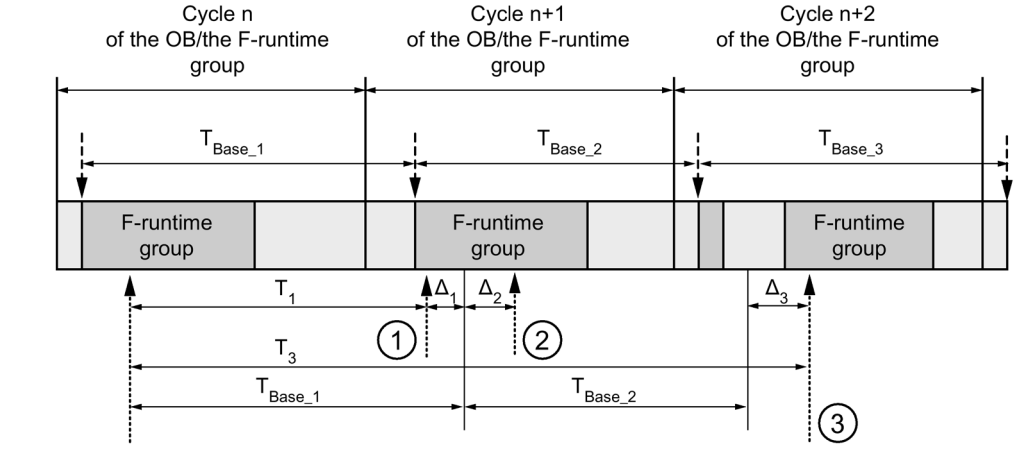
## Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits 1 to 5 are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Incorrect TIM_DEL setting | Time delay setting < 0 | Set time delay > 0 |
| Bit 1 | Reserved | — | — |
| Bit 2 | Reserved | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Acknowledgment not possible because emergency STOP is still active | Emergency STOP switch is locked | Release interlocking of emergency STOP switch |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of emergency STOP switch | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Emergency STOP switch is defective | Check emergency STOP switch |
| | | Wiring fault | Check wiring of emergency STOP switch |
| Bit 5 | If enable is missing: input ACK has a permanent signal state of 1 | Acknowledgment button defective | Check acknowledgment button |
| | | Wiring fault | Check wiring of acknowl-edgment button |
| Bit 6 | Acknowledgment required (= state of ACK_REQ) | — | — |
| Bit 7 | State of output Q | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




----► = Time base update

·····► = Call time of an instruction with time processing

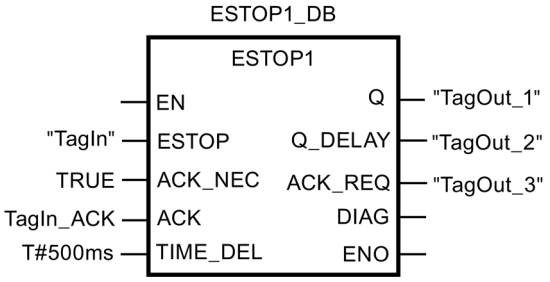
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

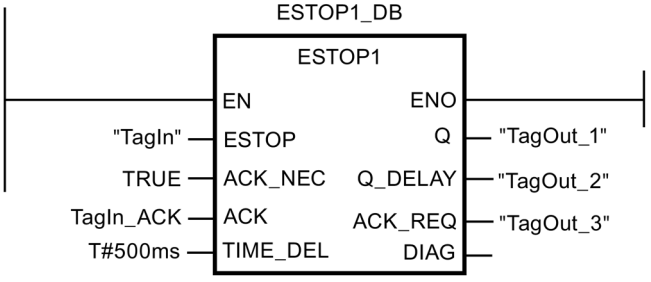
② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:




### 13.2.3.2 TWO_HAND: Two-hand monitoring (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

This instruction implements two-hand monitoring.

**Note**

This instruction is only available for S7-300 and S7-400 F-CPUs. For S7-1200/1500 F-CPUs, you use the instruction "Two-hand monitoring with enable". The application "Two-hand monitoring with enable" replaces the instruction "Two-hand monitoring" with compatible functions.

If pushbuttons IN1 and IN2 are activated within the permitted discrepancy time DISCTIME ≤ 500 ms (IN1/IN2 = 1) (synchronous activation), output signal Q is set to 1. If the time difference between activation of pushbutton IN1 and pushbutton IN2 is greater than DISCTIME, then the pushbuttons must be released and reactivated.

Q is reset to 0 as soon as one of the pushbuttons is released (IN1/IN2 = 0). Enable signal Q can be reset to 1 only if the other pushbutton has been released, and if both pushbuttons are then reactivated within the discrepancy time. Enable signal Q can never be set to 1 if the discrepancy time is set to values less than 0 or greater than 500 ms.

Every call of the "Two-hand monitoring" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., TWO_HAND_DB_1) or a multi-instance (e.g., TWO_HAND_Instance_1) for the "Two-hand monitoring" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

The instruction supports the requirements in accordance with EN 574:1996 + A1:2008.

**Note:** Only one signal per pushbutton can be evaluated in the instruction. Discrepancy monitoring of the NC and NO contacts of pushbuttons IN1 and IN2 is performed directly during suitable configuration (type of sensor interconnection: 2-channel non equivalent) directly through the F-I/O with inputs. The normally open contact must be wired in such a way that it supplies the useful signal (see manual for the F-I/O you are using). In order to keep the discrepancy time from influencing the response time, you must assign "Supply value 0" for the behavior of discrepancy during configuration. If a discrepancy is detected, a fail-safe value of 0 is entered in the process image of the inputs (PII) for the pushbutton and QBAD or QBAD_I_xx = 1 is set in the relevant F-I/O DB. (See also F-I/O access (Page 122))

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

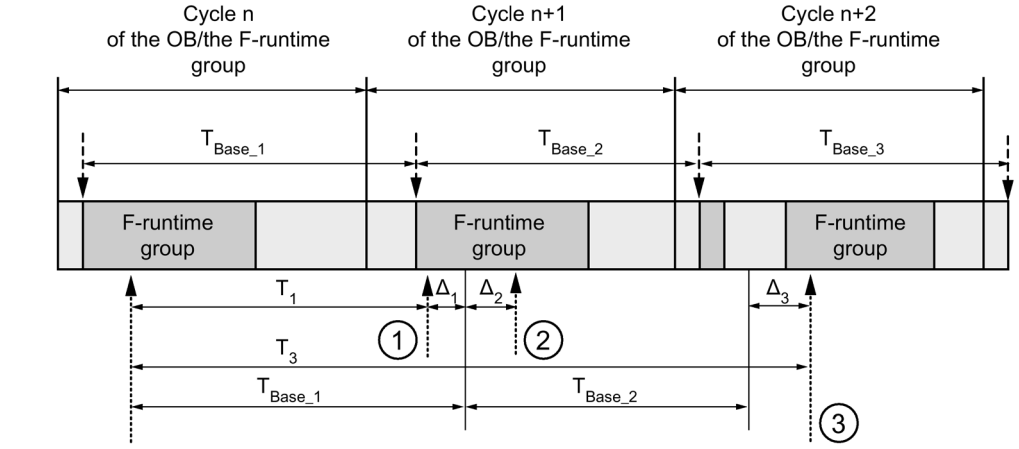
You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | BOOL | Pushbutton 1 |
| IN2 | Input | BOOL | Pushbutton 2 |
| DISCTIME | Input | TIME | Discrepancy time (0 to 500 ms) |
| Q | Output | BOOL | 1=Enable |

## Timing imprecision resulting from the update time of the time base used in the instruction:




⇢ = Time base update

⇢ = Call time of an instruction with time processing

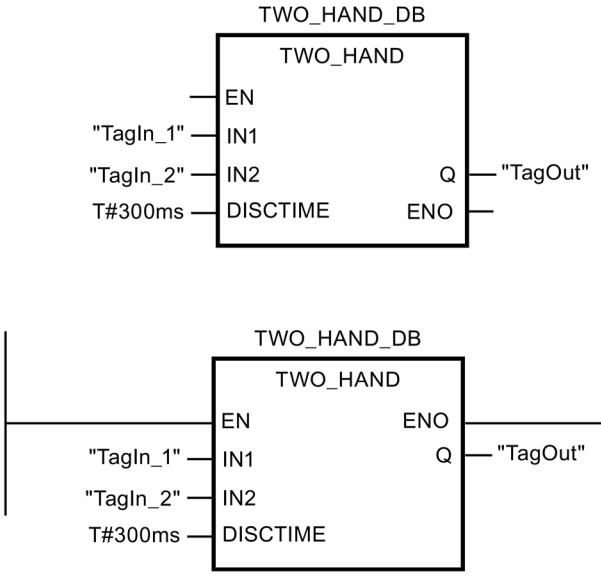
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:




### 13.2.3.3 TWO_H_EN: Two-hand monitoring with enable (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction implements two-hand monitoring with enable.

If pushbuttons IN1 and IN2 are activated within the permitted discrepancy time DISCTIME ≤ 500 ms (IN1/IN2 = 1) (synchronous activation), output signal Q is set to 1 when existing ENABLE = 1. If the time difference between activation of pushbutton IN1 and pushbutton IN2 is greater than DISCTIME, then the pushbuttons must be released and reactivated.

Q is reset to 0 as soon as one of the pushbuttons is released (IN1/IN 2 = 0) or ENABLE = 0. Enable signal Q can be reset to 1 only if the other pushbutton has been released, and if both pushbuttons are then reactivated within the discrepancy time when existing ENABLE = 1.

Every call of the "Two-hand monitoring with enable" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., TWO_H_EN_DB_1) or a multi-instance (e.g., TWO_H_EN_Instance_1) for the "Two-hand monitoring with enable" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

The instruction supports the requirements in accordance with EN 574:1996 + A1:2008.

**Note:** Only one signal per pushbutton can be evaluated in the instruction. Discrepancy monitoring of the NC and NO contacts of pushbuttons IN1 and IN2 is performed directly during suitable configuration (type of sensor interconnection: 2-channel non equivalent) directly through the F-I/O with inputs. The normally open contact must be wired in such a way that it supplies the useful signal (see manual for the F-I/O you are using). In order to keep the discrepancy time from influencing the response time, during the configuration of discrepancy behavior, you must configure "Supply value 0".

If a discrepancy is detected, a fail-safe value of 0 is entered in the process image input (PII) for the pushbutton and QBAD or QBAD_I_xx = 1 or value status = 0 is set in the corresponding F-I/O DB.

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | BOOL | Pushbutton 1 |
| IN2 | Input | BOOL | Pushbutton 2 |
| ENABLE | Input | BOOL | Enable input |
| DISCTIME | Input | TIME | Discrepancy time (0 to 500 ms) |
| Q | Output | BOOL | 1=Enable |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

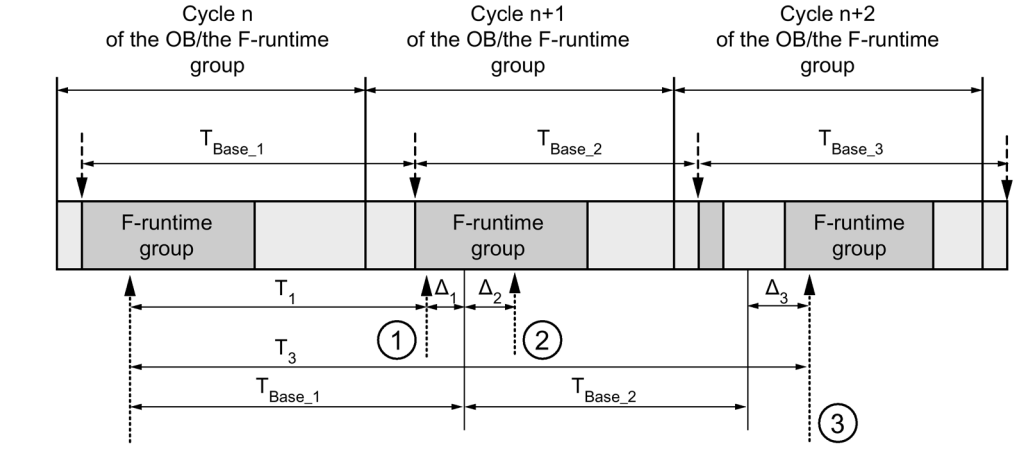
## Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits 0 to 5 are saved until the cause of the error has been eliminated.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Incorrect discrepancy time DISCTIME setting | Discrepancy time setting is <0 or > 500 ms | Set discrepancy time in range of 0 to 500 ms |
| Bit 1 | Discrepancy time elapsed | Discrepancy time setting is too low | If necessary, set a higher discrepancy time |
| | | Pushbuttons were not activated within the discrepancy time | Release pushbuttons and activate them within the discrepancy time |
| | | Wiring fault | Check wiring of pushbuttons |
| | | Pushbuttons defective | Check pushbuttons |
| | | Pushbuttons are wired to different F-I/O, and F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON on an F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 2 | Reserved | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Incorrect activation sequence | One pushbutton was not released | Release pushbuttons and activate them within the discrepancy time |
| | | Pushbuttons defective | Check pushbuttons |
| Bit 5 | ENABLE does not exist | ENABLE = 0 | Set ENABLE = 1, release pushbutton and activate it within the discrepancy time |
| Bit 6 | Reserved | — | — |
| Bit 7 | State of output Q | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




⇢ = Time base update

⇢ = Call time of an instruction with time processing

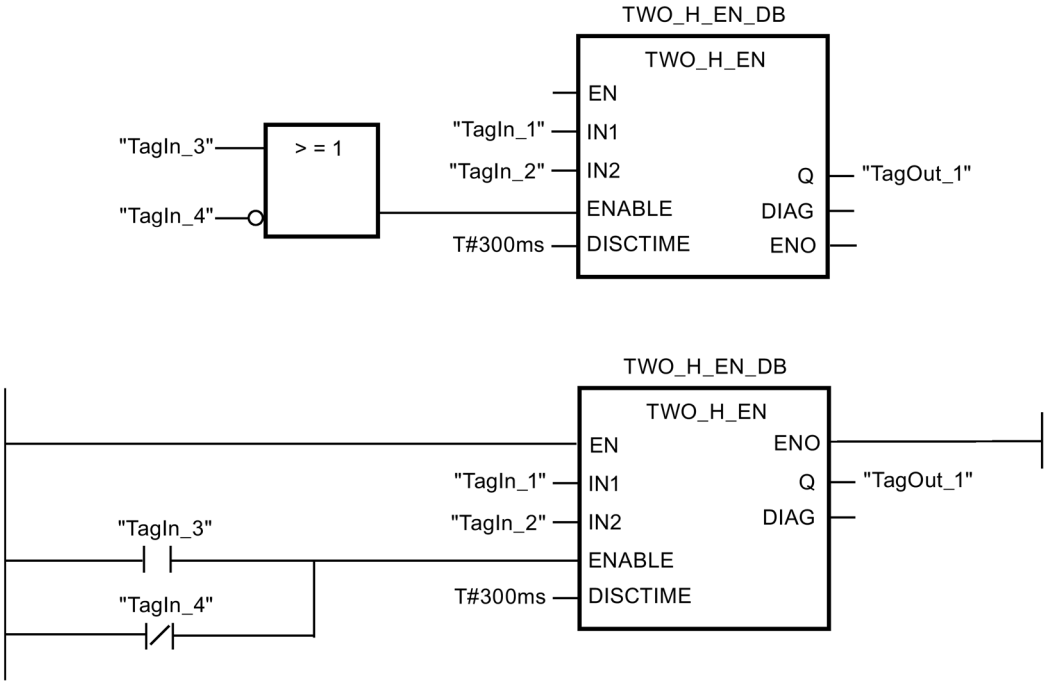
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.2.3.4 MUTING: Muting (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

This instruction performs parallel muting with two or four muting sensors.

**Note**

This instruction is only available for S7-300 and S7-400 F-CPUs. For S7-1200/1500 F-CPUs, you use the instruction "Parallel muting (Page 373)". The instruction "Parallel muting" replaces the instruction "Muting" with compatible functions.

Muting is a defined suppression of the protective function of light curtains. Light curtain muting can be used to introduce goods or objects into the danger area monitored by the light curtain without causing the machine to stop.

To utilize the muting function, at least two independently wired muting sensors must be present. The use of two or four muting sensors and correct integration into the production sequence must ensure that no persons enter the danger area while the light curtain is muted.

Every call of the "Muting" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., MUTING_DB_1) or a multi-instance (e.g., MUTING_Instance_1) for the "Muting" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

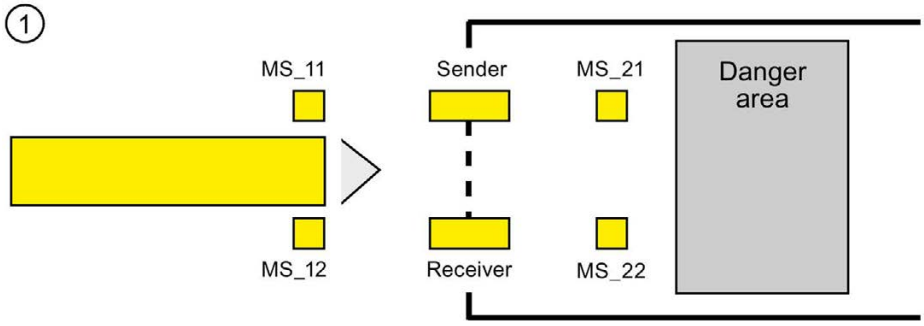
You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| MS_11 | Input | BOOL | Muting sensor 1 of sensor pair 1 |
| MS_12 | Input | BOOL | Muting sensor 2 of sensor pair 1 |
| MS_21 | Input | BOOL | Muting sensor 1 of sensor pair 2 |
| MS_22 | Input | BOOL | Muting sensor 2 of sensor pair 2 |
| STOP | Input | BOOL | 1=Conveyor system stopped |
| FREE | Input | BOOL | 1=Light curtain uninterrupted |
| QBAD_MUT | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal of the muting lamp channel |
| DISCTIM1 | Input | TIME | Discrepancy time of sensor pair 1 (0 to 3 s) |
| DISCTIM2 | Input | TIME | Discrepancy time of sensor pair 2 (0 to 3 s) |
| TIME_MAX | Input | TIME | Maximum muting time (0 to 10 min) |
| ACK | Input | BOOL | Acknowledgment of restart inhibit |
| Q | Output | BOOL | 1= Enable, not off |
| MUTING | Output | BOOL | Display of muting is active |
| ACK_REQ | Output | BOOL | Acknowledgment necessary |
| FAULT | Output | BOOL | Group error |
| DIAG | Output | BYTE | Service information |

### Schematic sequence of error-free muting procedure with 4 muting sensors (MS_11, MS_12, MS_21, MS_22)

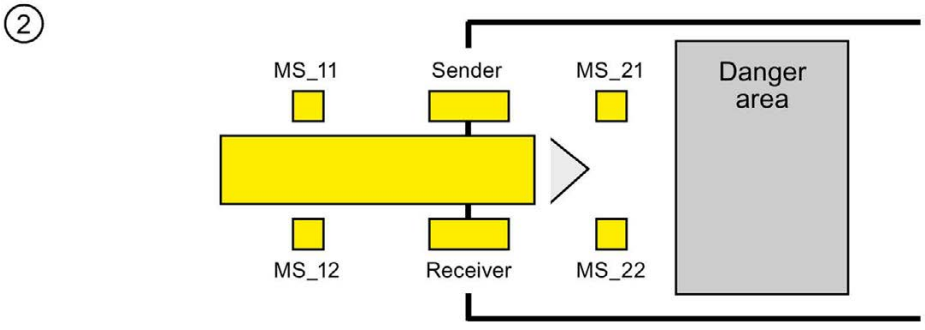



- If both muting sensors MS_11 and MS_12 are activated by the product within DISCTIM1 (apply signal state = 1), the instruction starts the MUTING function. Enable signal Q remains 1, even when input FREE = 0 (light curtain interrupted by product). The MUTING output for setting the muting lamp switches to 1.

**Note**

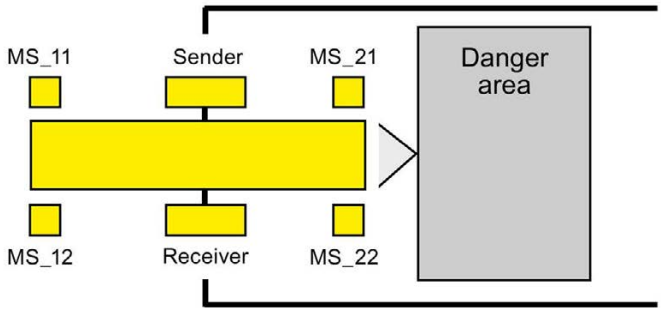
The muting lamp can be monitored using the QBAD_MUT input. To do this, you must wire the muting lamp to an output with wire break monitoring of an F-I/O and supply the QBAD_MUT input with the QBAD signal of the associated F-I/O or the QBAD_O_xx signal of the associated channel. If QBAD_MUT = 1, muting is terminated by the instruction. If monitoring of the muting lamp is not necessary, you do not have to supply input QBAD_MUT.

F-I/O that can promptly detect a wire break after activation of the muting operation must be used (*see manual for specific F-I/O*).



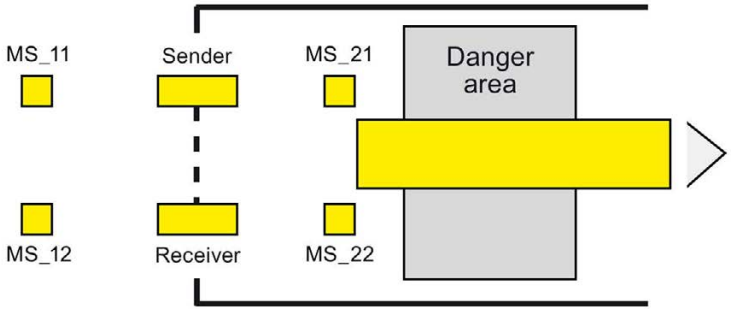

- As long as both muting sensors MS_11 and MS_12 continue to be activated, the MUTING function of the instruction causes Q to remain 1 and MUTING to remain 1 (so that the product can pass through the light curtain without causing the machine to stop).

③




- The two muting sensors MS_21 and MS_22 must be activated (within DISCTIM2) before muting sensors MS_11 and MS_12 are switched to inactive (apply signal state 0). In this way, the instruction retains the MUTING function. (Q = 1, MUTING = 1).

④

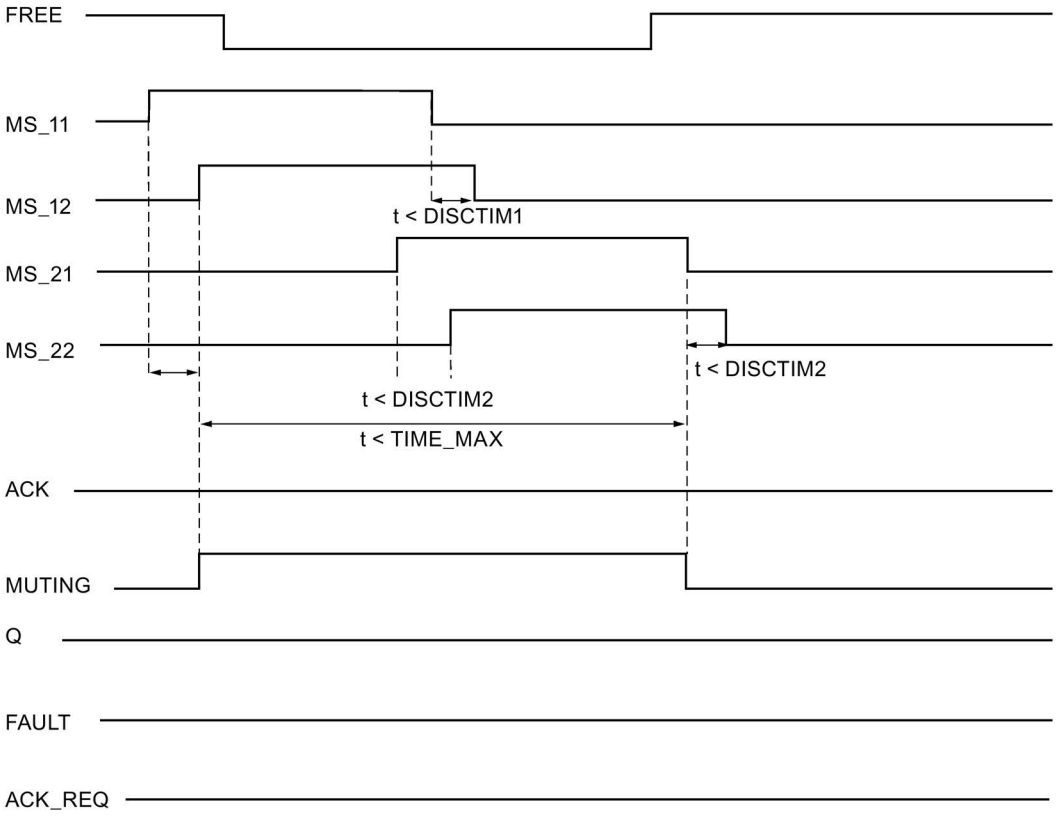



- Only if one of the two muting sensors MS_21 and MS_22 is switched to inactive (product enables sensors) is the MUTING function terminated (Q = 1, MUTING = 0). The maximum activation time for the MUTING function is the time set at input TIME_MAX.

**Note**

The MUTING function is also started if the product passes the light curtain in the reverse direction and the muting sensors are thus activated by the product in reverse order.

## Timing diagrams for error-free muting procedure with 4 muting sensors

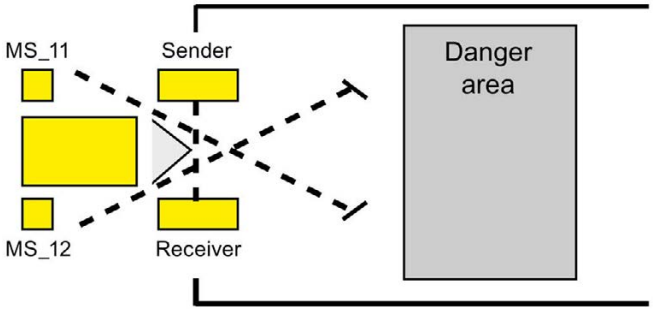



## Schematic sequence of muting procedure with reflection light barriers

If reflection light barriers are used as muting sensors, they are generally arranged diagonally.

In general, this arrangement of reflection light barriers as muting sensors requires only two light barriers, and only MS_11 and MS_12 are interconnected.

The sequence is similar to that of the muting procedure with 4 muting sensors. Step 3 is omitted. In step 4, replace MS_21 and MS_22 with MS_11 and MS_12, respectively.



### Restart inhibit upon interruption of light curtain (if MUTING is not active), when errors occur, and during F-system startup

Enable signal Q cannot be set to 1 or becomes 0, if:

- Light curtain is interrupted (e.g., by a person or material transport) while the MUTING function is not active
- The muting lamp monitoring function responds at input QBAD_MUT
- Sensor pair 1 (MS_11 and MS_12) or sensor pair 2 (MS_21 and MS_22) is not activated or deactivated during discrepancy time DISCTIM1 or DISCTIM2, respectively
- The MUTING function is active longer than the maximum muting time TIME_MAX
- Discrepancy times DISCTIM1 and DISCTIM2 have been set to values < 0 or > 3 s
- Maximum muting time TIME_MAX has been set to a value< 0 or > 10 min

In the identified cases, output FAULT (group error) is set to 1 (restart inhibit). If the MUTING function is started, it will be terminated and the Muting output becomes 0.

**⚠ WARNING**

When a valid combination of muting sensors is immediately detected at startup of the F-system (for example, because the muting sensors are interconnected to inputs of a standard I/O that immediately provide process values during the F-system startup), the MUTING function is immediately started and the MUTING output and enable signal Q are set to 1. The FAULT output (group error) is not set to 1 (no restart inhibit!). *(S035)*

### Acknowledgment of restart inhibit

Enable signal Q becomes 1 again, when:

- The light curtain is no longer interrupted
- Errors, if present, are eliminated (see output DIAG)

  and

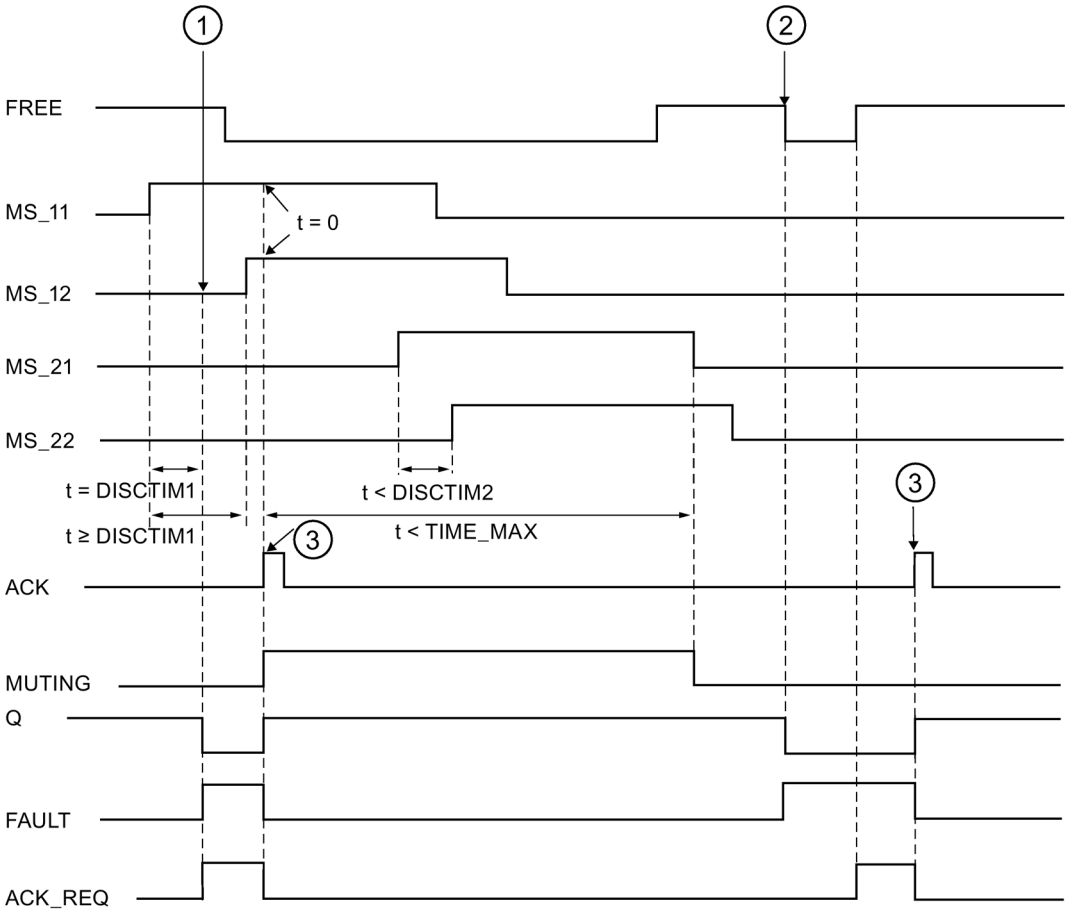
- A user acknowledgment with positive edge occurs at input ACK (see also Implementation of user acknowledgment (Page 151)).

The FAULT output is set to 0. Output ACK_REQ = 1 signals that user acknowledgment at input ACK is required to eliminate the restart inhibit. The instruction sets ACK-REQ = 1 as soon as the light curtain is no longer interrupted or errors have been eliminated. Once acknowledgment has occurred, the instruction resets ACK_REQ to 0.

**Note**

Following discrepancy errors and once the maximum muting time has been exceeded, ACK_REQ is immediately set to 1. As soon as a user acknowledgment has taken place at input ACK, discrepancy times DISCTIM1 and DISCTIM2 and maximum muting time TIME_MAX are reset.

## Timing diagrams for discrepancy errors at sensor pair 1 or interruption of the light curtain (if MUTING is not active)




① Sensor pair 1 (MS_11 and MS_12) is not activated within discrepancy time DISCTIM1.

② The light curtain is interrupted even though the MUTING function is not active.

③ Acknowledgment

### Behavior with stopped conveyor equipment

If, while the conveyor equipment has stopped, the monitoring for one of the following reasons is to be deactivated:

- To comply with discrepancy time DISCTIM1 or DISCTIM2
- To comply with maximum muting time TIME_MAX

you must supply input STOP with a "1" signal for as long as the conveyor equipment is stopped. As soon as the conveyor equipment is running again (STOP = 0), discrepancy times DISCTIM1 and DISCTIM2 and maximum muting time TIME_MAX are reset.

**⚠ WARNING**

When STOP = 1, the discrepancy monitoring is shut down. During this time, if inputs MSx1/MSx2 of a sensor pair both take a signal state of 1 due to an undetected error, e.g., because both muting sensors fail to 1, the error is not detected and the MUTING function can be started unintentionally. *(S036)*

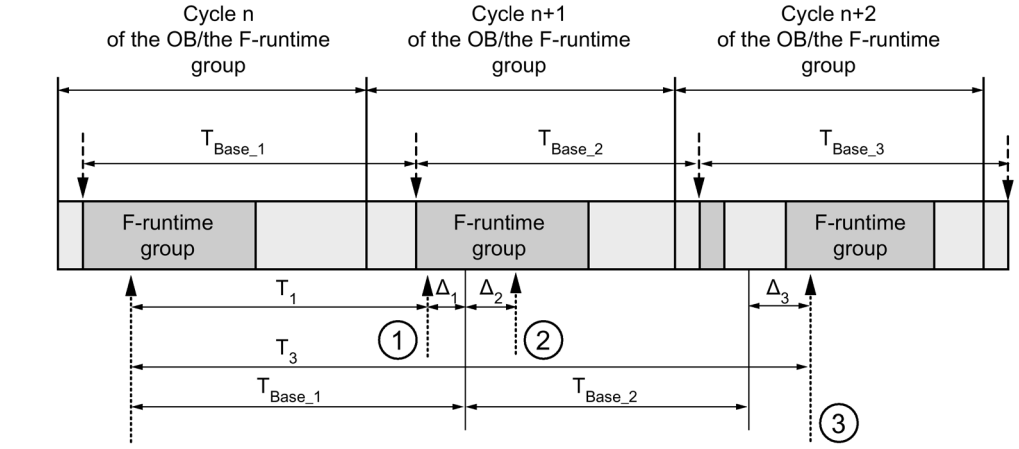
### Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Discrepancy error or incorrect discrepancy time DISCTIM 1 setting for sensor pair 1 | Malfunction in production sequence | Malfunction in production sequence eliminated |
| | | Sensor defective | Check sensors |
| | | Wiring fault | Check wiring of sensors |
| | | Sensors are wired to different F-I/O, and F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON on an F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Discrepancy time setting is too low | If necessary, set a higher discrepancy time |
| | | Discrepancy time setting is < 0 s or > 3 s | Set discrepancy time in range between 0 s and 3 s |
| Bit 1 | Discrepancy error or incorrect discrepancy time DISCTIM 2 setting for sensor pair 2 | Same as Bit 0 | Same as Bit 0 |
| Bit 2 | Maximum muting time exceeded or incorrect muting time TIME_MAX setting | Malfunction in production sequence | Malfunction in production sequence eliminated |
| | | Maximum muting time setting is too low | If necessary, set a higher maximum muting time |
| | | Muting time setting is < 0 s or > 10 min | Set muting time in range from 0 s to 10 min |
| Bit 3 | Light curtain interrupted and muting not active | Light curtain is defective | Check light curtain |
| | | Wiring fault | Check wiring of light curtain (FREE input) |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of light curtain (FREE input) | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | See other DIAG bits | |
| Bit 4 | Muting lamp is defective or cannot be set | Muting lamp is defective | Replace muting lamp |
| | | Wiring fault | Check wiring of muting lamp |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of muting lamp | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 5 | Reserved | — | — |
| Bit 6 | Reserved | — | — |
| Bit 7 | Reserved | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




⇢ = Time base update

⇢ = Call time of an instruction with time processing

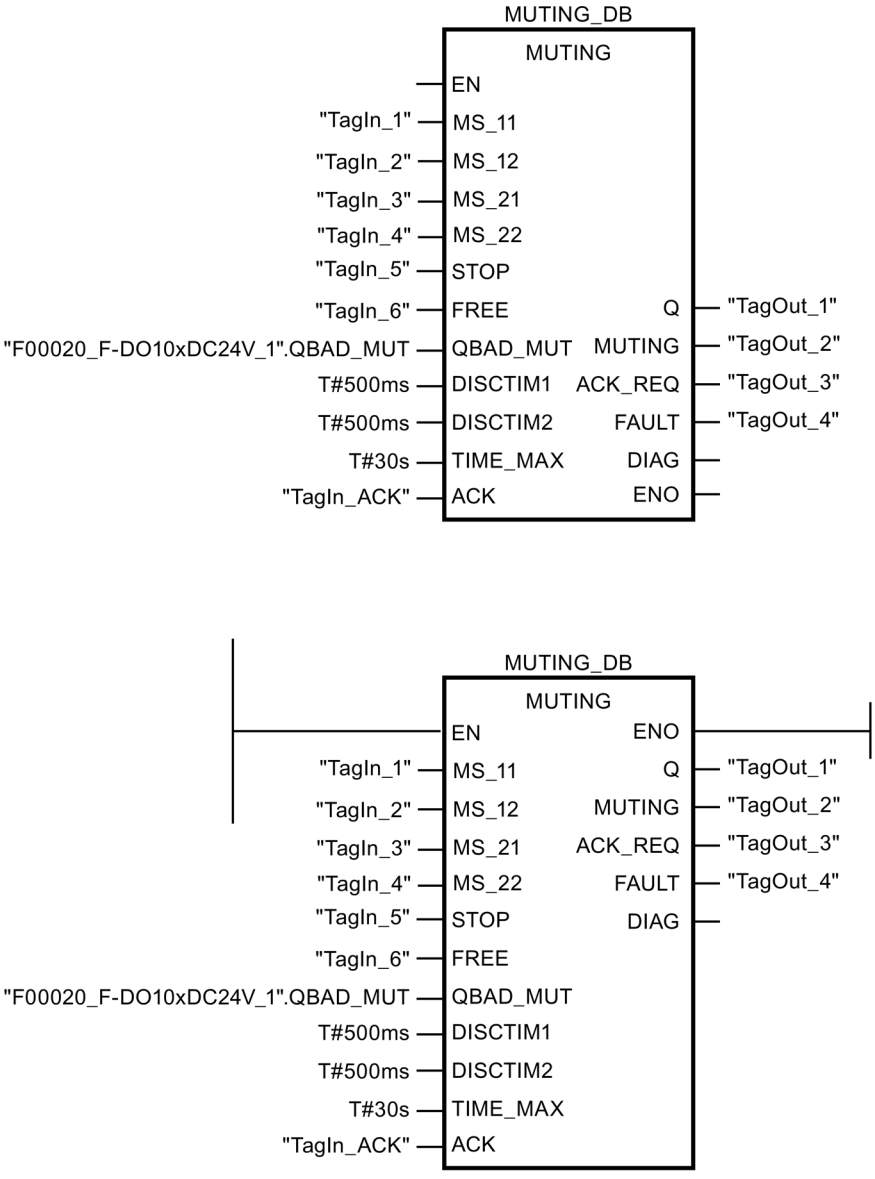
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.2.3.5 MUT_P: Parallel muting (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction performs parallel muting with two or four muting sensors.

Muting is a defined suppression of the protective function of light curtains. Light curtain muting can be used to introduce goods or objects into the danger area monitored by the light curtain without causing the machine to stop.

To utilize the muting function, at least two independently wired muting sensors must be present. The use of two or four muting sensors and correct integration into the production sequence must ensure that no persons enter the danger area while the light curtain is muted.

Every call of the "Parallel muting" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., MUT_P_DB_1) or a multi-instance (e.g., MUT_P_Instance_1) for the "Parallel muting" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| MS_11 | Input | BOOL | Muting sensor 11 |
| MS_12 | Input | BOOL | Muting sensor 12 |
| MS_21 | Input | BOOL | Muting sensor 21 |
| MS_22 | Input | BOOL | Muting sensor 22 |
| STOP | Input | BOOL | 1=Conveyor system stopped |
| FREE | Input | BOOL | 1=Light curtain uninterrupted |
| ENABLE | Input | BOOL | 1=Enable MUTING |
| QBAD_MUT | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal / inverted value status of the muting lamp channel |
| ACK | Input | BOOL | Acknowledgment of restart inhibit |
| DISCTIM1 | Input | TIME | Discrepancy time of sensor pair 1 (0 to 3 s) |
| DISCTIM2 | Input | TIME | Discrepancy time of sensor pair 2 (0 to 3 s) |
| TIME_MAX | Input | TIME | Maximum muting time (0 to 10 min) |
| Q | Output | BOOL | 1= Enable, not off |
| MUTING | Output | BOOL | Display of muting is active |
| ACK_REQ | Output | BOOL | Acknowledgment necessary |
| FAULT | Output | BOOL | Group error |
| DIAG | Output | WORD | Service information |

## Instruction versions

A number of versions are available for this instruction:

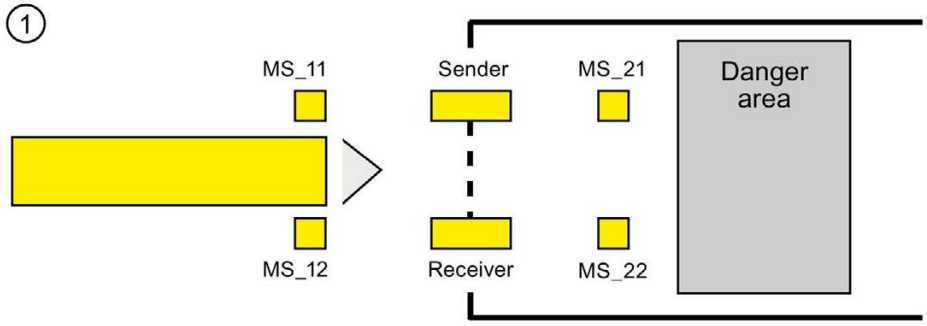
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x* | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x* | — | — | These versions have identical functions to version V1.0.<br>The output DIAG can now be correctly interconnected with the operand of data type WORD. |
| 1.2 | x* | — | x | |
| 1.3 | x* | x | x | |

* S7-300/400: When a restart inhibit (output FAULT = 1) and ENABLE = 1 is present, output ACK_REQ is set to 1 even if not at least one muting sensor is activated. Use the DIAG bits 5 and 6 for additional information.

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

### Schematic sequence of error-free muting procedure with 4 muting sensors (MS_11, MS_12, MS_21, MS_22)

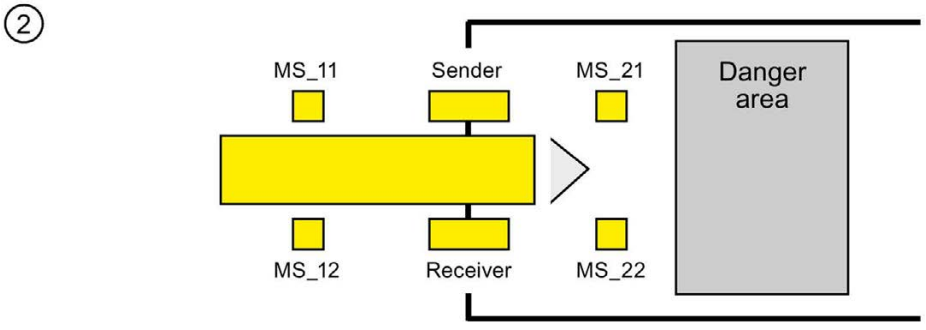



- If muting sensors MS_11 and MS_12 are both activated by the product within DISCTIM1 (apply signal state = 1) and MUTING is enabled by setting the ENABLE input to 1, the instruction starts the MUTING function. Enable signal Q remains 1, even when input FREE = 0 (light curtain interrupted by product). The MUTING output for setting the muting lamp switches to 1.

**Note**

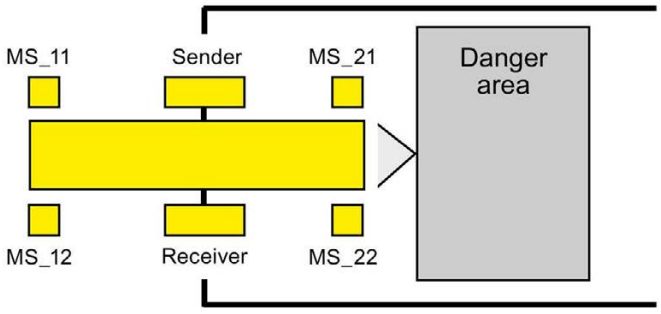
The muting lamp can be monitored using the QBAD_MUT input. To do this, you must wire the muting lamp to an output with wire break monitoring of an F-I/O and supply the QBAD_MUT input with the QBAD signal of the associated F-I/O or the QBAD_O_xx signal / with inverted value statues of the associated channel. If QBAD_MUT = 1, muting is terminated by the instruction. If monitoring of the muting lamp is not necessary, you do not have to supply input QBAD_MUT.

F-I/O that can promptly detect a wire break after activation of the muting operation must be used (*see manual for specific F-I/O*).



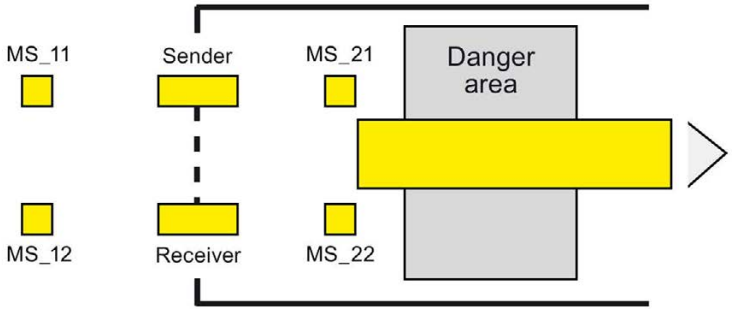

- As long as both muting sensors MS_11 and MS_12 continue to be activated, the MUTING function of the instruction causes Q to remain 1 and MUTING to remain 1 (so that the product can pass through the light curtain without causing the machine to stop). Each of the two muting sensors MS_11 and MS_12 may be switched to inactive (t < DISCTIM1) for a short time (apply signal state 0).

③




- Muting sensors MS_21 and MS_22 must both be activated (within DISCTIM2) before muting sensors MS_11 and MS_12 are switched to inactive (apply signal state 0). In this way, the instruction retains the MUTING function. (Q = 1, MUTING = 1).

④

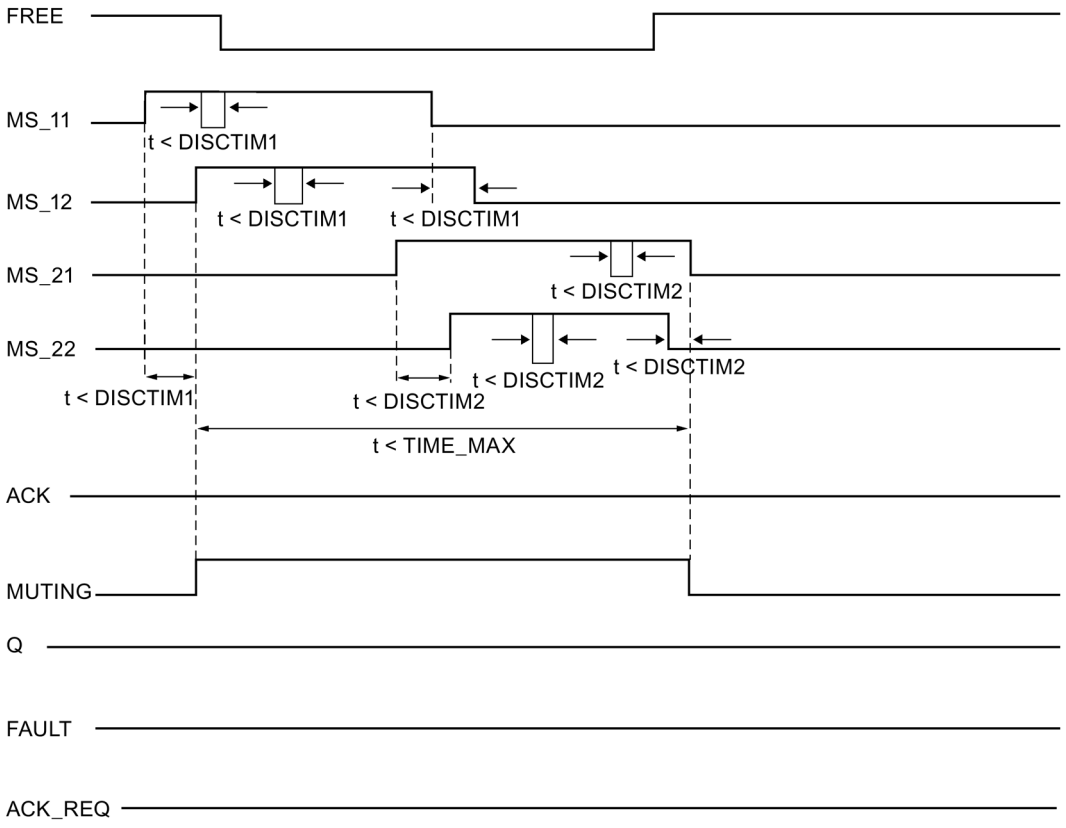



Only if muting sensors MS_21 and MS_22 are both switched to inactive (product enables sensors) is the MUTING function terminated (Q = 1, MUTING = 0). The maximum activation time for the MUTING function is the time set at input TIME_MAX.

**Note**

The MUTING function is also started if the product passes the light curtain in the reverse direction and the muting sensors are thus activated by the product in reverse order.

## Timing diagrams for error-free muting procedure with 4 muting sensors

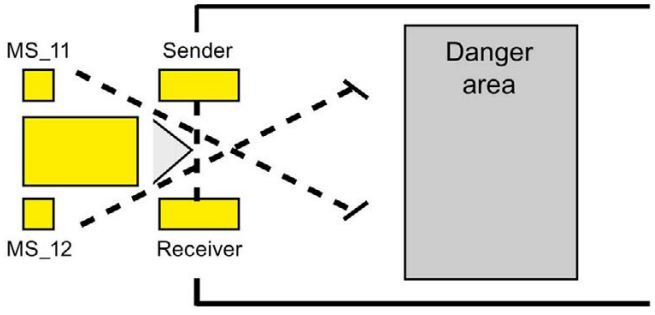



## Schematic sequence of muting procedure with reflection light barriers

If reflection light barriers are used as muting sensors, they are generally arranged diagonally.

In general, this arrangement of reflection light barriers as muting sensors requires only two light barriers, and only MS_11 and MS_12 are interconnected.

The sequence is similar to that of the muting procedure with 4 muting sensors. Step 3 is omitted. In step 4, replace MS_21 and MS_22 with MS_11 and MS_12, respectively.



### Restart inhibit upon interruption of light curtain (MUTING is not active), as well as when errors occur and during F-system startup

Enable signal Q cannot be set to 1 or becomes 0, if:

- Light curtain is interrupted (e.g., by a person or material transport) while the MUTING function is not active
- Light curtain is (being) interrupted and the muting lamp monitoring at input QBAD_MUT is set to 1
- Light curtain is (being) interrupted and the MUTING function is not enabled by setting input ENABLE to 1
- Sensor pair 1 (MS_11 and MS_12) or sensor pair 2 (MS_21 and MS_22) is not activated or deactivated during discrepancy time DISCTIM1 or DISCTIM2, respectively
- The MUTING function is active longer than the maximum muting time TIME_MAX
- Discrepancy times DISCTIM1 and DISCTIM2 have been set to values < 0 or > 3 s
- Maximum muting time TIME_MAX has been set to a value< 0 or > 10 min
- The F-system starts up (regardless of whether or not the light curtain is interrupted, because the F-I/O is passivated after F-system startup and, thus, the FREE input is initially supplied with 0)

In the identified cases, output FAULT (group error) is set to 1 (restart inhibit). If the MUTING function is started, it will be terminated and the Muting output becomes 0.

### User acknowledgment of restart inhibit (no muting sensor is activated or ENABLE = 0)

Enable signal Q becomes 1 again, when:

- The light curtain is no longer interrupted
- Errors, if present, are eliminated (see output DIAG)

  and

- A user acknowledgment with positive edge occurs at input ACK (see also Implementation of user acknowledgment (Page 151)).

The FAULT output is set to 0. Output ACK_REQ = 1 (and DIAG bit 6) signals that user acknowledgment at input ACK is required to eliminate the restart inhibit. The instruction sets ACK_REQ = 1 as soon as the light curtain is no longer interrupted or the errors have been eliminated. Once acknowledgment has occurred, the instruction resets ACK_REQ to 0.

### User Acknowledgment of restart inhibit (at least one muting sensor is activated and ENABLE = 1)

Enable signal Q becomes 1 again, when:

- Errors, if present, are eliminated (see output DIAG)
- FREE occurs until a valid combination of muting sensors is detected

The FAULT output is set to 0. The MUTING function is restarted, if necessary, and the MUTING output becomes 1 if a valid combination of muting sensors is detected. When ENABLE = 1, output ACK_REQ = 1 (and DIAG bit 5) signals that FREE is necessary for error elimination and for removal of the restart inhibit.*After successful FREE, ACK_REQ is reset to 0 by the instruction.

**Note**

Once the maximum muting time is exceeded, TIME_MAX is reset as soon as the MUTING function is restarted.

### FREE function

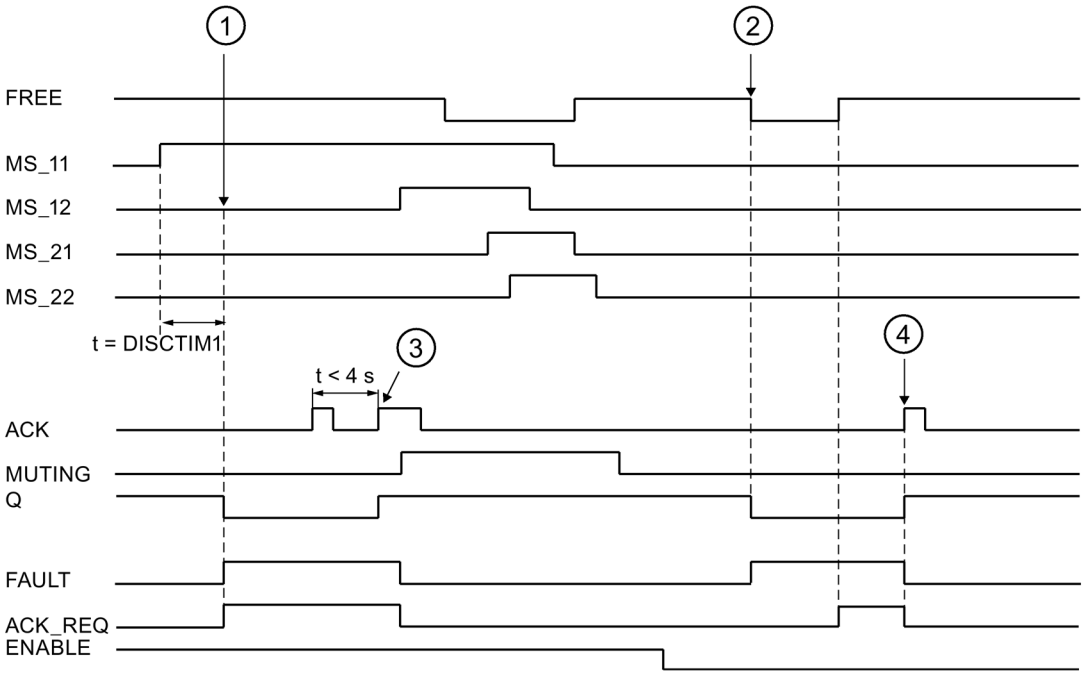
If an error cannot be corrected immediately, the FREE function can be used to free the muting range. Enable signal Q and output MUTING =1 temporarily. The FREE function can be used if:

- ENABLE = 1
- At least one muting sensor is activated
- A user acknowledgment with rising edge at input ACK occurs twice within 4 s, and the second user acknowledgment at input ACK remains at a signal state of 1 (acknowledgment button remains activated)

**⚠ WARNING**

When using the FREE function, the action must be observed. A dangerous situation must be able to be interrupted at any time by releasing the acknowledgment button. The acknowledgment button must be mounted in such a way the entire danger area can be observed. *(S037)*

### Timing diagrams for discrepancy errors at sensor pair 1 or interruption of the light curtain (MUTING is not active)




① Sensor pair 1 (MS_11 and MS_22) is not activated within discrepancy time DISCTIM1.

② The light curtain is interrupted even though there is no enable (ENABLE=0)

③ FREE function

④ Acknowledgment

### Behavior with stopped conveyor equipment

If, while the conveyor equipment has stopped, the monitoring for one of the following reasons is to be deactivated:

- To comply with discrepancy time DISCTIM1 or DISCTIM2
- To comply with maximum muting time TIME_MAX

You must supply input STOP with a "1" signal for as long as the conveyor equipment is stopped. As soon as the conveyor equipment is running again (STOP = 0), discrepancy times DISCTIM1 and DISCTIM2 and maximum muting time TIME_MAX are reset.

**⚠ WARNING**

When STOP = 1 or ENABLE = 0, discrepancy monitoring is shut down. During this time, if inputs MSx1/MSx2 of a sensor pair both take a signal state of 1 due to an undetected error, e.g., because both muting sensors fail to 1, the fault is not detected and the MUTING function can be started unintentionally (when ENABLE =1). *(S038)*

## Output DIAG

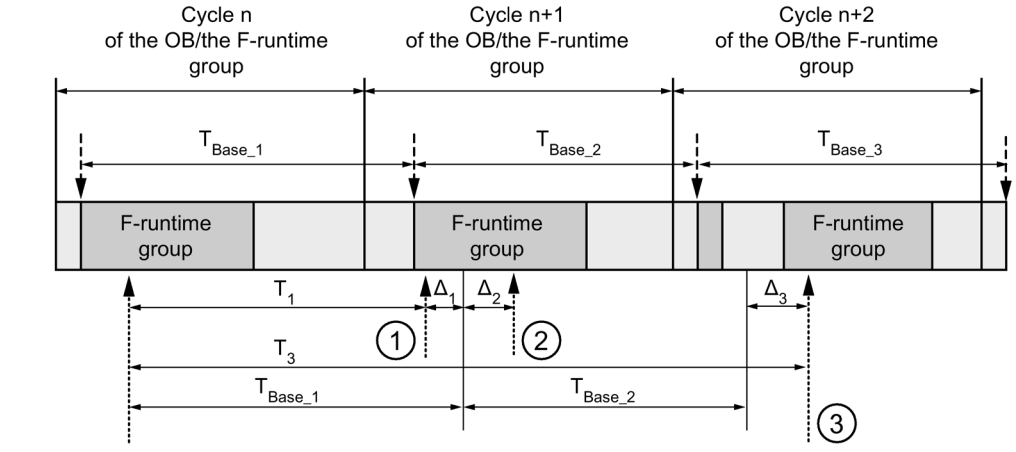
The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits 0 to 6 are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Discrepancy error or incorrect discrepancy time DISCTIM 1 setting for sensor pair 1 | Malfunction in production sequence | Malfunction in production sequence eliminated |
| | | Sensor defective | Check sensors |
| | | Wiring fault | Check wiring of sensors |
| | | Sensors are wired to different F-I/O, and F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON on an F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Discrepancy time setting is too low | If necessary, set a higher discrepancy time |
| | | Discrepancy time setting is < 0 s or > 3 s | Set discrepancy time in range between 0 s and 3 s |
| Bit 1 | Discrepancy error or incorrect discrepancy time DISCTIM 2 setting for sensor pair 2 | Same as Bit 0 | Same as Bit 0 |
| Bit 2 | Maximum muting time exceeded or incorrect muting time TIME_MAX setting | Malfunction in production sequence | Malfunction in production sequence eliminated |
| | | Maximum muting time setting is too low | If necessary, set a higher maximum muting time |
| | | Muting time setting is < 0 s or > 10 min | Set muting time in range from 0 s to 10 min |
| Bit 3 | Light curtain interrupted and muting not active | ENABLE = 0 | Set ENABLE = 1 |
| | | Light curtain is defective | Check light curtain |
| | | Wiring fault | Check wiring of light curtain (FREE input) |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of light curtain<br>(FREE input) | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Startup of F-system | For FREE, see DIAG Bit 5 |
| | | See other DIAG bits | |
| Bit 4 | Muting lamp is defective or cannot be set | Muting lamp is defective | Replace muting lamp |
| | | Wiring fault | Check wiring of muting lamp |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of muting lamp | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 5 | FREE is necessary | See other DIAG bits | Two rising edges at ACK within 4 s, and activate acknowledgment button until ACK_REQ = 0 |
| Bit 6 | Acknowledgment necessary | — | — |
| Bit 7 | State of output Q | — | — |
| Bit 8 | State of output MUTING | — | — |
| Bit 9 | FREE active | — | — |
| Bit 10 | Reserved | — | — |
| ... | | | |
| Bit 15 | Reserved | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




⋯⋯► = Time base update

⋯⋯► = Call time of an instruction with time processing

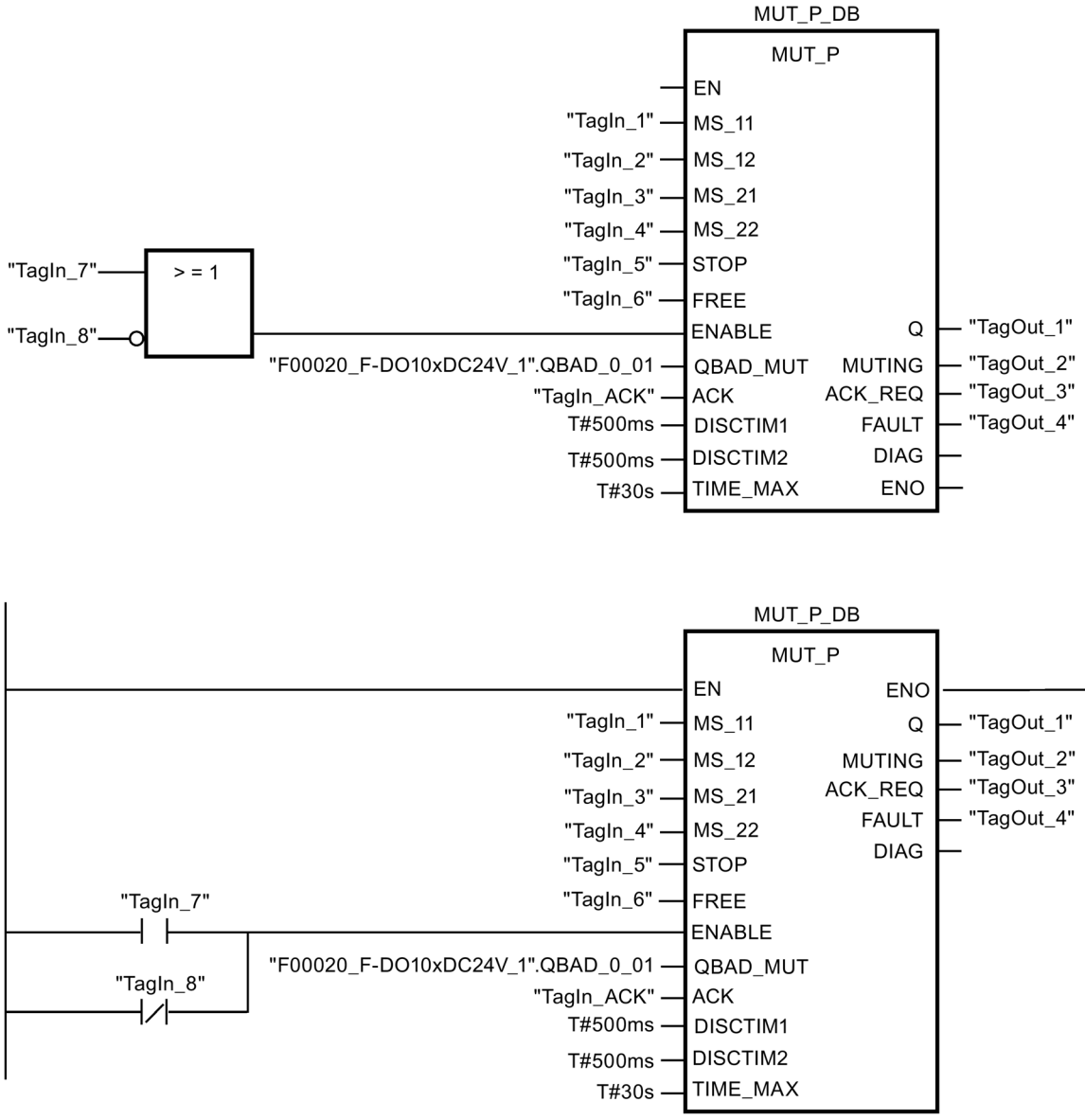
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.2.3.6 EV1oo2DI: 1oo2 evaluation with discrepancy analysis (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction implements a 1oo2 evaluation of two single-channel sensors combined with a discrepancy analysis.

Output Q is set to 1, if the signal states of inputs IN1 and IN2 both equal 1 and no discrepancy error DISC_FLT is stored. if the signal state of one or both inputs is 0, output Q is set to 0.

As soon as the signal states of inputs IN1 and IN2 are different, the discrepancy time DISCTIME is started. If the signal states of the two inputs are still different once the discrepancy time expires, a discrepancy error is detected and DISC_FLT is set to 1 (restart inhibit).

If the discrepancy between inputs IN1 and IN2 is no longer detected, the discrepancy error is acknowledged according to the parameter assignment of ACK_NEC:

- If ACK_NEC = 0, the acknowledgment is automatic.
- If ACK_NEC = 1, you must use a rising edge at input ACK to acknowledge the discrepancy error.

The output ACK_REQ = 1 signals that a user acknowledgment at input ACK is necessary to acknowledge the discrepancy error (cancel the restart inhibit). The instruction sets ACK_REQ = 1 as soon as discrepancy is no longer detected. After acknowledgment or if, prior to acknowledgment, there is once again a discrepancy between inputs IN1 and IN2, the instruction resets ACK_REQ to 0.

Output Q can never be set to 1 if the discrepancy time setting is < 0 or > 60 s. In this case, output DISC_FLT is also set to 1 (restart inhibit). The call interval of the safety program (e.g., OB35) must be less than the discrepancy time setting.

Every call of the "1oo2 evaluation with discrepancy analysis" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., EV1oo2DI_DB_1) or a multi-instance (e.g., EV1oo2DI_Instance_1) for the "1oo2 evaluation with discrepancy analysis" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

The ACK_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S033)*

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | BOOL | Sensor 1 |
| IN2 | Input | BOOL | Sensor 2 |
| DISCTIME | Input | TIME | Discrepancy time (0 to 60 s) |
| ACK_NEC | Input | BOOL | 1 = acknowledgment necessary for discrepancy error |
| ACK | Input | BOOL | Acknowledgment of discrepancy error |
| Q | Output | BOOL | Output |
| ACK_REQ | Output | BOOL | 1 = acknowledgment required |
| DISC_FLT | Output | BOOL | 1 = discrepancy error |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

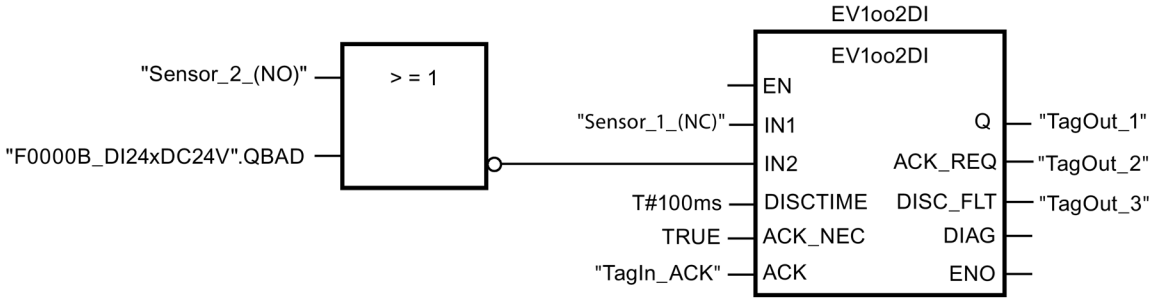
## Activating inputs IN1 and IN2

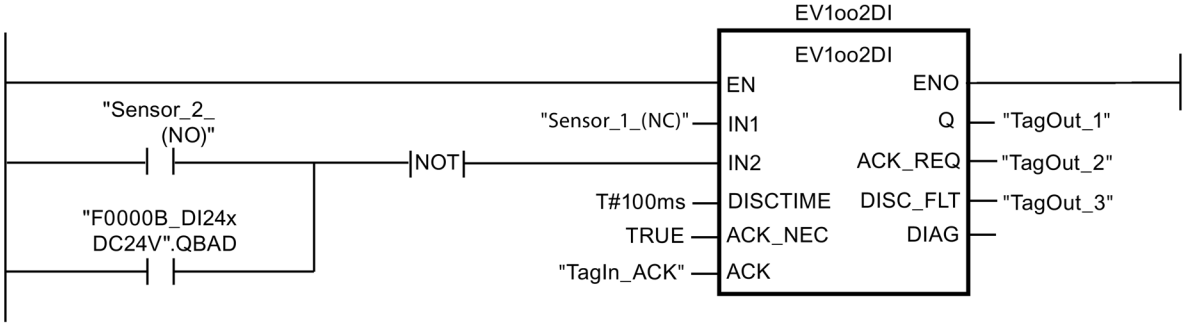
Inputs IN1 and IN2 must both be activated in such a way that their safe state is 0.

## Example with QBAD or QBAD_I_xx signal

For non-equivalent signals you need to OR the input (IN1 and IN2) with which you assign the encoder signal to the safe state 1, with the QBAD signal of the associated F-I/O or the QBAD_I_xx signal of the associated channel (with S7-300/400 F-CPUs) and negate the result. Signal state 0 is then at input IN1 or IN2 when fail-safe values are output.

Network1: EV1oo2DI with non-equivalent signals

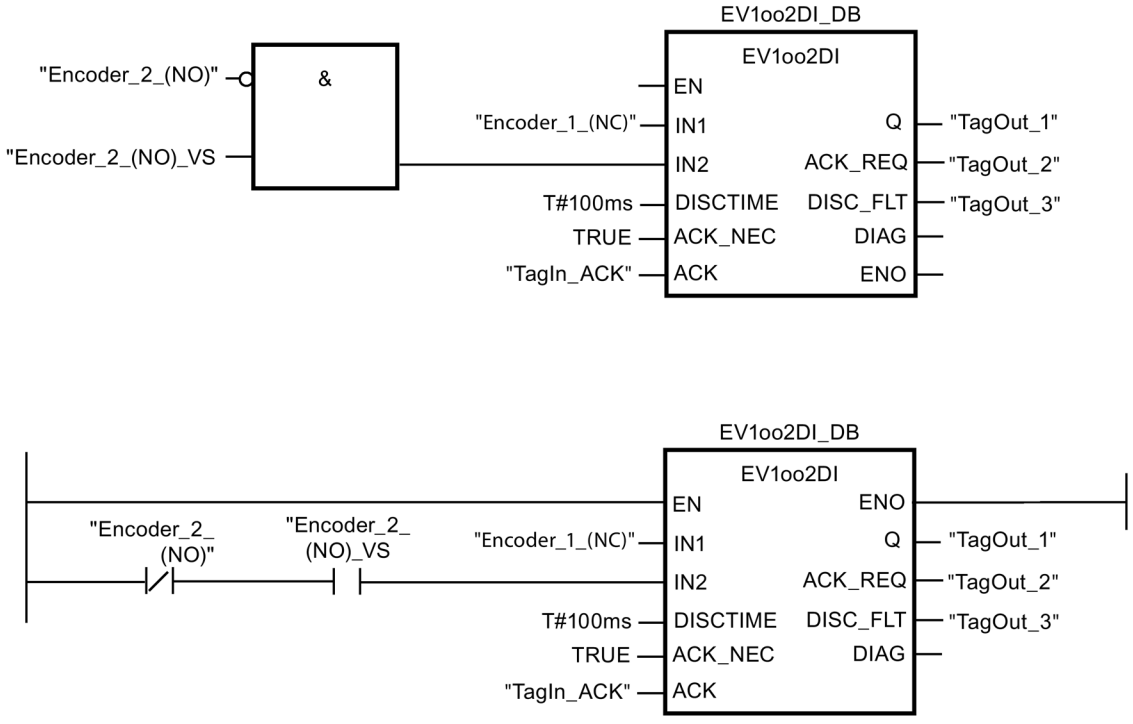


## Example with value status

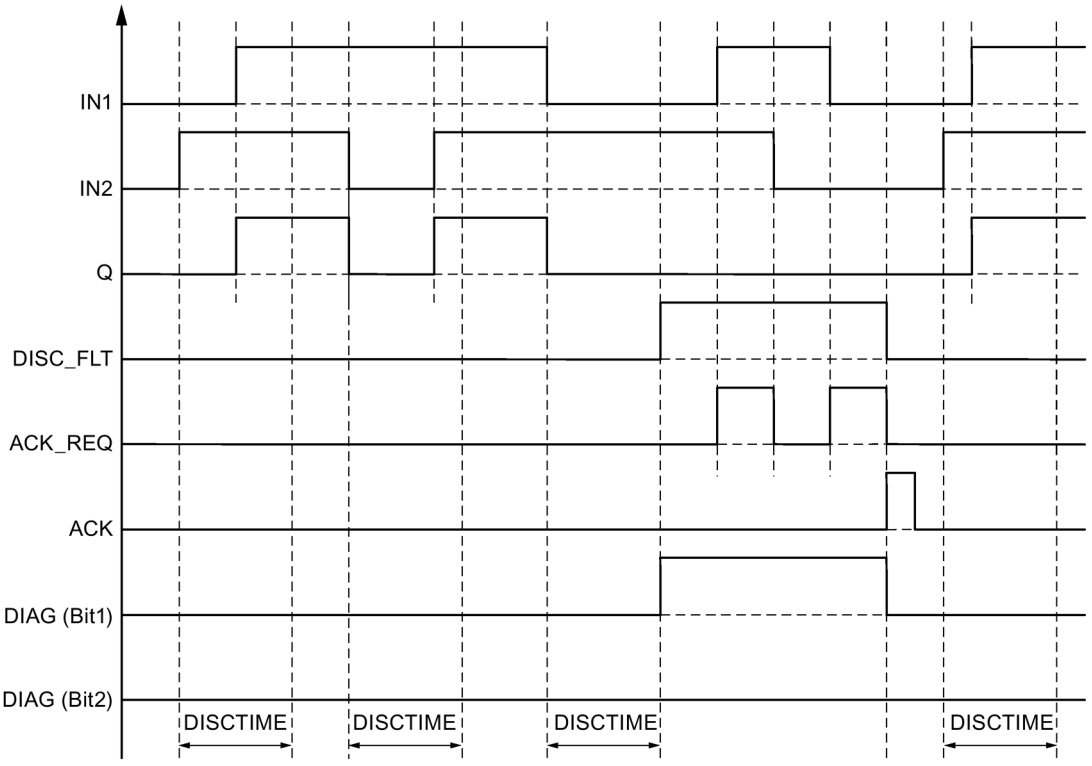
For nonequivalent signals, you have to negate the input (IN1 or IN2) with which you have assigned the encoder signal to a safe state of 1 and AND it with the value status of the associated channel. Signal state 0 is then at input IN1 or IN2 when fail-safe values are output.

Network1: EV1oo2DI with non-equivalent signals



## Timing diagrams EV1oo2DI

If ACK_NEC = 1:




## Startup characteristics

**Note**

If the sensors at inputs IN1 and IN2 are assigned to different F-I/O, it is possible that the fail-safe values are output for different lengths of time following startup of the F-system due to different startup characteristics of the F-I/O. If the signal states of inputs IN1 and IN2 remain different after the discrepancy time DISCTIME has expired, a discrepancy error is detected after the F-system starts up.

If ACK_NEC = 1 you must acknowledge the discrepancy error with a rising edge at input ACK.

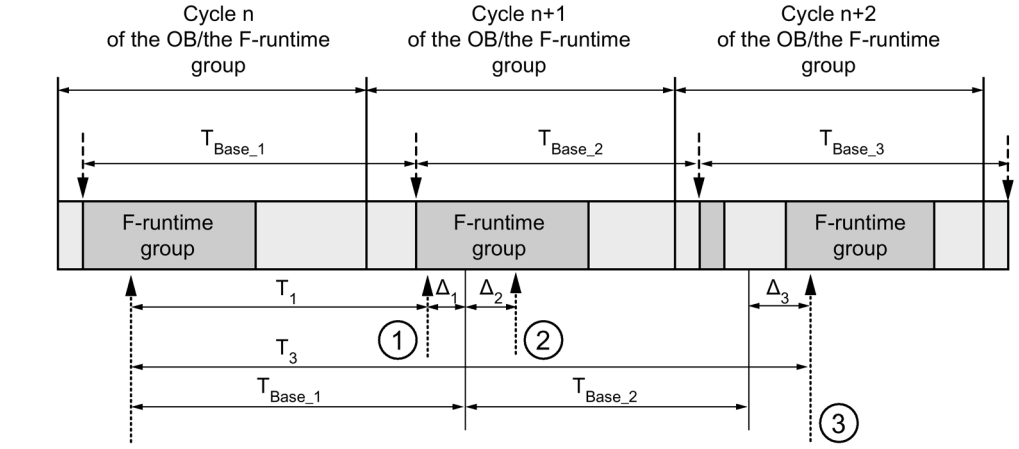
## Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Discrepancy error or incorrect discrepancy time setting (= status of DISC_FLT) | Sensor defective | Check sensors |
| | | Wiring fault | Check wiring of sensors |
| | | Sensors are wired to different F-I/O, and F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON on an F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Discrepancy time setting is too low | If necessary, set a higher discrepancy time |
| | | Discrepancy time setting is < 0 s or > 60 s | Set discrepancy time in range between 0 s and 60 s |
| Bit 1 | For discrepancy errors: last signal state change was at input IN1 | — | — |
| Bit 2 | For discrepancy errors: last signal state change was at input IN2 | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Reserved | — | — |
| Bit 5 | For discrepancy errors: input ACK has a permanent signal state of 1 | Acknowledgment button defective | Replace acknowledgment button |
| | | Wiring fault | Check wiring of acknowledgment button |
| Bit 6 | Acknowledgment necessary | — | — |
| Bit 7 | State of output Q | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




⇢ = Time base update

⇢ = Call time of an instruction with time processing

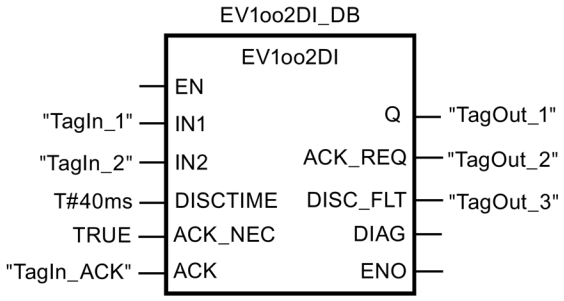
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

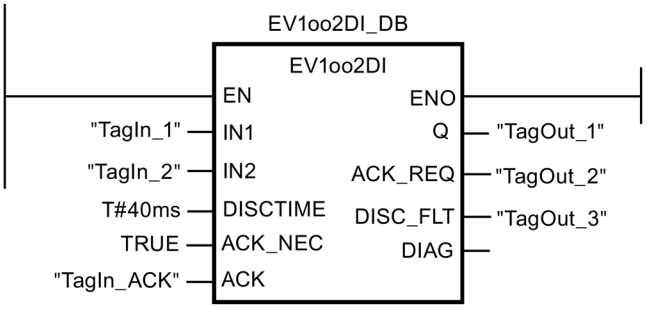
② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.2.3.7 FDBACK: Feedback monitoring (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction implements feedback monitoring.

The signal state of output Q is checked to see whether it corresponds to the inverse signal state of the feedback input FEEDBACK.

Output Q is set to 1 as soon as input ON = 1. Requirement for this is that the feedback input FEEDBACK = 1 and no feedback error is saved.

Output Q is reset to 0, as soon as input ON = 0 or if a feedback error is detected.

A feedback error ERROR = 1 is detected if the inverse signal state of the feedback input FEEDBACK (to input Q) does not follow the signal state of output Q within the maximum tolerable feedback time. The feedback error is saved.

If a discrepancy is detected between the feedback input FEEDBACK and the output Q after a feedback error, the feedback error is acknowledged in accordance with the parameter assignment of ACK_NEC:

- If ACK_NEC = 0, the acknowledgment is automatic.
- If ACK_NEC = 1, you must acknowledge the feedback error with a rising edge at input ACK.

The ACK_REQ = 1 output then signals that a user acknowledgment is necessary at input ACK to acknowledge the feedback error. Following an acknowledgment, the instruction resets ACK_REQ to 0.

To avoid a feedback errors from being detected and acknowledgment from being required when the F-I/O controlled by the Q output are passivated, you need to supply input QBAD_FIO with the QBAD signal of the associated F-I/O or the QBAD_O_xx signal / inverted value status of the associated channel.

Every call of the "Feedback monitoring" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., FDBACK_DB_1) or a multi-instance (e.g., FDBACK_Instance_1) for the "Feedback monitoring" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

The ACK_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S033)*

**WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision *(S034)*.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ON | Input | BOOL | 1= Enable output |
| FEEDBACK | Input | BOOL | Feedback input |
| QBAD_FIO | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal / inverted value status of the Q output |
| ACK_NEC | Input | BOOL | 1=Acknowledgment necessary |
| ACK | Input | BOOL | Acknowledgment |
| FDB_TIME | Input | TIME | Feedback time |
| Q | Output | BOOL | Output |
| ERROR | Output | BOOL | Feedback error |
| ACK_REQ | Output | BOOL | Acknowledgment request |
| DIAG | Output | BYTE | Service information |

## Instruction versions

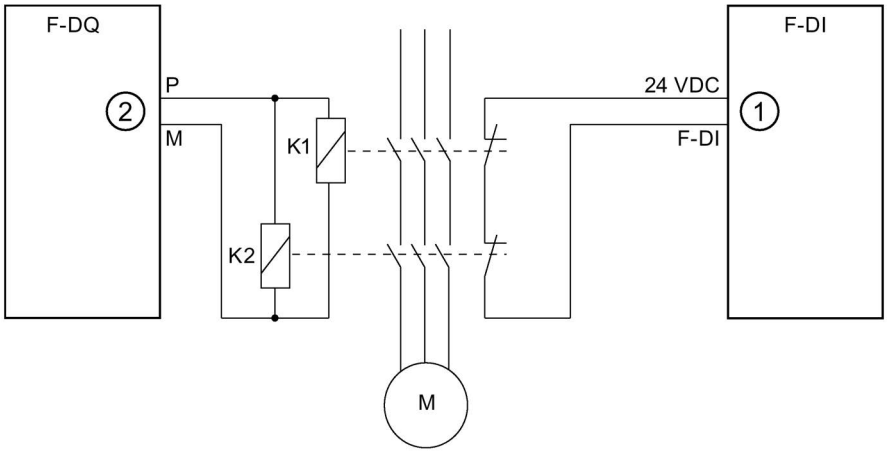
A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | Version 1.0 requires that the F_TOF block with the number FB 186 is available in the project tree in the "Program blocks/System blocks/STEP 7 Safety" folder.<br>When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically. If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. You will then avoid number conflicts. |
| 1.1 | x | — | — | These versions are functionally identical to version V1.0, but do not require the F_TOF block to have a particular number. |
| 1.2 | x | — | x | |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Interconnection example




① Sent to the FEEDBACK input of the instruction
② from output Q of the instruction

## Startup characteristics

After an F-system startup, the instruction does not have be acknowledged when no errors are present.

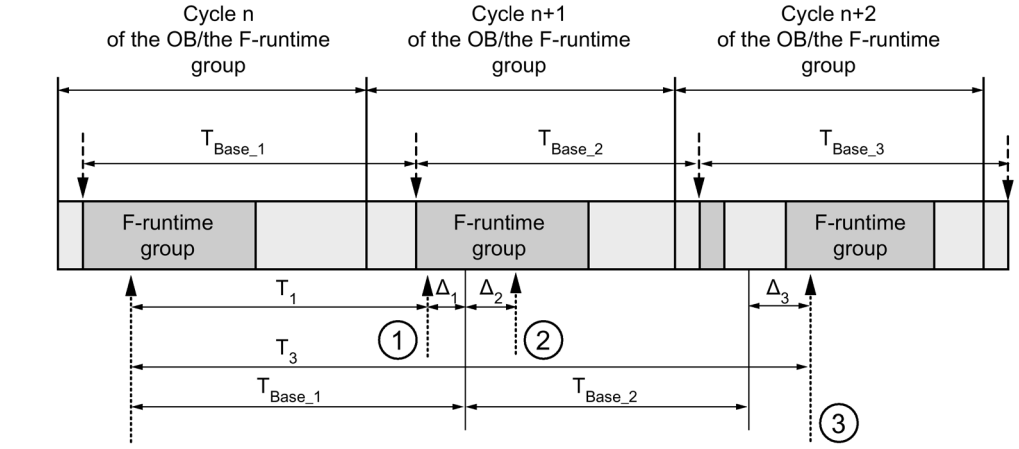
## Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits 0, 2, and 5 are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Feedback error or incorrect feedback time setting (= state of ERROR) | Feedback time setting < 0 | Set feedback time > 0 |
| | | Feedback time setting is too low | If necessary, set a higher feedback time |
| | | Wiring fault | Check wiring of actuator and feedback contact |
| | | Actuator or feedback contact is defective | Check actuator and feedback contact |
| | | I/O fault or channel fault of feedback input | Check I/O |
| Bit 1 | Passivation of F-I/O/channel controlled by output Q (= state of QBAD_FIO) | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 2 | After feedback error: feedback input has permanent signal state of 0 | F-I/O fault or channel fault of feedback input | Check I/O |
| | | Feedback contact is defective | Check feedback contact |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of feedback input | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 3 | Reserved | — | — |
| Bit 4 | Reserved | — | — |
| Bit 5 | For feedback error: input ACK has a permanent signal state of 1 | Acknowledgment button defective | Check acknowledgment button |
| | | Wiring fault | Check wiring of acknowledgment button |
| Bit 6 | Acknowledgment required (= state of ACK_REQ) | — | — |
| Bit 7 | State of output Q | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




⇢ = Time base update

⇢ = Call time of an instruction with time processing

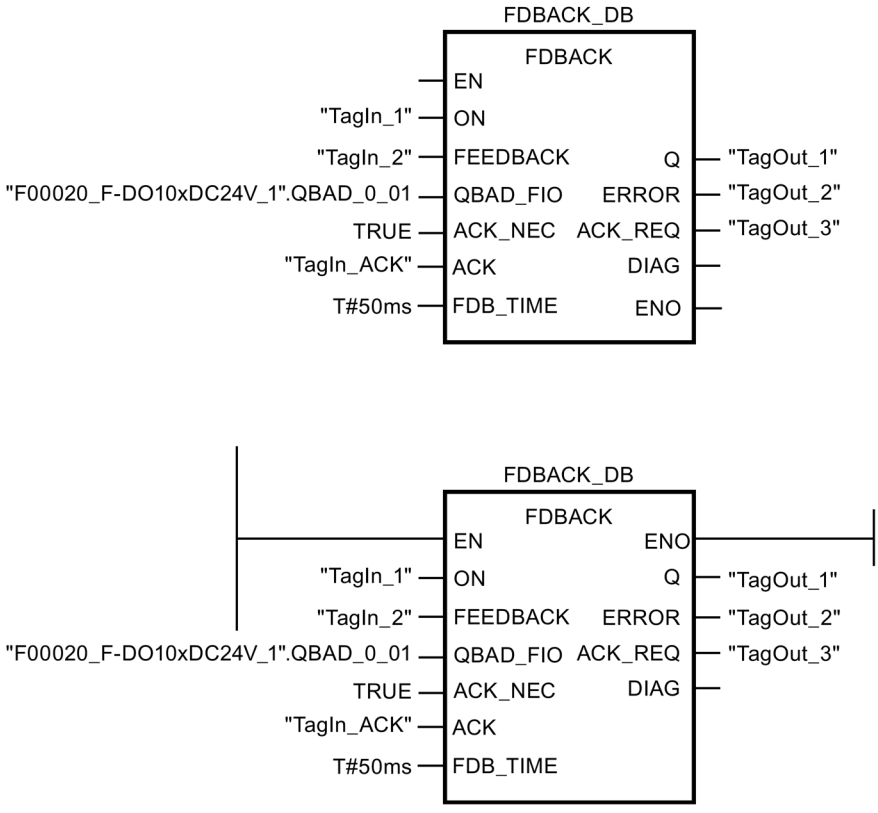
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

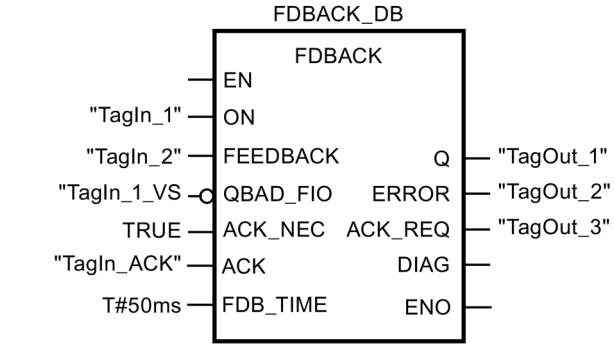
③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

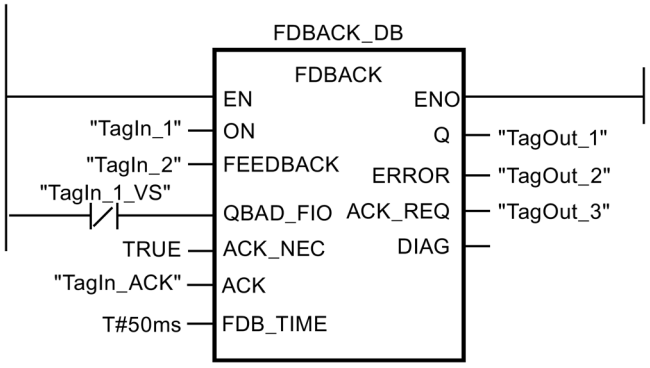
## Example

The following example shows how the instruction for S7-300/400 F-CPUs works:



The following example shows how the instruction for S7-1200/1500 F-CPUs works:



### 13.2.3.8 SFDOOR: Safety door monitoring (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction implements safety door monitoring.

Enable signal Q is reset to 0 as soon as one of the inputs IN1 or IN2 take a signal state of 0 (safety door is opened). The enable signal can be reset to 1, only if:

- Inputs IN1 and IN2 both take a signal state of 0 prior to opening the door (safety door has been completely opened)
- Inputs IN1 and IN2 then both take a signal state of 1 (safety door is closed)
- An acknowledgment occurs

The acknowledgment for the enable takes place according to the parameter assignment at input ACK_NEC:

- If ACK_NEC = 0, the acknowledgment is automatic.
- If ACK_NEC = 1, you must use a rising edge at input ACK for acknowledging the enable.

Output ACK_REQ = 1 is used to signal that a user acknowledgment is required at input ACK for the acknowledgment. The instruction sets ACK_REQ = 1 as soon as the door is closed. Following an acknowledgment, the instruction resets ACK_REQ to 0.

In order for the instruction to recognize whether inputs IN1 and IN2 are 0 merely due to passivation of the associated F-I/O, you need to supply inputs QBAD_IN1 or QBAD_IN2 with the QBAD signal of the associated F-I/O or QBAD_I_xx signal / inverted value status of the associated channel. Among other things, this will prevent you from having to open the safety door completely prior to an acknowledgment in the event the F-I/O are passivated.

Every call of the "Safety door monitoring" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., SFDOOR_DB_1) or a multi-instance (e.g., SFDOOR_Instance_1) for the "Safety door monitoring" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

| **WARNING** |
| --- |
| The ACK_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S033)* |

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | BOOL | Input 1 |
| IN2 | Input | BOOL | Input 2 |
| QBAD_IN1 | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal / inverted value status of the channel of input IN1 |
| QBAD_IN2 | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal / inverted value status of the channel of input IN2 |
| OPEN_NEC | Input | BOOL | 1= Open necessary at startup |
| ACK_NEC | Input | BOOL | 1=Acknowledgment necessary |
| ACK | Input | BOOL | Acknowledgment |
| Q | Output | BOOL | 1= Enable, safety door closed |
| ACK_REQ | Output | BOOL | Acknowledgment request |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

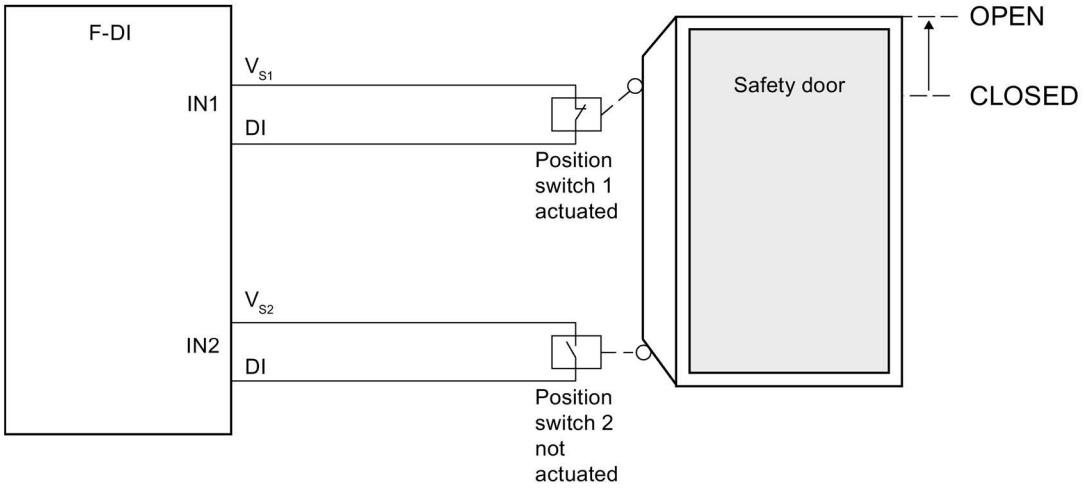
When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

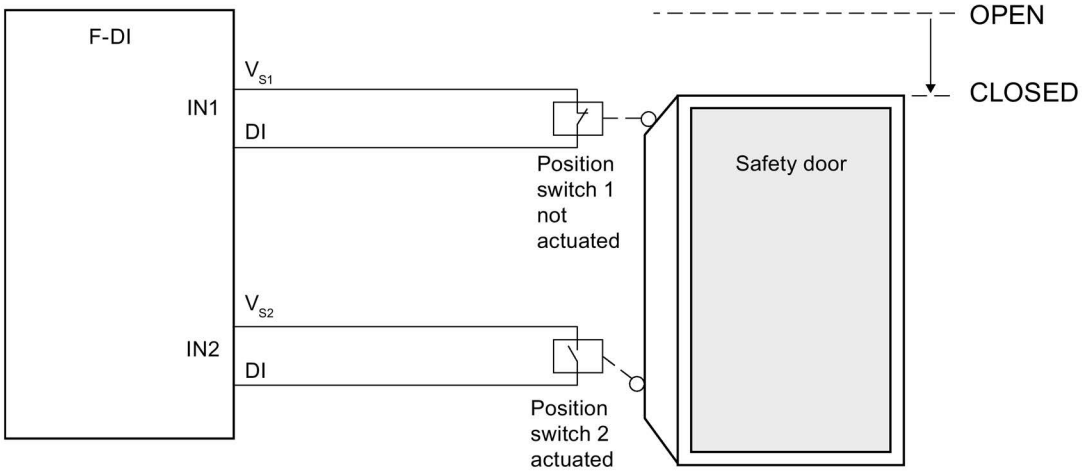
## Interconnection example

You must interconnect the NC contact of position switch 1 of the safety door at input IN1 and the NO contact of position switch 2 at input IN2. Position switch 1 must be mounted in such a way that it is positively operated when the safety door is open. Position switch 2 must be mounted in such a way that it is operated when the safety door is closed.

Safety door open:




Safety door closed:



## Startup characteristics

After an F-system startup, enable signal Q is reset to 0. The acknowledgment for the enable takes place according to the parameter assignment at inputs OPEN_NEC and ACK_NEC:

- When OPEN_NEC = 0, an automatic acknowledgment occurs **independently** of ACK_NEC, as soon as the two inputs IN1 and IN2 take signal state 1 for the first time following reintegration of the associated F-I/O (safety door is closed).
- When OPEN_NEC = 1 **or** if at least one of the IN1 and IN2 inputs still has a signal state of 0 after reintegration of the associated F-I/O, an automatic acknowledgment occurs **according** to ACK_NEC or you have to use a rising edge at input ACK for the enable. Prior to acknowledgment, inputs IN1 and IN2 both have to take a signal state of 0 (safety door has been completely opened) followed by a signal state of 1 (safety door is closed).

**⚠ WARNING**

The OPEN_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S039)*

## Output DIAG

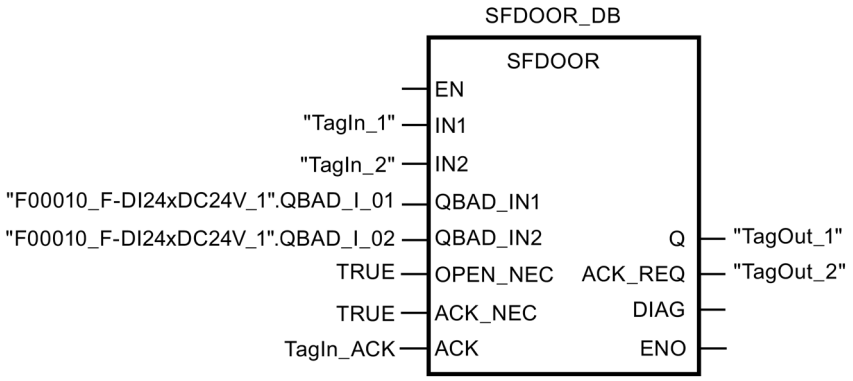
The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program.

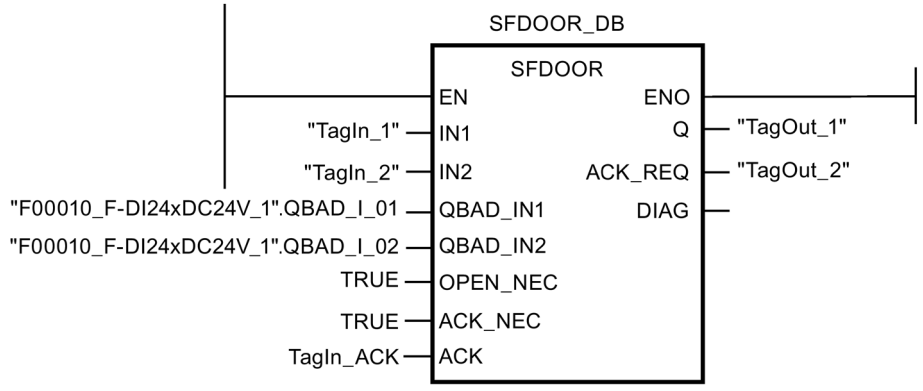
## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Reserved | — | — |
| Bit 1 | Signal state 0 is missing at both IN1 and IN2 inputs | Safety door was not completely opened when OPEN_NEC = 1 after F-system startup | Open safety door completely |
| | | Open safety door was not completely opened | Open safety door completely |
| | | Wiring fault | Check wiring of position switch |
| | | Position switch is defective | Check position switch |
| | | Position switch is incorrectly adjusted | Adjust position switch properly |
| Bit 2 | Signal state 1 is missing at both IN1 and IN2 inputs | Safety door was not closed | Close safety door |
| | | Wiring fault | Check wiring of position switch |
| | | Position switch is defective | Check position switch |
| | | Position switch is incorrectly adjusted | Adjust position switch properly |
| Bit 3 | QBAD_IN1 and/or QBAD_IN2 = 1 | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O or channel of IN1 and/or IN2 | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 4 | Reserved | — | — |
| Bit 5 | If enable is missing: input ACK has a permanent signal state of 1 | Acknowledgment button defective | Check acknowledgment button |
| | | Wiring fault | Check wiring of acknowledgment button |
| Bit 6 | Acknowledgment required (= state of ACK_REQ) | — | — |
| Bit 7 | State of output Q | — | — |

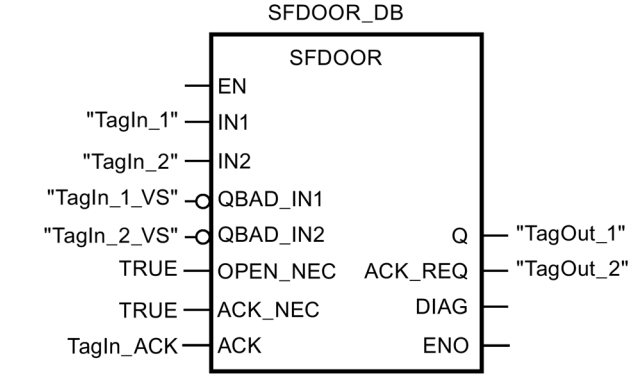
## Example

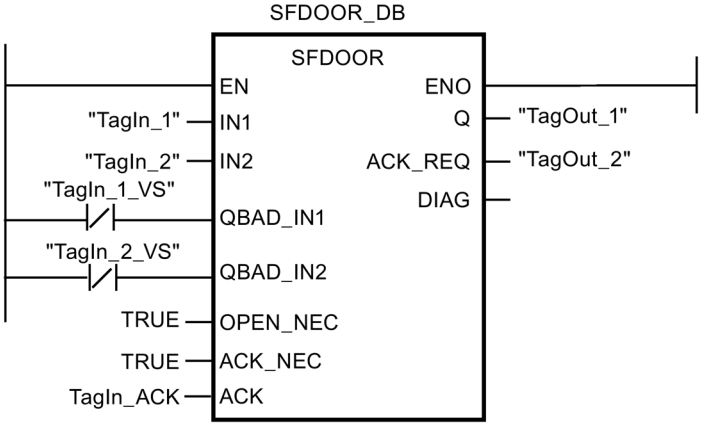
The following example shows how the instruction for S7-300/400 F-CPUs works:



The following example shows how the instruction for S7-1200/1500 F-CPUs works:



### 13.2.3.9 ACK_GL: Global acknowledgment of all F-I/O in an F-runtime group (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction creates an acknowledgment for the simultaneous reintegration of all F-I/O or channels of the F-I/O of an F-runtime group after communication errors, F-I/O errors, or channel faults.

A user acknowledgment (Page 151) with a positive edge at input ACK_GLOB is required for reintegration. The acknowledgment occurs analogously to the user acknowledgment via the ACK_REI tag of the F-I/O DB (Page 132), but it acts simultaneously on all F-I/O of the F-runtime group in which the instruction is called.

If you use the instruction ACK_GL, you do not have to provide a user acknowledgment for each F-I/O of the F-runtime group via the ACK_REI tag of the F-I/O DB.

Every call of the "Global acknowledgment of all F-I/O of a runtime group" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., ACK_GL_DB_1) or a multi-instance (e.g., ACK_GL_Instance_1) for the "Global acknowledgment of all F-I/O of a runtime group" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

An acknowledgment via the ACK_GL instruction is only possible if the tag ACK_REI of the F-I/O DB = 0. Accordingly, an acknowledgment via the tag ACK_REI of the F-I/O DB is only possible if the input ACK_GLOB of the instruction = 0.

The instruction is only allowed to be called once per F-runtime group.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ACK_GLOB | Input | BOOL | 1=acknowledgment for reintegration |

## Instruction versions

A number of versions are available for this instruction:

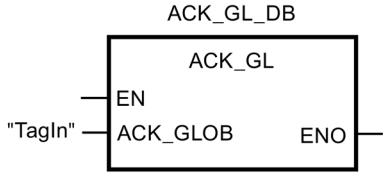
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

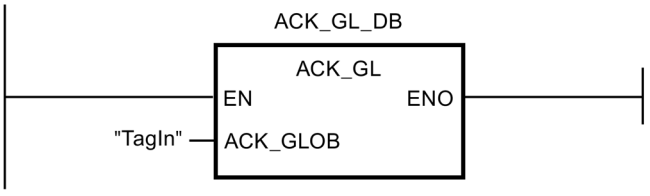
When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:



## 13.2.4 Timer operations

### 13.2.4.1 TP: Generate pulse (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Generate pulse" instruction to set output Q for an assigned period. The instruction is started if the result of logic operation (RLO) changes from "0" to "1" (positive edge) at input IN. The assigned period PT starts running when the instruction starts. Output Q is set for period PT, regardless of the subsequent sequence of the input signal. Also the detection of a new positive signal edge does not influence the signal state at output Q as long as period PT runs.

The current time value can be queried at output ET. The time value begins at T#0s and ends when the value of period PT is reached. If period PT is reached and the signal state at input IN is "0", output ET is reset.

Every call of the "Generate pulse" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Timer_DB_1) or a multi-instance (e.g., F_IEC_Timer_Instance_1) for the "Generate pulse" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction").
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

The operating system resets the instances of the "Generate pulse" instruction on a startup of the F-system.

**Note**

The functionality of this instruction differs from the corresponding standard TP instruction in the following points:

- When a call is made with PT = 0 ms, the TP instance is not reset completely (initialized). The instruction behaves in accordance with the timing diagrams: Only outputs Q and ET are reset. To restart the pulse, a new rising signal edge at input IN is required once PT is greater than 0 again.
- A call with PT < 0 ms resets outputs Q and ET. To restart the pulse, a new rising signal edge at input IN is required once PT is greater than 0 again.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | BOOL | Start input |
| PT | Input | TIME | Duration of pulse; must be positive. |
| Q | Output | BOOL | Pulse output |
| ET | Output | TIME | Current time value |

## Instruction versions

A number of versions are available for this instruction:

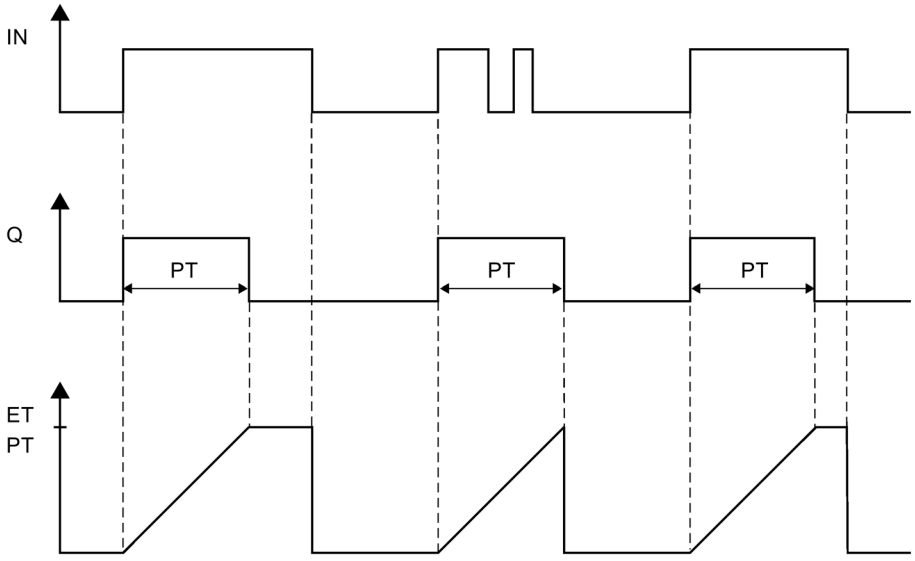
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

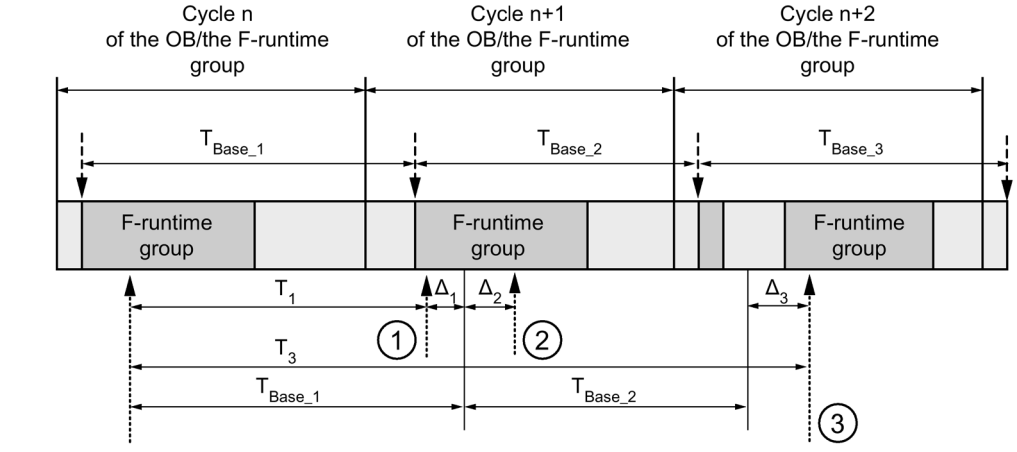
For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Pulse diagram

The following figure shows the pulse diagram of the instruction:



## Timing imprecision resulting from the update time of the time base used in the instruction:




----► = Time base update

·····► = Call time of an instruction with time processing

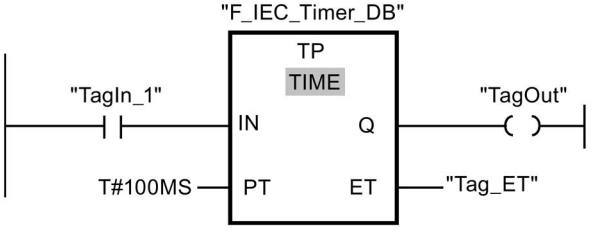
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:




If the signal state of operand "TagIn_1" changes from "0" to "1", the "Generate pulse" instruction is started and the period assigned at input PT (100 ms) runs, regardless of the further course of operand "TagIn_1".

Operand "TagOut" at output Q has signal state "1" as long as the period is running. Operand "Tag_ET" contains the current time value.

### 13.2.4.2 TON: Generate on-delay (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Generate on-delay" instruction to delay the setting of output Q by the assigned period PT. The "Generate on-delay" instruction is started when the result of logic operation (RLO) at input IN changes from "0" to "1" (positive signal edge). The assigned period PT starts running when the instruction starts. When period PT has expired, output Q is set to signal state "1". Output Q remains set as long as the start input is set to "1". When the signal state at the start input changes from "1" to "0", output Q is reset. The time function is restarted when a new positive signal edge is detected at the start input.

The current time value can be queried at output ET. The time value begins at T#0s and ends when the value of period PT is reached. Output ET is reset, as soon as the signal state at input IN changes to "0".

Every call of the "Generate on-delay" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Timer_DB_1) or a multi-instance (e.g., F_IEC_Timer_Instance_1) for the "Generate on-delay" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

The operating system resets the instances of the "Generate on-delay" instruction on a startup of the F-system.

**Note**

The functionality of this instruction differs from the corresponding standard TON instruction in the following points:

- When a call is made with PT = 0 ms, the instance of the TON is not reset completely (initialized). The instruction behaves in accordance with the timing diagrams: Only output ET is reset. To restart the on-delay, a new rising signal edge at input IN is required once PT is greater than 0 again.
- A call with PT < 0 ms resets outputs Q and ET. To restart the on-delay, a new rising signal edge at input IN is required once PT is greater than 0 again.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | BOOL | Start input |
| PT | Input | TIME | Duration of on-delay; must be positive. |
| Q | Output | BOOL | Output that is set after expiration of time PT. |
| ET | Output | TIME | Current time value |

## Instruction versions

A number of versions are available for this instruction:

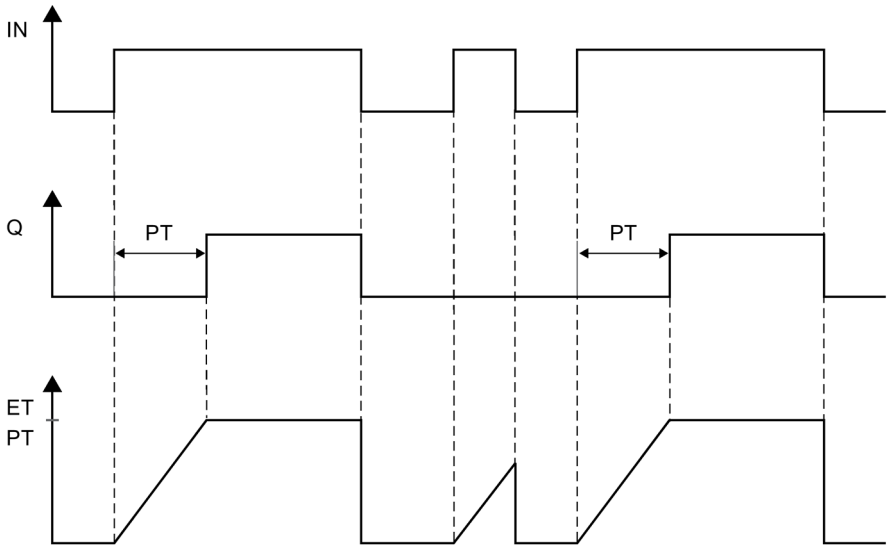
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

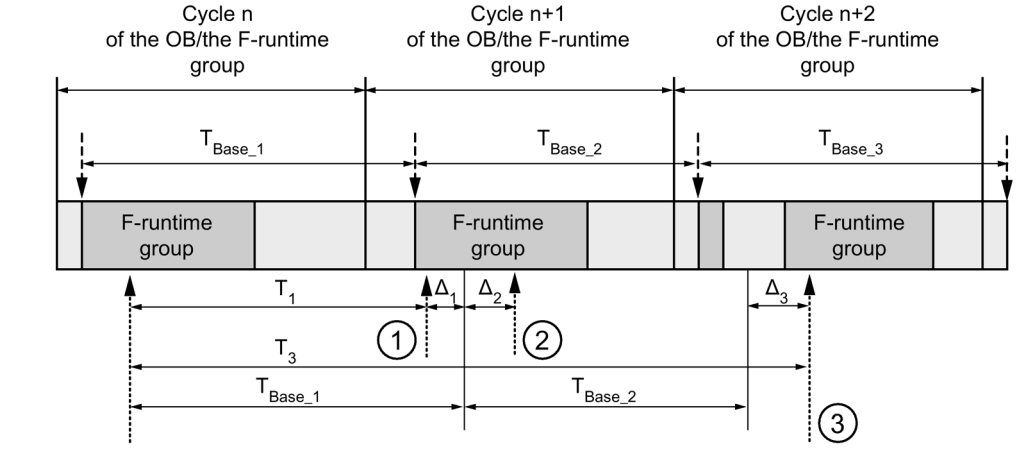
For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Pulse diagram

The following figure shows the pulse diagram of the instruction:



## Timing imprecision resulting from the update time of the time base used in the instruction:




----► = Time base update

·····► = Call time of an instruction with time processing

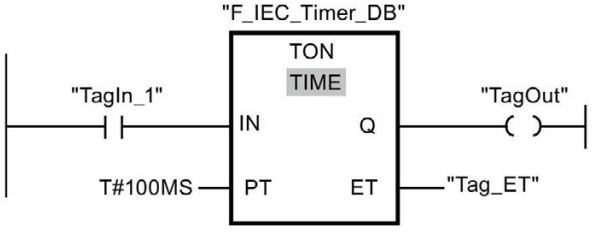
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1}$ + $T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:




When the signal state of operand "TagIn_1" changes from "0" to "1", the

"Generate on-delay" instruction is started and the period assigned at input PT (1 s) runs.

Operand "TagOut" at output Q feeds signal state "1" when the period has elapsed and remains set as long as operand "TagIn_1" still feeds signal state "1". Operand "Tag_ET" contains the current time value.

### 13.2.4.3 TOF: Generate off-delay (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Generate off-delay" instruction to delay resetting output Q by the assigned period PT. Output Q is set if the result of logic operation (RLO) changes from "0" to "1" (positive edge) at input IN. The assigned period PT starts when the signal state at input IN changes back to "0". Output Q remains set as long as period PT runs. After period PT expires, output Q is reset. If the signal state at input IN changes to "1" before period PT has expired, then the time is reset. The signal state at output Q remains at "1".

The current time value can be queried at output ET. The time value begins at T#0s and ends when the value of period PT is reached.

Every call of the "Generate off-delay" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is in inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Timer_DB_1) or a multi-instance (e.g., F_IEC_Timer_Instance_1) for the "Generate off-delay" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

The operating system resets the instances of the "Generate off-delay" instruction on a startup of the F-system.

**Note**

The functionality of this instruction differs from the corresponding standard TOF instruction in the following points:

- When a call is made with PT = 0 ms, the instance of the TOF is not reset completely (initialized). The instruction behaves in accordance with the timing diagrams: Only outputs Q and ET are reset. To restart the off-delay, another falling signal edge at input IN is required once PT is greater than 0 again.
- A call with PT < 0 ms resets outputs Q and ET. To restart the off-delay, another falling signal edge at input IN is required once PT is greater than 0 again.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | BOOL | Start input |
| PT | Input | TIME | Duration of off delay; must be positive. |
| Q | Output | BOOL | Output that is reset after expiration of time PT. |
| ET | Output | TIME | Current time value |

## Instruction versions

A number of versions are available for this instruction:

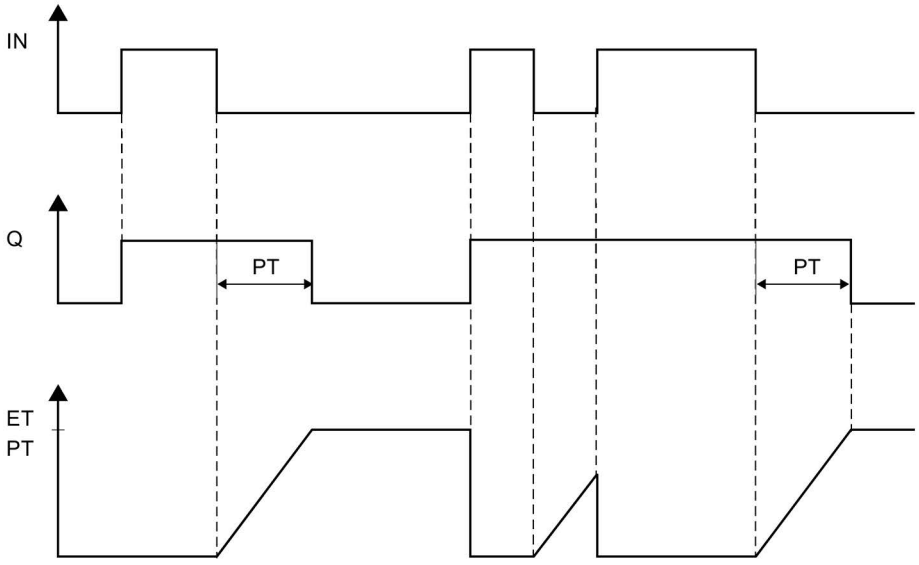
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

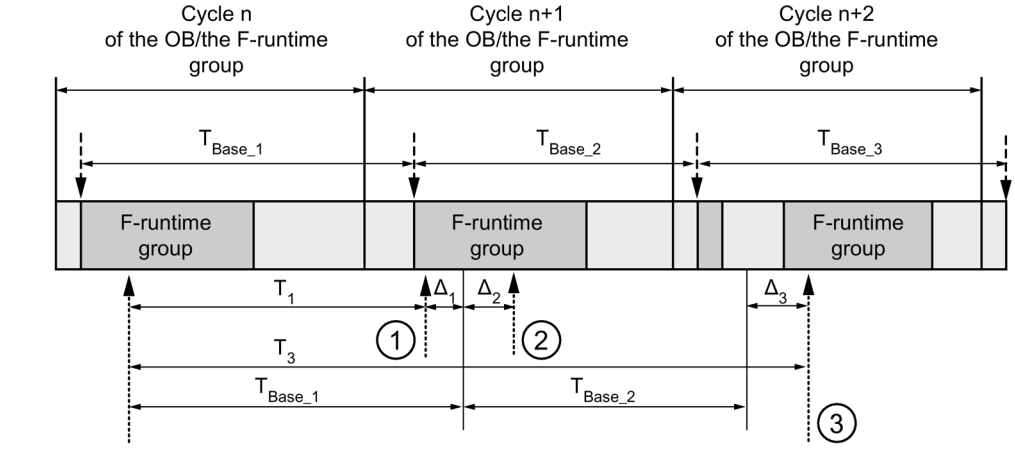
For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Pulse diagram

The following figure shows the pulse diagram of the instruction:



### Timing imprecision resulting from the update time of the time base used in the instruction:




----► = Time base update

·····► = Call time of an instruction with time processing

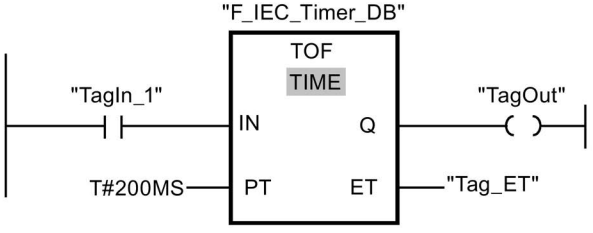
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1}$ + $T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:




If the signal state of operand "TagIn_1" changes from "0" to "1", the signal state of operand"TagOut" at output Q is set to "1".

If the signal state of operand "TagIn_1" changes back to "0", the period assigned at input PT (200 ms) runs.

The "TagOut" operand at output Q is set back to "0" when the period expires. Operand "Tag_ET" contains the current time value.

## 13.2.5 Counter operations

### 13.2.5.1 CTU: Count up (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Count up" instruction to increment the value at output CV. When the signal state at input CU changes from "0" to "1" (positive signal edge), the instruction is executed and the current count value at output CV is increased by one. The count value is increased on each detection of a positive signal edge until it reaches the high limit of the data type specified at output CV. When the high limit is reached, the signal state at input CU no longer affects the instruction.

The counter status can be queried at output Q. The signal state at output Q is determined by parameter PV. When the current count value is greater than or equal to the value of parameter PV, output Q is set to signal state "1". In all other cases, the signal state at output Q is "0".

The value at output CV is reset to zero when the signal state at input R changes to "1". As long as signal state "1" exists at input R, the signal state at input CU has no effect on the instruction.

Every call of the "Count up" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Counter_DB_1) or a multi-instance (e.g., F_IEC_Counter_Instance_1) for the "Count up" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

The operating system resets the instances of the "Count up" instruction on a startup of the F-system.

#### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CU | Input | BOOL | Counter input |
| R | Input | BOOL | Reset input |
| PV | Input | INT | Value for which output Q is set |
| Q | Output | BOOL | Counter status |
| CV | Output | INT | Current count value |

## Instruction versions

A number of versions are available for this instruction:

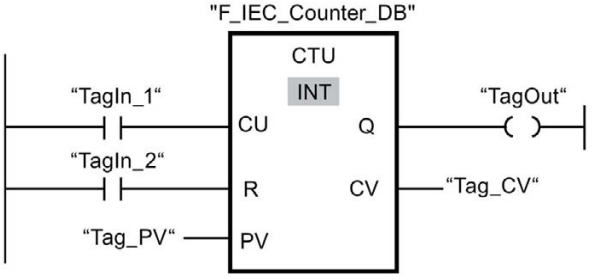
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




When the signal state of the "TagIn_1" operand changes from "0" to "1", the "Count up" instruction is executed and the current count value of the "Tag_CV" operand is increased by one. The count value is increased on every additional positive signal edge until the high limit of the specified data type (32767) is reached.

The value at parameter PV is taken as the limit for the determination of output "TagOut". Output "TagOut" delivers the signal state "1" as long as the current count value is greater than or equal to the value of operand "Tag_PV". In all other cases, output TagOut has signal state "0".

### 13.2.5.2 CTD: Count down (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Count down" instruction to decrement the value at output CV. When the signal state at input CD changes from "0" to "1" (positive signal edge), the instruction is executed and the current count value at output CV is decreased by one. The count value is decreased on each detection of a positive signal edge until it reaches the low limit of the specified data type. When the low limit is reached, the signal state at input CD no longer affects the instruction.

The counter status can be queried at output Q. When the current count value is less than or equal to zero, output Q is set to signal state "1". In all other cases, the signal state at output Q is "0".

The value at output CV is set to the value of parameter "PV" when the signal state at input LD changes to "1". As long as signal state "1" exists at input LD, the signal state at input CD has no effect on the instruction.

Every call of the "Count down" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Counter_DB_1) or a multi-instance (e.g., F_IEC_Counter_Instance_1) for the "Count down" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

The operating system resets the instances of the "Count down" instruction on a startup of the F-system.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CD | Input | BOOL | Counter input |
| LD | Input | BOOL | Load input |
| PV | Input | INT | Value at the output CV when LD = 1 is set |
| Q | Output | BOOL | Counter status |
| CV | Output | INT | Current count value |

## Instruction versions

A number of versions are available for this instruction:

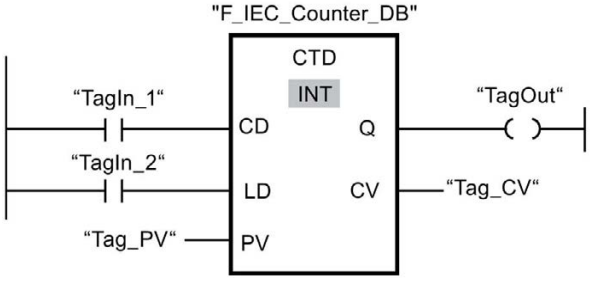
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




If the signal state of the "TagIn_1" operand changes from "0" to "1", the "Count down" instruction is executed and the current count value at output "Tag_CV" is decreased by one. The count value is decreased on each additional positive signal edge until the low limit of the specified data type (-32768) is reached.

Output "TagOut" delivers the signal state "1" as long as the current count value is less than or equal to zero. In all other cases, output TagOut has signal state "0".

### 13.2.5.3 CTUD: Count up and down (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Count up and down" instruction to increment and decrement the count value at output CV. If the signal state at input CU changes from "0" to "1" (positive signal edge), the current count value at output CV is increased by one. If the signal state at input CD changes from "0" to "1" (positive signal edge), the count value at output CV is decreased by one. If a positive signal edge is present at inputs CU and CD in one program cycle, the current count value at output CV remains unchanged.

The count value can be increased until it reaches the high limit of the data type specified at output CV. When the high limit is reached, the count value is no longer incremented on a positive signal edge. When the low limit of the specified data type is reached, the count value is no longer decreased.

When the signal state at input LD changes to "1", the count value at output CV is set to the value of parameter PV. As long as signal state "1" exists at input LD, the signal state at inputs CU and CD has no effect on the instruction.

The count value is set to zero, when the signal state at input R changes to "1". As long as signal state "1" exists at input R, the signal state at inputs CU, CD, and LD has no effect on the "Count up and down" instruction.

The status of the up counter can be queried at output QU. When the current count value is greater than or equal to the value of parameter PV, output QU delivers signal state "1". In all other cases, the signal state at output QU is "0".

The status of the down counter can be queried at output QD. When the current count value is lesser than or equal to zero, output QD delivers signal state "1". In all other cases, the signal state at output QD is "0".

Every call of the "Count up and down" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Counter_DB_1) or a multi-instance (e.g., F_IEC_Counter_Instance_1) for the "Count up and down" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

The operating system resets the instances of the "Count up and down" instruction on a cold restart of the F-system.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CU | Input | BOOL | Count up input |
| CD | Input | BOOL | Count down input |
| R | Input | BOOL | Reset input |
| LD | Input | BOOL | Load input |
| PV | Input | INT | Value set at the output QU/ at which the output CV is set at LD = 1. |
| QU | Output | BOOL | Status of up counter |
| QD | Output | BOOL | Status of down counter |
| CV | Output | INT | Current count value |

## Instruction versions

A number of versions are available for this instruction:

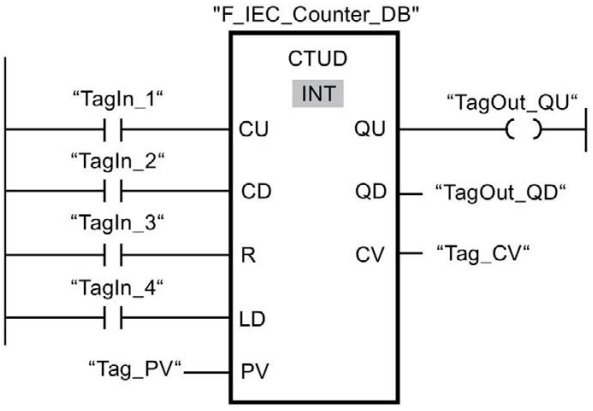
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




When the signal state at input "TagIn_1" or at input "TagIn_2" changes from "0" to "1" (positive signal edge), the "Count up and down" instruction is executed. When a positive signal edge is present at input "TagIn_1", the current count value of the "Tag_CV" operand is increased by one. When a positive signal edge is present at input "TagIn_2", the current count value at output "Tag_CV" is decreased by one. The count value is increased on each positive signal edge at input CU until it reaches the high limit of 32767. The count value is decreased on each positive signal edge at input CD until it reaches the low limit of -32768.

Output "TagOut_QU" delivers the signal state "1" as long as the current count value is greater than or equal to the value at input "Tag_PV". In all other cases, output TagOut_QU has signal state "0".

Output "TagOut_QD" delivers the signal state "1" as long as the current count value is less than or equal to zero. In all other cases, output TagOut_QD has signal state "0".

## 13.2.6 Comparator operations

### 13.2.6.1 CMP ==: Equal (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Equal" instruction to determine if the first comparison value (<Operand1>) is equal to the second comparison value (<Operand2>).

If the condition of the comparison is satisfied, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not satisfied, the instruction returns RLO "0". The RLO of the instruction is linked to the RLO of the entire current path as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Enter the first comparison value (<Operand1>) in the operand placeholder above the instruction. Enter the second comparison value (<Operand2>) in the operand placeholder below the instruction.

#### Parameters

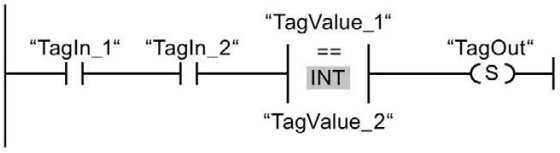
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | INT, DINT, TIME, WORD, DWORD | First value to compare |
| <Operand2> | Input | INT, DINT, TIME, WORD, DWORD | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

#### Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled ("TagValue_1" = "TagValue_2").

### 13.2.6.2 CMP <>: Unequal (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Not equal" instruction to determine if the first comparison value (<Operand1>) is not equal to the second comparison value (<Operand2>).

If the condition of the comparison is satisfied, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not satisfied, the instruction returns RLO "0". The RLO is linked to the RLO of the instruction of the entire current path as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Enter the first comparison value (<Operand1>) in the operand placeholder above the instruction. Enter the second comparison value (<Operand2>) in the operand placeholder below the instruction.

## Parameters

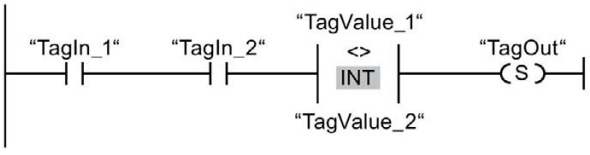
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | INT, DINT, TIME, WORD, DWORD | First value to compare |
| <Operand2> | Input | INT, DINT, TIME, WORD, DWORD | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Inputs "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled ("TagValue_1" <> "TagValue_2").

### 13.2.6.3 CMP >=: Greater or equal (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Greater or equal" instruction to determine if the first comparison value (<Operand1>) is greater than or equal to the second comparison value (<Operand2>). Both comparison values must be of the same data type.

If the condition of the comparison is satisfied, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not satisfied, the instruction returns RLO "0".

The RLO of the instruction is linked to the RLO of the entire current path as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Enter the first comparison value (<Operand1>) in the operand placeholder above the instruction. Enter the second comparison value (<Operand2>) in the operand placeholder below the instruction.

## Parameters

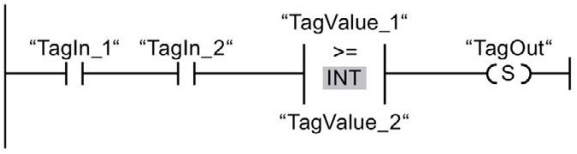
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | INT, DINT, TIME | First value to compare |
| <Operand2> | Input | INT, DINT, TIME | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled ("TagValue_1" >= "TagValue_2").

### 13.2.6.4 CMP <=: Less or equal (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Less or equal" instruction to determine if the first comparison value (<Operand1>) is less than or equal to the second comparison value (<Operand2>). Both comparison values must be of the same data type.

If the condition of the comparison is satisfied, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not satisfied, the instruction returns RLO "0".

The RLO of the instruction is linked to the RLO of the entire current path as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Enter the first comparison value (<Operand1>) in the operand placeholder above the instruction. Enter the second comparison value (<Operand2>) in the operand placeholder below the instruction.

## Parameters

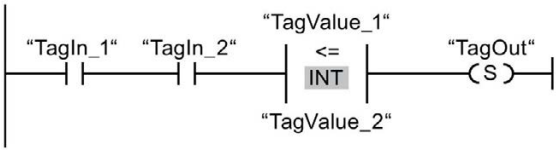
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | INT, DINT, TIME | First value to compare |
| <Operand2> | Input | INT, DINT, TIME | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled ("TagValue_1" <= "TagValue_2").

### 13.2.6.5 CMP >: Greater than (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Greater than" instruction to determine if the first comparison value (<Operand1>) is greater than the second comparison value (<Operand2>). Both comparison values must be of the same data type.

If the condition of the comparison is satisfied, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not satisfied, the instruction returns RLO "0".

The RLO of the instruction is linked to the RLO of the entire current path as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Enter the first comparison value (<Operand1>) in the operand placeholder above the instruction. Enter the second comparison value (<Operand2>) in the operand placeholder below the instruction.

## Parameters

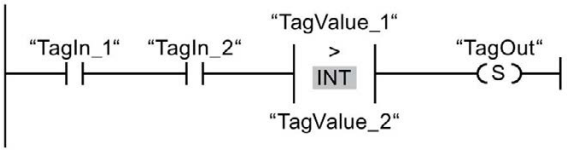
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | INT, DINT, TIME | First value to compare |
| <Operand2> | Input | INT, DINT, TIME | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled ("TagValue1" > "TagValue2").

### 13.2.6.6 CMP <: Less than (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Less than" instruction to determine if the first comparison value (<Operand1>) is less than the second comparison value (<Operand2>). Both comparison values must be of the same data type.

If the condition of the comparison is satisfied, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not satisfied, the instruction returns RLO "0".

The RLO of the instruction is linked to the RLO of the entire current path as follows:

- By AND, when the comparison instruction is connected in series.
- By OR, when the comparison instruction is connected in parallel.

Enter the first comparison value (<Operand1>) in the operand placeholder above the instruction. Enter the second comparison value (<Operand2>) in the operand placeholder below the instruction.

## Parameters

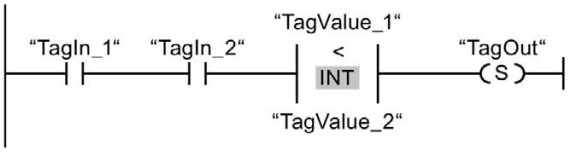
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | INT, DINT, TIME | First value to compare |
| <Operand2> | Input | INT, DINT, TIME | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The condition of the comparison instruction is fulfilled ("TagValue_1" < "TagValue_2").

## 13.2.7 Math functions

### 13.2.7.1 ADD: Add (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Add" instruction to add the value at input IN1 and the value at input IN2 and query the sum at the OUT output (OUT = IN1 + IN2).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic alarms.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

#### Parameters

The following table shows the parameters of the instruction:

Table 13- 1

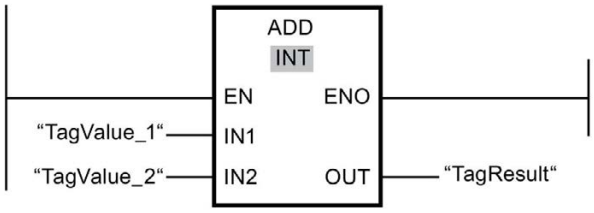
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT | First addend |
| IN2 | Input | INT, DINT | Second addend |
| OUT | Output | INT, DINT | Total |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

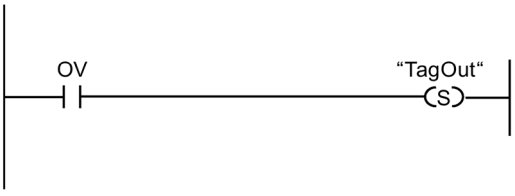
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Add" instruction is always executed (regardless of the signal state at enable input EN).

The value of the "TagValue_1" operand is added to value of the TagValue_2 operand. The result of the addition is stored in the "TagResult" operand.

(S7-300, S7-400) If an overflow occurs during execution of the "Add" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

---| |--- OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 478)

---| / |--- OV: Get negated status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 479)

### 13.2.7.2 SUB: Subtract (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Subtract" instruction to subtract the value at input IN2 from the value at input IN1 and query the difference at the OUT output (OUT = IN1 – IN2).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic alarms.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

## Parameters

The following table shows the parameters of the instruction:

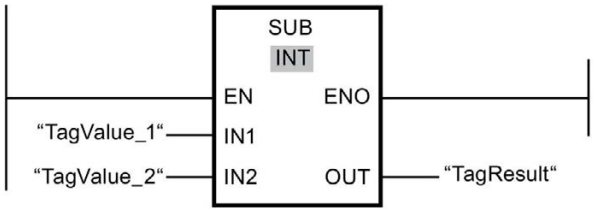
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT | Minuend |
| IN2 | Input | INT, DINT | Subtrahend |
| OUT | Output | INT, DINT | Difference |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

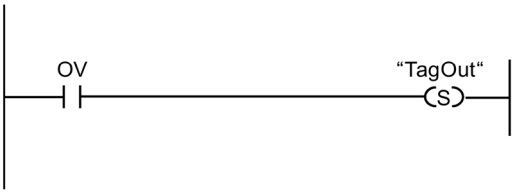
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Subtract" instruction is always executed (regardless of the signal state at enable input EN).

The value of operand "TagValue_2" is subtracted from the value of operand "TagValue_1". The result of the addition is stored in operand "TagResult".

(S7-300, S7-400) If an overflow occurs during execution of the "Subtract" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

---| |--- OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 478)

---| / |--- OV: Get negated status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 479)

### 13.2.7.3 MUL: Multiply (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Multiply" instruction to multiply the value at input IN1 by the value at input IN2 and query the product at output OUT (OUT = IN1 × IN2).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic alarms.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

## Parameters

The following table shows the parameters of the instruction:

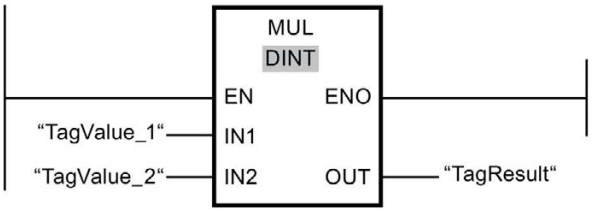
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT | Multiplier |
| IN2 | Input | INT, DINT | Multiplicand |
| OUT | Output | INT, DINT | Product |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

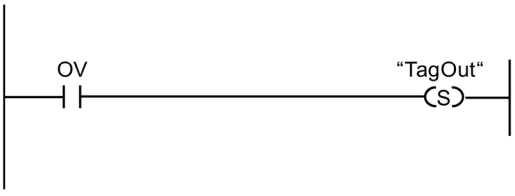
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Multiply" instruction is always executed (regardless of the signal state at enable input EN).

The value of the "TagValue_1" operand is multiplied by the value of the "TagValue_2" operand. The result of the multiplication is stored in the "TagResult" operand.

(S7-300, S7-400) If an overflow occurs during execution of the "Multiply" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

---| |--- OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 478)

---| / |--- OV: Get negated status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 479)

### 13.2.7.4 DIV: Divide (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Divide" instruction to divide the value at input IN1 by the value at input IN2 and query the quotient at the OUT output (OUT = IN1 / IN2).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts in this case like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

**Note**

S7-300/400:

If the divisor (input IN2) of a DIV instruction = 0, the quotient of the division (result of division at output OUT) = 0. The result reacts like the corresponding instruction in a standard block. The F-CPU does *not* go to STOP mode.

S7-1200/1500:

If the divisor (input IN2) of a DIV instruction = 0, the F-CPU goes to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages. We recommend that you rule out a divisor (input IN2) = 0 when creating the program.

## Parameters

The following table shows the parameters of the instruction:

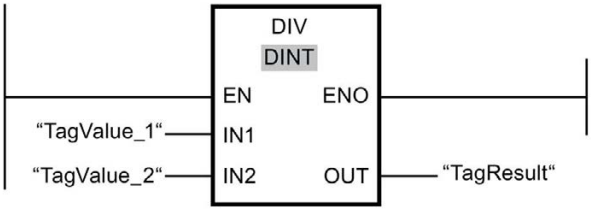
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT | Dividend |
| IN2 | Input | INT, DINT | Divisor |
| OUT | Output | INT, DINT | Quotient |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

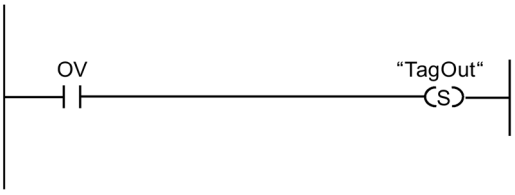
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Divide" instruction is always executed (regardless of the signal state at enable input EN).

The value of operand "TagValue_1" is divided by the value of operand "TagValue_2". The result of the division is stored in operand "TagResult".

(S7-300, S7-400) If an overflow occurs during execution of the "Divide" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

---| |--- OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 478)

---| / |--- OV: Get negated status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 479)

### 13.2.7.5 NEG: Create twos complement (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Create twos complement" instruction to change the sign of the value at input IN input and query the result at output OUT. If there is a positive value at input IN, for example, the negative equivalent of this value is sent to output OUT.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic alarms.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

## Parameters

The following table shows the parameters of the instruction:

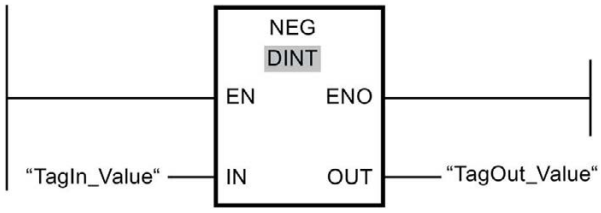
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT, DINT | Input value |
| OUT | Output | INT, DINT | Twos complement of the input value |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

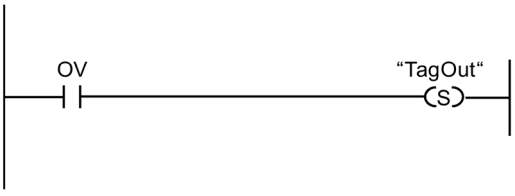
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Create twos complement" instruction is always executed (regardless of the signal state at enable input EN).

The sign of the "TagIn_Value" operand is changed and the result is stored in the "TagOut_Value" operand.

(S7-300, S7-400) If an overflow occurs during execution of the "Create twos complement" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

---| |--- OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 478)

---| / |--- OV: Get negated status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 479)

## 13.2.8 Move operations

### 13.2.8.1 MOVE: Move value (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Move value" instruction to transfer the content of the operand at input IN to the operand at output OUT1.

Only identical operand widths can be specified for input IN and output OUT1.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

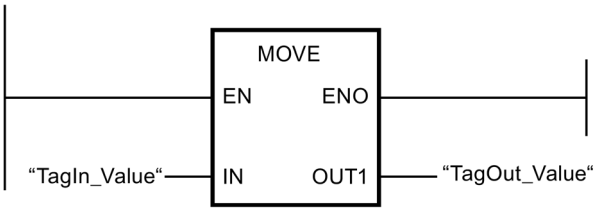
#### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT, DINT, WORD, DWORD, TIME | Source value |
| OUT1 | Output | INT, DINT, WORD, DWORD, TIME | Destination address |

#### Example

The following example shows how the instruction works:




The instruction is always executed regardless of the signal state at enable input "EN". The instruction copies the content of operand "TagIn_Value" to operand "TagOut_Value".

### 13.2.8.2 WR_FDB: Write value indirectly to an F-DB (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

This instruction writes the value specified in input IN to the tag addressed by INI_ADDR and OFFSET in an F-DB.

The address of the tags addressed using INI_ADDR and OFFSET must be within the address range defined by addresses INI_ADDR and END_ADDR.

If the F-CPU has gone to STOP mode with diagnostic event ID 75E2, check to determine if this condition is satisfied.

The start address of the area in an F-DB to which the value at input IN is to be written is transferred using input INI_ADDR. The associated offset in this area is transferred using input OFFSET.

The addresses transferred in input INI_ADDR or END_ADDR must point to a tag of the selected data type in an F-DB. Only tags of the selected data type are permitted between the INI_ADDR and END_ADDR addresses. The INI_ADDR address must be smaller than the END_ADDR address.

As shown in the following example, the INI_ADDR and END_ADDR addresses must be transferred fully-qualified as "DBx".DBWy or in the corresponding symbolic representation. Transfers in other forms are not permitted.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT, DINT | Value to be written to the F-DB |
| INI_ADDR | Input | POINTER | Start address of the area in an F-DB |
| END_ADDR | Input | POINTER | End address of the area in an F-DB |
| OFFSET | Input | INT | Offset |

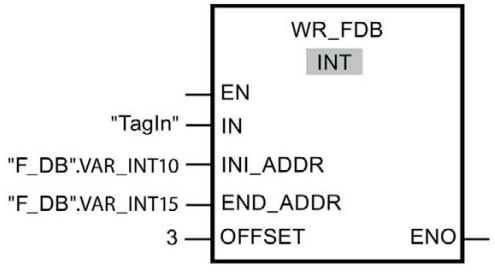
You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Examples of parameter assignment of INI_ADDR, END_ADDR, and OFFS

| Name | Data type | Initial value | Comment |
|---|---|---|---|
| Static | | | |
| VAR_BOOL10 | Bool | false | |
| VAR_BOOL11 | Bool | false | |
| VAR_BOOL12 | Bool | false | |
| VAR_BOOL13 | Bool | false | |
| VAR_TIME10 | Time | T#0MS | |
| VAR_TIME11 | Time | T#0MS | |
| VAR_INT10 | Int | 0 | <- INI_ADDR = "F-DB".VAR_INT10 Example 1 |
| VAR_INT11 | Int | 0 | |
| VAR_INT12 | Int | 0 | |
| VAR_INT13 | Int | 0 | <- OFFSET = 3 |
| VAR_INT14 | Int | 0 | |
| VAR_INT15 | Int | 0 | <- END_ADDR = "F-DB".VAR_INT15 |
| VAR_BOOL20 | Bool | false | |
| VAR_BOOL21 | Bool | false | |
| VAR_BOOL22 | Bool | false | |
| VAR_BOOL23 | Bool | false | |
| VAR_INT20 | Int | 0 | <- INI_ADDR = "F-DB".VAR_INT20 Example 2 |
| VAR_INT21 | Int | 0 | |
| VAR_INT22 | Int | 0 | |
| VAR_INT23 | Int | 0 | <- END_ADDR = "F-DB".VAR_INT23 |
| VAR_INT30 | Int | 0 | <- INI_ADDR = "F-DB".VAR_INT30 Example 3 |
| VAR_INT31 | Int | 0 | <- OFFSET = 1 |
| VAR_INT32 | Int | 0 | |
| VAR_INT33 | Int | 0 | |
| VAR_INT34 | Int | 0 | <- END_ADDR = "F-DB".VAR_INT34 |
| VAR_TIME20 | TIME | T#0MS | |
| VAR_DINT10 | DInt | 0 | <- INI_ADDR = "F-DB".VAR_DINT10 Example 4 |
| VAR_DINT11 | DInt | 0 | |
| VAR_DINT12 | DInt | 0 | <- OFFSET = 2 |
| VAR_DINT13 | DInt | 0 | <- END_ADDR = "F-DB".VAR_DINT13 |

## Example

The following example shows how the instruction works:



### 13.2.8.3 RD_FDB: Read value indirectly from an F-DB (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

This instruction reads the tag addressed via INI_ADDR and OFFSET in an F-DB and provides it at output OUT.

The address of the tags addressed using INI_ADDR and OFFSET must be within the address range defined by addresses INI_ADDR and END_ADDR.

If the F-CPU has gone to STOP mode with diagnostic event ID 75E2, check to determine if this condition is satisfied.

The start address of the area in an F-DB from which the tag is to be read is transferred using input INI_ADDR. The associated offset in this area is transferred using input OFFSET.

The addresses transferred in input INI_ADDR or END_ADDR must point to a tag of the selected data type in an F-DB. Only tags of the selected data type are permitted between the INI_ADDR and END_ADDR addresses. The INI_ADDR address must be smaller than the END_ADDR address.

The INI_ADDR and END_ADDR addresses must be transferred fully-qualified as "DBx".DBWy or in the corresponding symbolic representation. Transfers in other forms are not permitted. Examples of parameter assignment of INI_ADDR, END_ADDR, and OFFSET are contained in WR_FDB: Write value indirectly to an F-DB (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 445).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameters

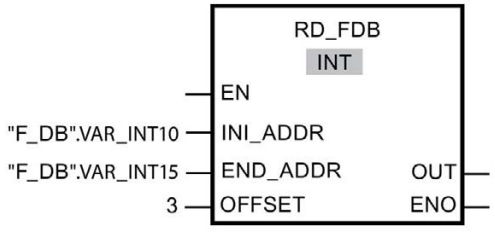
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| INI_ADDR | Input | POINTER | Start address of the area in an F-DB |
| END_ADDR | Input | POINTER | End address of the area in an F-DB |
| OFFSET | Input | INT | Offset |
| OUT | Output | INT, DINT | Value to be read from the F-DB |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

### Example

The following example shows how the instruction works:




## 13.2.9 Conversion operations

### 13.2.9.1 CONVERT: Convert value (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

### Description

The "Convert value" instruction reads the content of parameter IN and converts it according to the data types selected in the instruction box. The converted value is output at output OUT.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

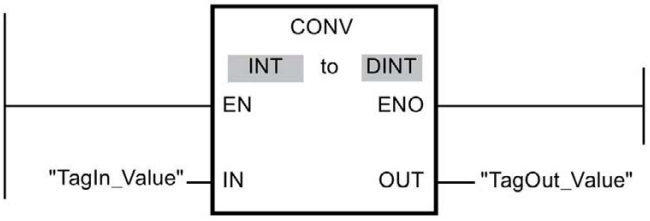
### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT | Value to be converted. |
| OUT | Output | DINT | Result of the conversion |

## Example

The following example shows how the instruction works:




The instruction is always executed regardless of the signal state at enable input EN. The content of the operand "TagIn_Value" is read and converted to an integer (32 bit). The result is stored in operand "TagOut_Value".

### 13.2.9.2 BO_W: Convert 16 data elements of data type BOOL to a data element of data type WORD (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction converts the 16 values of data type BOOL at inputs IN0 to IN15 to a value of data type WORD, which is made available at output OUT. The conversion takes place as follows: The i-th bit of the WORD value is set to 0 (or 1), if the value at input INi = 0 (or 1).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN0 | Input | BOOL | Bit 0 of WORD value |
| IN1 | Input | BOOL | Bit 1 of WORD value |
| ... | | | ... |
| IN15 | Input | BOOL | Bit 15 of WORD value |
| OUT | Output | WORD | WORD value consisting of IN0 to IN15 |

## Instruction versions

A number of versions are available for this instruction:

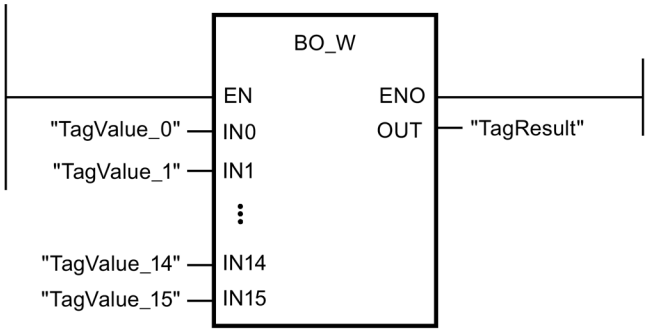
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | These versions have identical functions to version V1.0. |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




The following table shows how the instruction works using specific operand values:

| Parameter | Operand | Value |
|---|---|---|
| IN0 | TagValue_0 | FALSE |
| IN1 | TagValue_1 | FALSE |
| ... | | ... |
| IN13 | TagValue_13 | FALSE |
| IN14 | TagValue_14 | TRUE |
| IN15 | TagValue_15 | TRUE |
| OUT | TagResult | W#16#C000 |

The values of operands "TagValue_0" to " TagValue_15" are combined to form data type WORD and assigned to operand "TagResult".

### 13.2.9.3 W_BO: Convert 16 data elements of data type WORD to a data element of data type BOOL (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction converts the value of data type WORD at input IN to 16 values of data type BOOL, which are provided at outputs OUT0 to OUT15. The conversion takes place as follows: Output OUTi is set to 0 (or 1), if the i-th bit of the WORD value is 0 (or 1).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | WORD | WORD value |
| OUT0 | Output | BOOL | Bit 0 of WORD value |
| OUT1 | Output | BOOL | Bit 1 of WORD value |
| ... | | | ... |
| OUT15 | Output | BOOL | Bit 15 of WORD value |

## Instruction versions

A number of versions are available for this instruction:

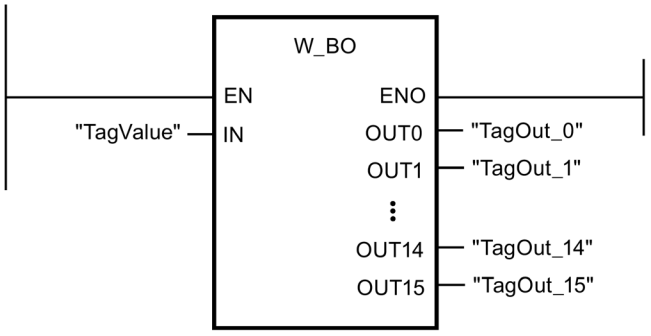
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | These versions have identical functions to version V1.0. |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




The following table shows how the instruction works using specific operand values:

| Parameter | Operand | Value |
|---|---|---|
| IN | TagValue | W#16#C000 |
| OUT0 | TagOUT_0 | FALSE |
| OUT1 | TagOUT_1 | FALSE |
| ... | | ... |
| OUT13 | TagOUT_13 | FALSE |
| OUT14 | TagOUT_14 | TRUE |
| OUT15 | TagOUT_15 | TRUE |

The value of operand "TagValue" of data type WORD is converted to the 16 values "TagOUT_0" to "TagOUT_15" of data type BOOL.

### 13.2.9.4 SCALE: Scale values (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500)

## Description

This instruction scales the value at input IN in physical units between the low limit value at input LO_LIM and the high limit value at input HI_LIM. It is assumed that the value at input IN is between 0 and 27648. The scaling result is provided at output OUT.

The instruction uses the following equation:

OUT = [ IN × (HI_LIM – LO_LIM) ] / 27648 + LO_LIM

As long as the value at input IN is greater than 27648, output OUT is linked to HI_LIM and OUT_HI is set to 1.

As long as the value at input IN is less than 0, output OUT is linked to LO_LIM and OUT_LO is set to 1.

For inverse scaling, you must assign LO_LIM > HI_LIM. With inverse scaling, the output value at output OUT decreases while the input value at input IN increases.

Every call of the "Scale values" instruction must be assigned a data area in which the instruction data are stored. In addition, when the instruction is inserted in the program, the "Call options" dialog is automatically opened, where you can create a data block (single instance) (e.g., SCALE_DB_1) or a multi-instance (e.g., SCALE_Instance_1) for the "Scale values" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT | Input value to be scaled in physical units |
| HI_LIM | Input | INT | High limit value of value range of OUT |
| LO_LIM | Input | INT | Low limit value of value range of OUT |
| OUT | Output | INT | Result of scaling |
| OUT_HI | Output | BOOL | 1 = Input value > 27648: OUT = HI_LIM |
| OUT_LO | Output | BOOL | 1 = Input value < 0: OUT = LO_LIM |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | This version has identical functions to version V1.0. |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Behavior in the event of overflow or underflow of analog values and fail-safe value output

**Note**

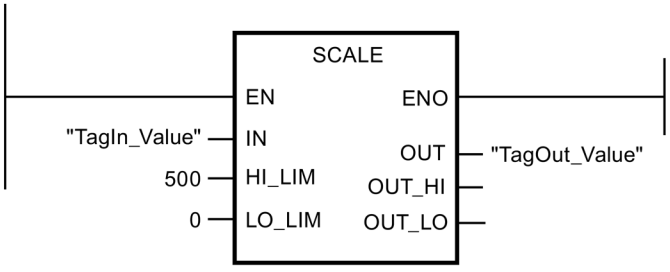
If inputs from the PII of an SM 336; AI 6 x 13Bit or SM 336; F-AI 6 x 0/4 ... 20 mA HART are used as input values, note that the F-system detects an overflow or underflow of a channel of this F-SM as an F-I/O fault or channel fault. The fail-safe value 0 is provided in place of $7FFF_H$ (for overflow) or $8000_H$ (for underflow) in the PII for the safety program.

If other fail-safe values should be output in this case, you need to evaluate the QBAD signal of the associated F-I/O or QBAD_I_xx signal / value status of the corresponding channel.

If the value in the PII of the F-SM is within the overrange or underrange, but is > 27648 or < 0, you can likewise branch to the output of an individual fail-safe value by evaluating outputs OUT_HI and OUT_LO, respectively.

## Example

The following example shows how the instruction works:




When operand "TagIn_Value" = 20000, the result is "TagOut_Value" 361.

## 13.2.10 Program control operations

### 13.2.10.1 ---( JMP ): Jump if RLO = 1 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Jump if RLO = 1" instruction to interrupt the linear execution of the program and resume it in another network. The destination network must be identified by a jump label (Page 459) (LABEL). The description of the jump label is specified in the placeholder above the instruction.

The specified jump label must be in the same block in which the instruction is executed. The name you specify can only occur once in the block.

If the result of logic operation (RLO) at the input of the instruction is "1" or the input is not connected, the jump to the network identified by the jump label is executed. The jump direction can be towards higher or lower network numbers.

If the result of logic operation (RLO) at the input of the instruction is "0", the program continues executing in the next network.

**Note**

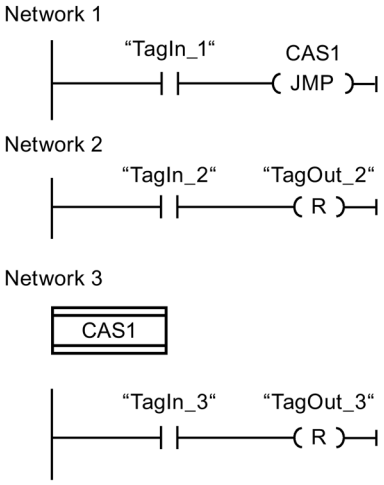
(S7-1200, S7-1500)
If the jump destination (jump label) for an instruction "JMP" or "JMPN" is above the associated instruction "JMP" or "JMPN" (backwards jump), you cannot insert any other instructions for program control (JMP, JMPN, RET, jump label) between them.
**Exception:** You can insert an instruction "JMP" or "JMPN" between them if you also insert the associated jump destination in between as well as below the associated instruction "JMP" or "JMPN" (forward jump).
Non-compliance can lead to compilation errors or to the CPU going to STOP.

**Note**

You are not permitted to insert any SENDDP or SENDS7 instructions between an instruction JMP or JMPN and the associated jump destination (jump label).

## Example

The following example shows how the instruction works:




When operand "TagIn_1" has signal state "1", the "Jump if RLO = 1" instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. When input "TagIn_3" has signal state "1", output "TagOut_3" is reset.

### 13.2.10.2 ---( JMPN ): Jump if RLO = 0 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Jump if RLO = 0" instruction to interrupt the linear execution of the program and resume it in another network, when the result of logic operation at the input of the instruction is "0". The destination network must be identified by a jump label (Page 459) (LABEL). The description of the jump label is specified in the placeholder above the instruction.

The specified jump label must be in the same block in which the instruction is executed. The name you specify can only occur once in the block.

If the result of logic operation (RLO) at the input of the instruction is "0", the jump to the network identified by the jump label is executed. The jump direction can be towards higher or lower network numbers.

If the result of logic operation (RLO) at the input of the instruction is "1", the program continues executing in the next network.

**Note**

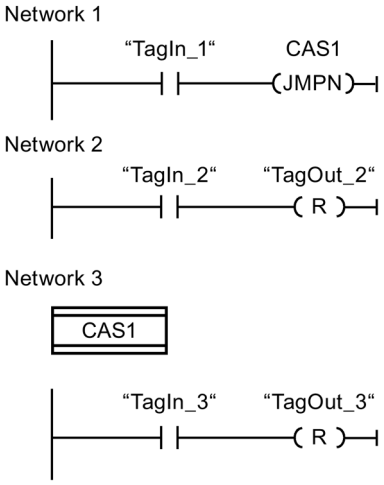
(S7-1200, S7-1500)
If the jump destination (jump label) for an instruction "JMP" or "JMPN" is above the associated instruction "JMP" or "JMPN" (backwards jump), you cannot insert any other instructions for program control (JMP, JMPN, RET, jump label) between them.
**Exception:** You can insert an instruction "JMP" or "JMPN" between them if you also insert the associated jump destination in between as well as below the associated instruction "JMP" or "JMPN" (forward jump).
Non-compliance can lead to compilation errors or to the CPU going to STOP.

**Note**

You are not permitted to insert any SENDDP or SENDS7 instructions between an instruction JMP or JMPN and the associated jump destination (jump label).

## Example

The following example shows how the instruction works:




When operand "TagIn_1" has signal state "0", the "Jump if RLO = 0" instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. When input "TagIn_3" has signal state "1", output "TagOut_3" is reset.

### 13.2.10.3 LABEL: Jump label (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

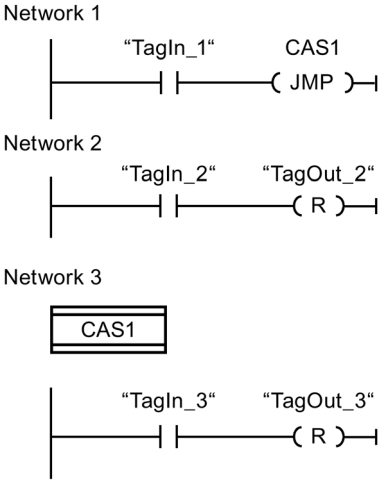
You can use a jump label to specify a destination network, in which the program execution should resume after a jump.

The jump label and the instruction in which the jump label is specified must be located in the same block. The name of a jump label can only be assigned once in a block.

Only one jump label can be placed in a network. To each jump label can be jumped from several locations.

## Example

The following example shows how the instruction works:




When operand "TagIn_1" has signal state "1", the "Jump if RLO = 1" instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. When input "TagIn_3" has signal state "1", output "TagOut_3" is reset.

## See also

---( JMP ): Jump if RLO = 1 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 455)

---( JMPN ): Jump if RLO = 0 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 457)

--(RET): Return (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 460)

### 13.2.10.4 --(RET): Return (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Return" instruction to stop the processing of a block.

If the result of logic operation (RLO) at the input of the "Return" instruction is "1", program execution is terminated in the currently called block and continued in the calling block (for example, in the main safety block) after the call function. If the RLO at the input of the "Return" instruction is "0", the instruction is not executed. Program execution continues in the next network of the called block.

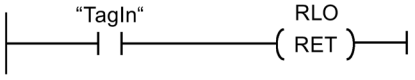
Influencing the status of the call function (ENO) is irrelevant, because the enable output "ENO" cannot be connected.

**Note**

You cannot program a RET instruction before a main safety block call.

## Example

The following example shows how the instruction works:




When the "TagIn" operand delivers signal state "1", the "Return" instruction is executed. Program execution is terminated in the called block and continues in the calling block.

## See also

---( JMP ): Jump if RLO = 1 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 455)

---( JMPN ): Jump if RLO = 0 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 457)

LABEL: Jump label (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 459)

### 13.2.10.5 ---(OPN): Open global data block (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

You can use the "Open global data block" instruction to open a data block. The number of the data block is transferred to the DB register. Subsequent DB commands access the relevant blocks depending on the register contents.

**Note**

Note when using the "Open global data block" instruction that the content of the DB register can be changed following calls of F-FB/F-FC and "fully qualified DB accesses," such that there is no guarantee that the last data block you opened with "Open global data block" is still open.

You should therefore use the following method for addressing data to avoid errors when accessing data of the DB register:

- Use symbolic addressing.
- Use only fully qualified DB accesses.

If you still want to use the "Open global data block" operation, you must ensure that the DB register is restored by repeating the "Open global data block" instruction following calls of F-FB/F-FC and "fully qualified DB accesses." Otherwise, a malfunction could result.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Data block> | Input | BLOCK_DB | Data block that is opened |

## "Fully qualified DB access"

The initial access to data of a data block in an F-FB/F-FC must always be a "fully qualified DB access," or it must be preceded by the "Open global data block" instruction. This also applies to the initial access to data of a data block after a jump label.

An example of "fully qualified DB access" and "non-fully qualified DB access" is provided in Restrictions in the programming languages FBD/LAD (Page 90).

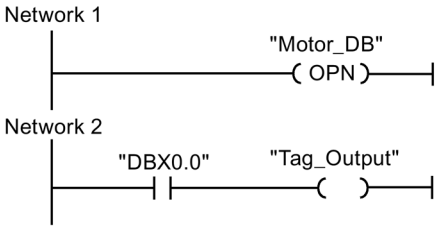
## Access to instance DBs

You can also access instance DBs of F-FBs with fully qualified access, e.g., for transfer of block parameters. It is not possible to access static local data in single/multi-instances of other F-FBs.

Note that accessing instance DBs of F-FBs that are not called in the safety program can cause the F-CPU to go to STOP mode.

### Example

The following example shows how the instruction works:




The "Motor_DB" data block is called in network 1. The number of the data block is transferred to the DB register. The "DBX0.0" operand is queried in network 2. The signal state of the "DBX0.0" operand is assigned to the "Tag_Output" operand.

## 13.2.11 Word logic operations

### 13.2.11.1 AND: AND logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

### Description

You can use the "AND logic operation" instruction to combine the value at input IN1 to the value at input IN2 bit-by-bit by AND logic and query the result at output OUT.

When the instruction is executed, bit 0 of the value at input IN1 and bit 0 of the value at input IN2 are ANDed. The result is stored in bit 0 of output OUT. The same logic operation is executed for all other bits of the specified values.

The result bit has signal state "1" only when both of the bits in the logic operation also have signal state "1". If one of the two bits of the logic operation has signal state "0", the corresponding result bit is reset.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

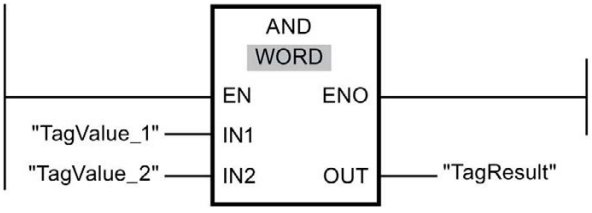
### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | WORD | First value of logic operation |
| IN2 | Input | WORD | Second value of logic operation |
| OUT | Output | WORD | Result of the instruction |

## Example

The following example shows how the instruction works:




| | |
|---|---|
| IN1 | "TagValue_1" = 01010101 01010101 |
| IN2 | "TagValue_2" = 00000000 00001111 |
| OUT | "TagResult" = 00000000 00000101 |

The instruction is always executed regardless of the signal state at enable input "EN". The value of the "TagValue_1" operand and the value of the "TagValue_2" operand are ANDed. The result is mapped bit-by-bit and output in the "TagResult" operand.

### 13.2.11.2 OR: OR logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "OR logic operation" instruction to connect the value at input IN1 input to the value at input IN2 bit-by-bit by OR logic and query the result at output OR.

When the instruction is executed, bit 0 of the value at input IN1 and bit 0 of the value at input IN2 are ORed. The result is stored in bit 0 of output OUT. The same logic operation is executed for all bits of the specified tags.

The result bit has signal state "1" when at least one of the two bits in the logic operation has signal state "1". If both of the bits of the logic operation have signal state "0", the corresponding result bit is reset.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

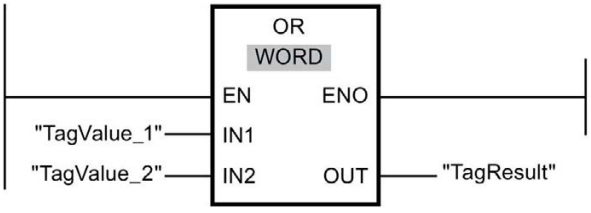
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | WORD | First value of logic operation |
| IN2 | Input | WORD | Second value of logic operation |
| OUT | Output | WORD | Result of the instruction |

### Example

The following example shows how the instruction works:




| | |
|---|---|
| IN1 | "TagValue_1" = 01010101 01010101 |
| IN2 | "TagValue_2" = 00000000 00001111 |
| OUT | "TagResult" = 01010101 01011111 |

The instruction is always executed regardless of the signal state at enable input "EN". The value of the "TagValue_1" operand and the value of the "TagValue_2" operand are ORed. The result is mapped bit-by-bit and output in the "TagResult" operand.

## 13.2.11.3 XOR: EXCLUSIVE OR logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

### Description

You can use the "EXCLUSIVE OR logic operation" instruction to combine the value at input IN1 and the value at input IN2 bit-by-bit by EXCLUSIVE OR logic and query the result at output OUT.

When the instruction is executed, bit 0 of the value at input IN1 input and bit 0 of the value at input IN2 are logically combined by EXCLUSIVE OR. The result is stored in bit 0 of output OUT. The same logic operation is executed for all other bits of the specified value.

The result bit has signal state "1" when one of the two bits in the logic operation has signal state "1". If both of the bits of the logic operation have signal state "1" or "0", the corresponding result bit is reset.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

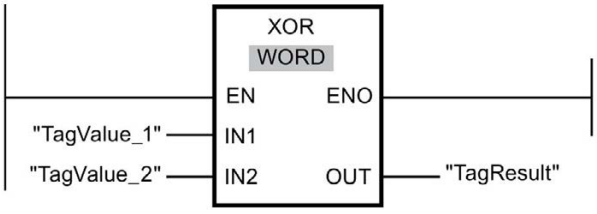
### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | WORD | First value of logic operation |
| IN2 | Input | WORD | Second value of logic operation |
| OUT | Output | WORD | Result of the instruction |

## Example

The following example shows how the instruction works:




| | |
|---|---|
| IN1 | "TagValue_1" = 01010101 01010101 |
| IN2 | "TagValue_2" = 00000000 00001111 |
| OUT | "TagResult" = 01010101 01011010 |

The instruction is always executed regardless of the signal state at enable input "EN". The value of the "TagValue_1" operand and the value of the "TagValue_2" operand are logically combined by EXCLUSIVE OR. The result is mapped bit-by-bit and output in the "TagResult" operand.

## 13.2.12 Shift and rotate

### 13.2.12.1 SHR: Shift right (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

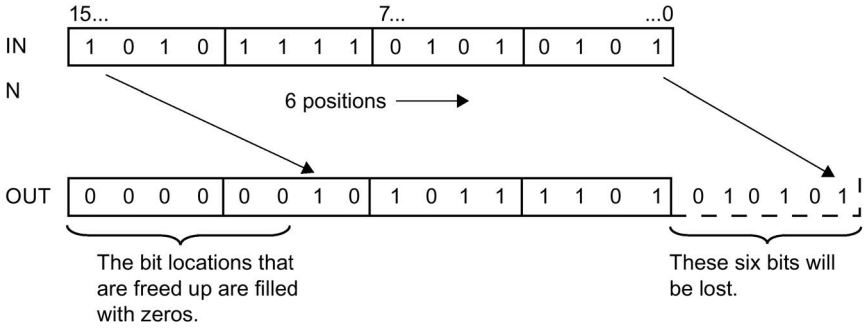
Use the "Shift right" instruction to shift the content of the operand at input IN bit-by-bit to the right and query the result at output OUT. Use input N to specify the number of bit positions by which the specified value is to be moved.

If the value at input N is "0", the value at input IN is copied to the operand at output OUT.

If the value at input N is greater than the number of available bit positions, the operand value at input IN is shifted to the right by the available number of bit positions.

The bit locations that are freed up in the left area of the operand during the shift operation are filled with zeros.

The following figure shows how the content of an operand of data type WORD is moved by 6 bit positions to the right:




Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

S7-300/400:

Only the low-byte is evaluated from input N.

S7-1200/1500:

If the value at input N < 0, the output OUT is set to 0.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | WORD | Value that is shifted |
| N | Input | INT | Number of bit positions by which the value is shifted |
| OUT | Output | WORD | Result of the instruction |

## Instruction versions

A number of versions are available for this instruction:

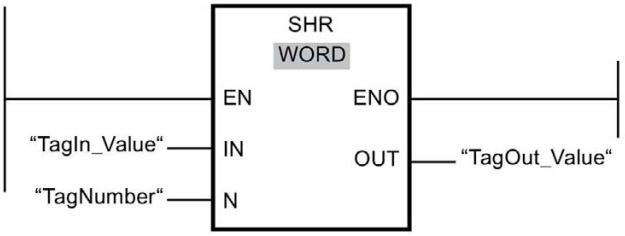
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | These versions have identical functions to version V1.0. |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




The following table shows how the instruction works using specific operand values:

| Parameter | Operand | Value |
|---|---|---|
| IN | TagIn_Value | 0011 1111 1010 1111 |
| N | TagNumber | 3 |
| OUT | TagOut_Value | 0000 0111 1111 0101 |

The instruction is always executed regardless of the signal state at enable input "EN". The content of the operand "TagIn_Value" is moved three bit positions to the right. The result is output at output "TagOut_Value".

### 13.2.12.2 SHL: Shift left (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

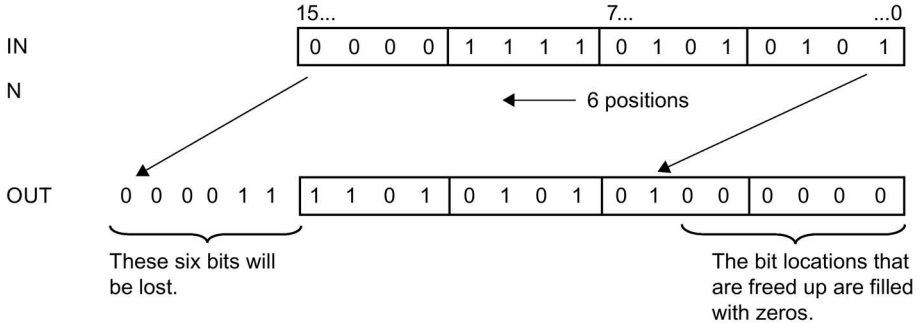
Use the "Shift left" instruction to shift the content of the operand at input IN bit-by-bit to the left and query the result at output OUT. Use input N to specify the number of bit positions by which the specified value is to be moved.

If the value at input N is "0", the value at input IN is copied to the operand at output OUT.

If the value at input N is greater than the number of available bit positions, the operand value at input IN is shifted to the left by the available number of bit positions.

The bit positions that are freed up in the right area of the operand during the shift operation are filled with zeros.

The following figure shows how the content of an operand of data type WORD is moved by 6 bit positions to the left:




Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

S7-300/400:

Only the low-byte is evaluated from input N.

S7-1200/1500:

If the value at input N < 0, the output OUT is set to 0.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | WORD | Value that is shifted |
| N | Input | INT | Number of bit positions by which the value is shifted |
| OUT | Output | WORD | Result of the instruction |

## Instruction versions

A number of versions are available for this instruction:

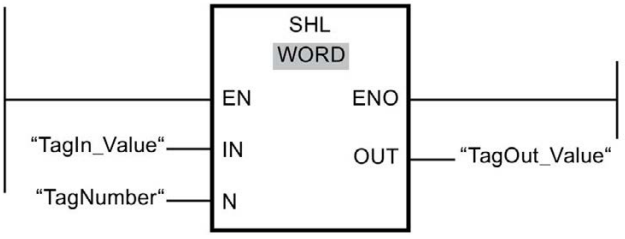
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | These versions have identical functions to version V1.0. |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




The following table shows how the instruction works using specific operand values:

| Parameter | Operand | Value |
|---|---|---|
| IN | TagIn_Value | 0011 1111 1010 1111 |
| N | TagNumber | 4 |
| OUT | TagOut_Value | 1111 1010 1111 0000 |

The instruction is always executed regardless of the signal state at enable input "EN". The content of the operand "TagIn_Value" is moved four bit positions to the left. The result is output at output "TagOut_Value".

## 13.2.13 Operating

### 13.2.13.1 ACK_OP: Fail-safe acknowledgment (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description (S7-300, S7-400)

This instruction enables fail-safe acknowledgment from an HMI system. It allows, for example, reintegration of F-I/O to be controlled from the HMI system. Acknowledgment takes place in two steps:

- Input/output parameter IN changes to a value of 6 for exactly one cycle.
- Input/output parameter IN changes to a value of 9 within a minute for exactly one cycle

Once the in/out parameter IN has changed to a value of 6, the instruction evaluates whether this parameter has changed to a value of 9 after 1 second, at the earliest, or one minute, at the latest. Output OUT (output for acknowledgment) is then set to 1 for one cycle.

If an invalid value is input or if in/out parameter IN has not changed to the value 9 within one minute or the change occurred before one second has elapsed, then in/out parameter IN is reset to 0, and both steps listed above must be repeated.

During the time in which in/out parameter IN must change from 6 to the value 9, output Q is set to 1. Otherwise, Q has a value of 0.

Every call of the "Fail-safe acknowledgment" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., ACK_OP_DB_1) or a multi-instance (e.g., ACK_OP_Instance_1) for the "Fail-safe acknowledgment" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

**Note**

A separate data area must be used for each call of ACK_OP. Each call can be processed only once in an F-runtime group cycle.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

The two acknowledgment steps must **not** be triggered by one single operation, for example by automatically storing them along with the time conditions in a program and using a function key to trigger them.
Having two separate acknowledgment steps also prevents erroneous triggering of an acknowledgment by your non-fail-safe HMI system. *(S013)*

**⚠ WARNING**

If you have HMI systems and F-CPUs that are interconnected and use the ACK_OP instruction for fail-safe acknowledgment, you need to ensure that the intended F-CPU will be addressed **before** you perform the two acknowledgment steps.

- To do this, store a network-wide* unique name for the F-CPU in a DB of your standard user program in each F-CPU.
- In your HMI system, set up a field from which you can read out the F-CPU name from the DB online before executing the two acknowledgment steps.
- Optional:

  in your HMI system, set up a field to permanently store the F-CPU name. Then, you can determine whether the intended F-CPU is being addressed by simply comparing the F-CPU name read out online with the permanently stored name.*(S014)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

**Note**

You can read out output Q by means of HMI systems or, if applicable, you can evaluate it in your standard user program.

You can provide the in/out IN with a separate memory word or DBW of a DB of the standard user program supply for each instance of the ACK_OP instruction.

## Description (S7-1200, S7-1500)

This instruction enables fail-safe acknowledgment from an HMI system. It allows, for example, reintegration of F-I/O to be controlled from the HMI system. Acknowledgment takes place in two steps:

- Input/output parameter IN changes to a value of 6 for exactly one cycle.
- Input/output parameter IN changes to the value at the ACK_ID input within a minute for exactly one cycle

Once the in/out parameter IN has changed to a value of 6, the instruction evaluates whether this parameter has changed to a value at the ACK_ID input after 1 second, at the earliest, or one minute, at the latest. Output OUT (output for acknowledgment) is then set to 1 for one cycle.

If an invalid value is input or if in/out parameter IN has not changed to the value at the ACK_ID input within one minute or the change occurred before one second has elapsed, then in/out parameter IN is reset to 0, and both steps listed above must be repeated.

During the time in which in/out parameter IN must change from 6 to the value at the ACK_ID input, output Q is set to 1. Otherwise, Q has a value of 0.

Every call of the "Fail-safe acknowledgment" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., ACK_OP_DB_1) or a multi-instance (e.g., ACK_OP_Instance_1) for the "Fail-safe acknowledgment" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

**Note**

A separate data area must be used for each call of ACK_OP. Each call can be processed only once in an F-runtime group cycle.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

The two acknowledgment steps must **not** be triggered by one single operation, for example by automatically storing them along with the time conditions in a program and using a function key to trigger them.
Having two separate acknowledgment steps also prevents erroneous triggering of an acknowledgment by your non-fail-safe HMI system. *(S013)*

**⚠ WARNING**

If you have HMI systems and F-CPUs that are interconnected and use the ACK_OP instruction for fail-safe acknowledgment, you need to ensure that the intended F-CPU will be addressed **before** you perform the two acknowledgment steps.

Alternative 1:

- The value for each identifier of the acknowledgment (ACK_ID input; data type: INT) can be freely selected in the range from 9 to 30000, but must be unique network-wide* for all instances of the ACK_OP instruction.
  You must supply the ACK_ID input with constant values when calling the instruction. Direct read or write access in the associated instance DB is not permitted in the safety program!

Alternative 2:

- To do this, store a network-wide* unique name for the F-CPU in a DB of your standard user program in each F-CPU.
- In your HMI system, set up a field from which you can read out the F-CPU name from the DB online before executing the two acknowledgment steps.
- Optional:
  in your HMI system, set up a field to permanently store the F-CPU name. Then, you can determine whether the intended F-CPU is being addressed by simply comparing the F-CPU name read out online with the permanently stored name. *(S047)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

**Note**

You can read out output Q by means of HMI systems or, if applicable, you can evaluate it in your standard user program.

You need to provide the in/out IN with a separate memory word or DBW of a DB of the standard user program supply for each instance of the ACK_OP instruction.

## Parameters (S7-300, S7-400)

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | InOut | INT | Input variable from HMI system |
| OUT | Output | BOOL | Output for acknowledgment |
| Q | Output | BOOL | Time status |

## Parameters (S7-1200, S7-1500)

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ACK_ID | Input | INT | Identifier of the acknowledgment (9 to 30000) |
| IN | InOut | INT | Input variable from HMI system |
| OUT | Output | BOOL | Output for acknowledgment |
| Q | Output | BOOL | Time status |

## Instruction versions

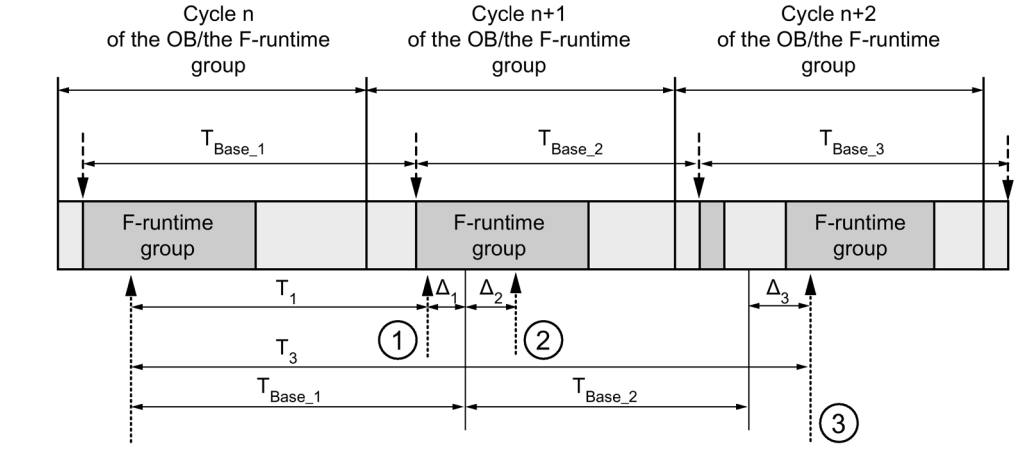
A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions are identical in function to version V1.0 for S7-300/400 F-CPUs.<br>The input ACK_ID must also be taken into consideration for S7-1200/1500 F-CPUs. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Timing imprecision resulting from the update time of the time base used in the instruction:




① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

An example how the instruction is used is available under Implementing User Acknowledgment in the Safety Program of the F-CPU of a DP Master or IO controller (Page 151).

## See also

Implementing user acknowledgment in the safety program of the F-CPU of a I-slave or I-device (S7-300, S7-400, S7-1500) (Page 156)

## 13.2.14 Additional instructions

### 13.2.14.1 ---| |--- OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

#### Description

You can use the "Get status bit OV" instruction to detect whether a number range overflow occurred in the last arithmetic instruction processed. The "Get status bit OV" instruction functions like a normally open contact. If the query is fulfilled, the instruction has signal state "1". If the condition is not fulfilled the instruction has signal state "0".

The "Get status bit OV" evaluation must be inserted in the network that follows the instruction that influences the OV. This network must not contain any jump labels.

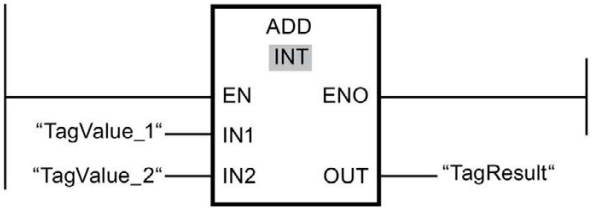
**Note**

The execution time of the OV-affecting instruction is extended when the "Get status bit OV " instruction is used (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).

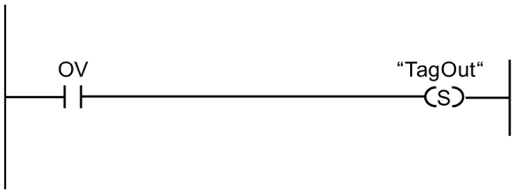
#### Example

The following example shows how the instruction works:

Network 1:




Network 2:




The "Add" instruction is always executed (regardless of the signal state at enable input EN).

The value of the "TagValue_1" operand is added to value of the TagValue_2 operand. The result of the addition is stored in the "TagResult" operand.

If an overflow occurs during execution of the "Add" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

### 13.2.14.2 ---| / |--- OV: Get negated status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

You can use the "Get negated status bit OV" instruction to detect whether a number range overflow occurred in the last arithmetic instruction processed. The "Get negated status bit OV" instruction functions like a normally closed contact. If the query is satisfied, the instruction has signal state "0". If the query not satisfied, the instruction has signal state "1".

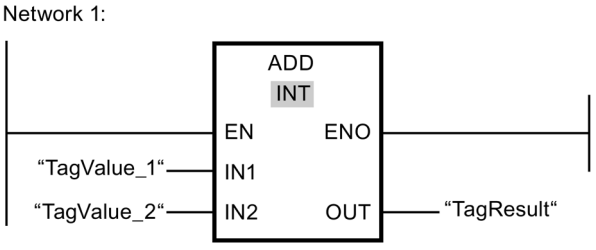
The "Get negated status bit OV" evaluation must be inserted in the network following the instruction that influences the OV. This network must not contain any jump labels.

**Note**

The execution time of the OV-affecting instruction is extended when the "Get negated status bit OV " instruction is used (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).

## Example

The following example shows how the instruction works:





The "Add" instruction is always executed (regardless of the signal state at enable input EN).

The value of the "TagValue_1" operand is added to value of the TagValue_2 operand. The result of the addition is stored in the "TagResult" operand.

If an overflow does *not* occur during execution of the "Add" instruction, the status bit OV is reset to "0". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## 13.2.15 Communication

### 13.2.15.1 PROFIBUS/PROFINET

#### SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500)

#### Introduction

You use the SENDDP and RCVDP instructions for fail-safe sending and receiving of data using:

- Safety-related master-master communication
- Safety-related master-master communication for S7 Distributed Safety
- Safety-related master-I-slave communication
- Safety-related I-slave-I-slave communication
- Safety-related IO controller-IO controller communication
- Safety-related IO controller-IO controller communication for S7 Distributed Safety
- Safety-related IO controller-I-device communication
- Safety-related IO controller-I-slave communication

#### Description

The SENDDP instruction sends 16 data elements of data type BOOL and 2 data elements of data type INT or one data element of the data type DINT (S7-1500) in a fail-safe manner to another F-CPU via PROFIBUS DP/PROFINET IO. The data can be received there by the related RCVDP instruction.

Every call of this instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., RCVDP_DB_1) for these instructions. Once it is created, the new data block can be found in the "STEP 7 Safety" folder in the project tree under "Program blocks > System blocks". For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

With the SENDDP instruction, the data to be sent (for example, outputs of other F-blocks/instructions) are available at input SD_BO_xx or SD_I_xx or SD_DI_00 as alternative.

With the RCVDP instruction, the data received are available at outputs RD_BO_xx and RD_I_xx or RD_DI_00 as alternative for additional processing by other F-blocks/instructions.

(S7-1500) At the DINTMODE input of the SENDDP instruction you specify if the data at the inputs SD_I_00 and SD_I_01 or the data at the input SD_DI_00 is sent.

The operating mode of the F-CPU with the SENDDP instruction is provided at output SENDMODE. If the F-CPU with the SENDDP instruction is in disabled safety mode, output SENDMODE = 1.

Communication between F-CPUs takes place in the background by means of a special safety protocol. You must define the communication relationship between a SENDDP instruction in one F-CPU and a RCVDP instruction in the other F-CPU by specifying an address relationship at the DP_DP_ID inputs of the SENDDP and RCVDP instructions. Associated SENDDP and RCVDP instructions are assigned the same value for DP_DP_ID.

**⚠ WARNING**

The value for each address relationship (DP_DP_ID input; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

**Note**

Within a safety program, you must assign a different start address (S7-300, S7-400) or HW identifier (S7-1200, S7-1500) for every call of the SENDDP and RCVDP instructions at input LADDR.

A separate instance DB must be used for each call of the SENDDP and RCVDP instructions. You must not declare and call these instructions as multi-instances.

The input and outputs of the RCVDP instruction cannot be initialized with temporary or static local data of the main safety block.

The inputs of the RCVDP instruction cannot be initialized with outputs (using fully qualified DB accesses) of a RCVDP or RCVS7 instruction called in an upstream network.

The RD_D_00 output must not be evaluated for DINTMODE = 0; the RD_I_xx outputs of the RCVDP instruction must not be evaluated for DINTMODE = 1.

(S7-1500) The outputs of the SENDDP and RCVDP instructions must not be supplied with tags from the standard user program. Exception: RET_DPRD, RET_DPWR and DIAG outputs.

You cannot use an actual parameter for an output of an RCVDP instruction, if it is already being used for an input of the same or another RCVDP or RCVS7 instruction.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You are not permitted to insert any SENDDP instructions between an instruction JMP or JMPN and the associated jump destination (jump label).

You cannot insert a RET instruction prior to a SENDDP instruction.

## SENDDP parameter

The following table shows the parameters of the SENDDP instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| SD_BO_00 | Input | BOOL | Send data BOOL 00 |
| ... | | | ... |
| SD_BO_15 | Input | BOOL | Send data BOOL 15 |
| SD_I_00 | Input | INT | Send data INT 00 |
| SD_I_01 | Input | INT | Send data INT 01 |
| SD_DI_00 | Input | DINT | (S7-1500)<br>(hidden)<br>Send data DINT 00 |
| DINTMODE | Input | DINT | (S7-1500)<br>(hidden)<br>0=SD_I_00 u. SD_I_01 are sent<br>1=SD_DI_00 is sent |
| DP_DP_ID | Input | INT | Network-wide unique value for the address relationship between a SENDDP and RCVDP instruction |
| TIMEOUT | Input | TIME | Monitoring time in ms for safety-related communication (see also Monitoring and response times (Page 661)) |
| LADDR | Input | INT | The start address (S7-300, S7-400) or HW identifier (S7-1200, S7-1500) of the address area/transfer area:<br>• Of DP/DP coupler for safety-related master-master communication<br>• For safety-related master-I-slave communication<br>• For safety-related I-slave-I-slave communication<br>• Of PN/PN coupler for safety-related IO controller-IO controller communication<br>• For safety-related IO controller-I-device communication<br>• For safety-related IO controller-I-slave communication |
| ERROR | Output | BOOL | 1=Communication error |
| SUBS_ON | Output | BOOL | 1=RCVDP outputs fail-safe values |
| RET_DPRD | Output | WORD | Error code RET_VAL of the DPRD_DAT instruction<br>(for a description of error codes, refer to the online help for DPRD_DAT instruction ("Extended instructions > Distributed I/O > Other".)) |

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| RET_DPWR | Output | WORD | Error code RET_VAL of the DPWR_DAT instruction (for a description of error codes, refer to the online help for DPWR_DAT instruction ("Extended instructions > Distributed I/O > Other".)) |
| DIAG | Output | BYTE | Service information |

## RCVDP parameter:

The following table shows the parameters of the RCVDP instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ACK_REI | Input | BOOL | 1=Acknowledgment for reintegration of send data following communication error |
| SUBBO_00 | Input | BOOL | Fail-safe value for receive data BOOL 00 |
| ... | | | ... |
| SUBBO_15 | Input | BOOL | Fail-safe value for receive data BOOL 15 |
| SUBI_00 | Input | INT | Fail-safe value for receive data INT 00 |
| SUBI_01 | Input | INT | Fail-safe value for receive data INT 01 |
| SUBDI_00 | Input | DINT | (S7-1500)<br>(hidden)<br>Fail-safe value for receive data DINT 00 |
| DP_DP_ID | Input | INT | Network-wide unique value for the address relationship between a SENDDP and RCVDP instruction |
| TIMEOUT | Input | TIME | Monitoring time in ms for safety-related communication<br>(see also Monitoring and response times (Page 661)) |
| LADDR | Input | INT | The start address (S7-300, S7-400) or HW identifier (S7-1200, S7-1500) of the address area/transfer area:<br>• Of DP/DP coupler for safety-related master-master communication<br>• For safety-related master-I-slave communication<br>• For safety-related I-slave-I-slave communication<br>• Of PN/PN coupler for safety-related IO controller-IO controller communication<br>• For safety-related IO controller-I-device communication<br>• For safety-related IO controller-I-slave communication |
| ERROR | Output | BOOL | 1=Communication error |
| SUBS_ON | Output | BOOL | 1=Fail-safe values are output |
| ACK_REQ | Output | BOOL | 1=Acknowledgment for reintegration of send data required |
| SENDMODE | Output | BOOL | 1=F-CPU with SENDDP instruction in disabled safety mode |
| RD_BO_00 | Output | BOOL | Receive data BOOL 00 |
| ... | | | ... |
| RD_BO_15 | Output | BOOL | Receive data BOOL 15 |
| RD_I_00 | Output | INT | Receive data INT 00 |
| RD_I_01 | Output | INT | Receive data INT 01 |

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| RD_DI_00 | Output | DINT | (S7-1500)<br>(hidden)<br>Receive data DINT 00 |
| RET_DPRD | Output | WORD | Error code RET_VAL of the DPRD_DAT instruction (for a description of error codes, refer to the online help for DPRD_DAT instruction ("Extended instructions > Distributed I/O > Other".)) |
| RET_DPWR | Output | WORD | Error code RET_VAL of the DPWR_DAT instruction (for a description of error codes, refer to the online help for DPWR_DAT instruction ("Extended instructions > Distributed I/O > Other".)) |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for these instructions:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | This version has identical functions to version V1.0. |
| 1.3 | x | — | x | S7-300/400: This version has identical functions to version V1.0.<br>S7-1500: Instead of 2 data of data type INT, one data of data type DINT can be sent/received as alternative. Otherwise identical function as version V1.0. |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Placement

The RCVDP instruction must be inserted at the start of the main safety block and the SENDDP instruction at the end.

## Startup characteristics

After the sending and receiving F-systems are started up, communication must be established between the connection partners for the fist time (SENDDP and RCVDP instructions). During this time, the receiver (RCVDP instruction) outputs the fail-safe values present at its inputs SUBBO_xx and SUBI_xx or alternatively SUBDI_00.

The SENDDP and RCVDP instructions signal this at output SUBS_ON with 1. Output SENDMODE has a default of 0 and is not updated, as long as output SUBS_ON = 1.

## Behavior in the event of communication errors

If a communication error occurs, for example, due to a signature error (CRC) or when monitoring time TIMEOUT expires, outputs ERROR and SUBS_ON = 1 are set. The receiver (RCVDP instruction) then outputs the fail-safe values assigned at its SUBBO_xx and SUBI_xx or alternatively SUBDI_00 inputs. Output SENDMODE is not updated while output SUBS_ON = 1.

The send data of the SENDDP instruction present at inputs SD_BO_xx and SD_I_xx, SD_DI_00 as alternative, are only output again when communication errors are no longer detected (ACK_REQ = 1) and you acknowledge (Page 151) the RCVDP instruction with a positive edge at input ACK_REI.

**⚠ WARNING**

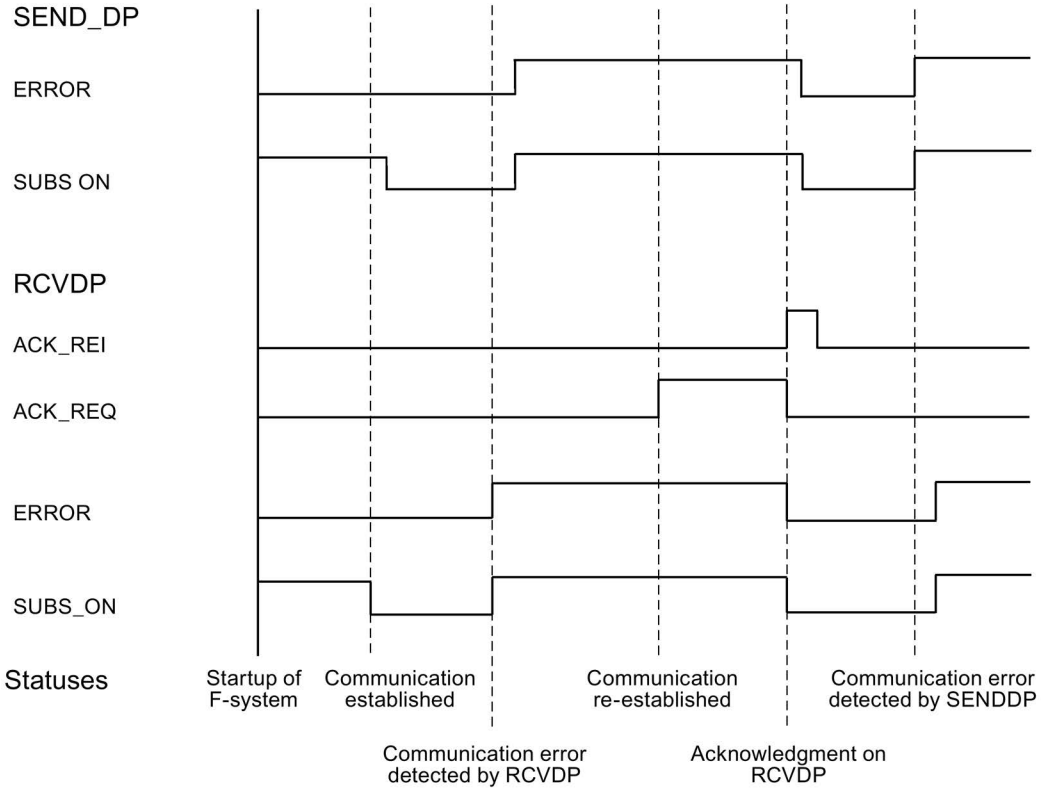
For the user acknowledgment, you must interconnect input ACK_REI with a signal generated by the operator input.

An interconnection with an automatically generated signal is not permitted. *(S040)*

Note that output ERROR (1=communication error) for a communication error will not be set unless communication between the connection partners (SENDDP and RCVDP instructions) has been previously established. If communication cannot be established after startup of the sending and receiving F-Systems, check the configuration of the safety-related CPU-CPU communication, the parameter assignment of the SENDDP and the RCVDP instructions, and the bus connection. You also obtain information on possible error causes by evaluating outputs RET_DPRD and RETDP_WR.

In general, always evaluate RET_DPRD and RETDP_WR, since it is possible that only one of the two outputs will contain error information.

## Timing diagrams SENDDP/RCVDP




## Output DIAG

In addition, non-fail-safe information about the type of communication errors that occurred is provided at output DIAG of the SENDDP and RCVDP instructions for service purposes.

You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until you acknowledge the errors at input ACK_REI of the RCVDP instruction.

## Structure of DIAG of the SENDDP/RCVDP instruction

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Reserved | — | — |
| Bit 1 | Reserved | — | — |
| Bit 2 | Reserved | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Timeout of SENDDP/RCVDP detected | Interference in bus connection to partner F-CPU. | Check bus connection and ensure that no external fault sources are present. |
| | | Monitoring time setting for F-CPU and partner F-CPU is too low. | Check assigned monitoring time TIMEOUT for SENDDP and RCVDP of both F-CPUs. If possible, set a higher value. Recompile safety program |
| | | DP/DP coupler or PN/PN coupler configuration is invalid. | Check DP/DP coupler or PN/PN coupler configuration. |
| | | Internal fault of DP/DP coupler or PN/PN coupler | Replace DP/DP coupler or PN/PN coupler |
| | | CP in STOP mode, or internal fault in CP | Switch CP to RUN mode, check diagnostic buffer of CP, and replace CP, if necessary |
| | | F-CPU/partner F-CPU in STOP mode, or internal fault in F-CPU/partner F-CPU | Switch F-CPUs to RUN mode, check diagnostic buffer of F-CPUs, and replace F-CPUs, if necessary |
| Bit 5 | Sequence number error of SENDDP/RCVDP detected | See description for bit 4 | See description for bit 4 |
| Bit 6 | CRC-error of SENDDP/RCVDP detected | See description for bit 4 | See description for bit 4 |
| Bit 7 | Reserved | — | — |

## See also

Configuring and programming communication (S7-300, S7-400) (Page 163)

Safety-related IO controller-IO controller communication (Page 166)

Safety-related master-master communication (Page 175)

Safety-related communication between IO controller and I-device (Page 185)

Safety-related master-I-slave communication (Page 192)

Safety-related IO controller-I-slave communication (Page 211)

Communication with S7 Distributed Safety via PN/PN coupler (IO controller-IO controller communication) (Page 220)

Communication with S7 Distributed Safety via DP/DP coupler (master-master communication) (Page 221)

### 13.2.15.2 S7 communication

## SENDS7 and RCVS7: Communication via S7 connections (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

### Introduction

You use the SENDS7 and RCVS7 instructions for fail-safe sending and receiving of data using S7 connections.

**Note**

In *STEP 7 Safety Advanced*, S7 connections are generally permitted over Industrial Ethernet only.

Safety-related communication via S7 connections is possible from and to F-CPUs with PROFINET interface or S7-400 F-CPUs with PROFINET-capable CPs. See also Safety-related communication via S7 connections (Page 212).

### Description

The SENDS7 instruction sends the send data contained in an F-communication DB to the F-communication DB of the associated RCVS7 instruction of another F-CPU in a fail-safe manner using an S7 connection.

Every call of this instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., SENDS7_DB_1) or a multi-instance (e.g., SENDS7_Instance_1) for this instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

Information on the F-communication DB is contained in "Programming safety-related communication via S7 connections (Page 215)".

An F-communication DB is an F-DB for safety-related CPU-CPU communication with special properties. You must specify the numbers of the F-communication DBs at inputs SEND_DB and RCV_DB of instructions SENDS7 and RCVS7.

The operating mode of the F-CPU with the SENDS7 instruction is provided at output SENDMODE of the RCVS7 instruction. If the F-CPU with the SENDS7 instruction is in disabled safety mode, then output SENDMODE = 1.

To reduce the bus load, you can temporarily shut down communication between the F-CPUs at input EN_SEND of the SENDS7 instruction. To do so, supply input EN_SEND with "0" (default = "1"). In this case, send data are no longer sent to the F-communication DB of the associated RCVS7 instruction, and the receiver provides fail-safe values for this period of time (initial values in its F-communication DB). If communication was already established between the partners, a communication error is detected.

You must specify the local ID - from the perspective of the F-CPU - of the S7 connection (from the connection table in the network view) at input ID of the SENDS7 instruction (see also Configuring (Page 37)).

Communication between F-CPUs takes place in the background by means of a special safety protocol. You must define a communication relationship between an SENDS7 instruction in one F-CPU and a communication relationship between an RCVS7 instruction and the other F-CPU by assigning an odd number at input R_ID (of the SENDS7 and RCVS7 instructions). Associated SENDS7 and RCVS7 instructions receive the same value for R_ID.

**⚠ WARNING**

The value for each address relationship (R_ID input; data type: DWORD) is user-defined; however, it must be an odd number and unique network-wide* for all safety-related communication connections. The value R_ID + 1 is internally assigned and must not be used.

You must supply inputs ID and R_ID with constant values when calling the instruction. Direct read or write access to the associated instance DB is not permitted in the safety program! *(S020)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

**Note**

A separate instance DB must be used for each call of the SENDS7 and RCVS7 instructions within a safety program. You must not declare and call these instructions as multi-instances.

The input and outputs of the RCVS7 instruction cannot be initialized with temporary or static local data of the main safety block.

The inputs of the RCVS7 instruction cannot be initialized with outputs (using fully qualified DB accesses) of a RCVS7 or RCVDP instruction called in an upstream network.

You cannot use an actual parameter for an output of an RCVS7 instruction, if it is already being used for an input of the same or another RCVS7 or RCVDP instruction. The F-CPU can go to STOP if this is not observed. A diagnostic event is entered in the diagnostic buffer of the F-CPU.

**Note**

You must not program any SENDS7 instruction between a JMP or JMPN instruction and the associated destination network of the JMP or JMPN instruction.

You cannot program a RET instruction prior to a SENDS7 instruction.

## SENDS7 parameter

The following table shows the parameters of the SENDS7 instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| SEND_DB | Input | BLOCK_DB | Number of F-communication DB |
| TIMEOUT | Input | TIME | Monitoring time in ms for safety-related communication (see also Monitoring and response times (Page 661)) |
| EN_SEND | Input | BOOL | 1= Send enable |
| ID | Input | WORD | Local ID of the S7 connection |
| R_ID | Input | DWORD | Network-wide unique value for an address relationship between a SENDS7 instruction and a RCVS7 instruction |
| ERROR | Output | BOOL | 1=Communication error |
| SUBS_ON | Output | BOOL | 1=Receiving block outputs fail-safe values |
| STAT_RCV | Output | WORD | Status parameter STATUS of the URCV instruction (You can find a description of error codes in the online help for the URCV instruction ("Communication > S7 Communication")) |
| STAT_SND | Output | WORD | Status parameter STATUS of the USEND instruction (You can find a description of error codes in the online help for the USEND instruction ("Communication > S7 Communication")) |
| DIAG | Output | BYTE | Service information |

## RCVS7 parameter

The following table shows the parameters of the RCVS7 instruction.

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ACK_REI | Input | BOOL | Acknowledgment for reintegration of send data after communication error |
| RCV_DB | Input | BLOCK_DB | Number of F-communication DB |
| TIMEOUT | Input | TIME | Monitoring time in ms for safety-related communication (see also Monitoring and response times (Page 661)) |
| ID | Input | WORD | Local ID of the S7 connection |
| R_ID | Input | DWORD | Network-wide unique value for an address relationship between a SENDS7 instruction and a RCVS7 instruction |
| ERROR | Output | BOOL | 1=Communication error |
| SUBS_ON | Output | BOOL | 1=Fail-safe values are output |
| ACK_REQ | Output | BOOL | 1=Acknowledgment for reintegration of send data required |
| SENDMODE | Output | BOOL | 1=F-CPU with the SENDS7 instruction in disabled safety mode |
| STAT_RCV | Output | WORD | Status parameter STATUS of the URCV instruction (You can find a description of error codes in the online help for the URCV instruction ("Communication > S7 Communication")) |
| STAT_SND | Output | WORD | Status parameter STATUS of the USEND instruction (You can find a description of error codes in the online help for the USEND instruction ("Communication > S7 Communication")) |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for these instructions:

| Version | S7-300/400 | S7-1500 | Function |
|---|---|---|---|
| 1.0 | x | — | |
| 1.1 | x | — | This version has identical functions to version V1.0.<br>It also supports later versions of internally called instructions.<br>When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.1 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.2 | x | — | This version has identical functions to version V1.0/1.1.<br>It also supports later versions of internally called instructions. |

When a new F-CPU is created with *STEP 7 Safety Advanced*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Startup characteristics

After the sending and receiving F-systems are started up, communication must be established between the connection partners for the fist time (SENDS7 and RCVS7 instructions). The receiver (RCVS7 instruction) provides fail-safe values for this time period (initial values in its F-communication DB).

The SENDS7 and RCVS7 instructions signal this at output SUBS_ON with 1. Output SENDMODE (RCVS7 instruction) has a default of 0 and is not updated, as long as output SUBS_ON = 1.

## Behavior in the event of communication errors

If a communication error occurs, for example, due to a signature error (CRC) or when monitoring time TIMEOUT expires, outputs ERROR and SUBS_ON = 1 are set. The receiver (RCVS7 instruction) then provides fail-safe values (initial values in its F-communication DB). Output SENDMODE is not updated while output SUBS_ON = 1.

The send data present in the F-communication DB (SENDS7 instruction) are not output before the communication error is no longer detected ((ACK_REQ = 1)) and you acknowledge (Page 151) with a positive edge at input ACK_REI of the RCVS7 instruction.

**⚠ WARNING**

For the user acknowledgment, you must interconnect input ACK_REI with a signal generated by the operator input.

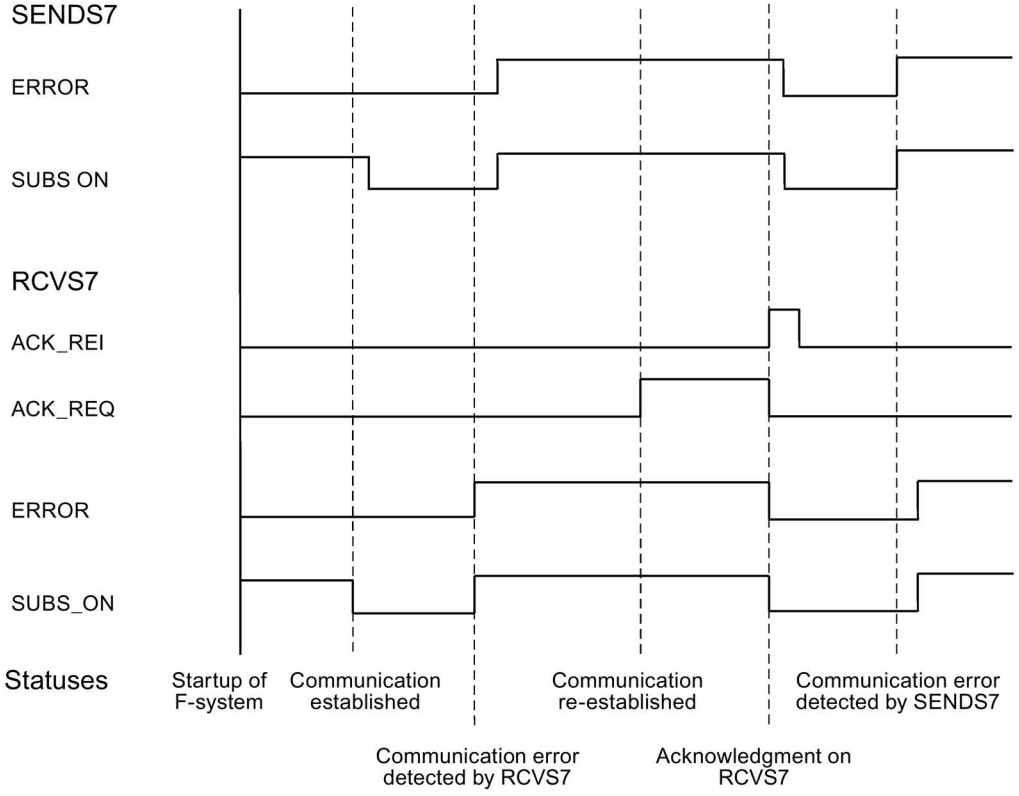
An interconnection with an automatically generated signal is not permitted. *(S040)*

Note that output ERROR (1=communication error) will be set for the first time on a communication error if communication has already been established between the connection partners (SENDS7 and RCVS7 instructions). If communication cannot be established after startup of the sending and receiving F-Systems, check the configuration of the safety-related CPU-CPU communication, the parameter assignment of the SENDS7 and the RCVS7 instructions, and the bus connection. You can also receive information on possible error causes by evaluating the STAT_RCV and STAT_SND outputs.

In general, always evaluate STAT_RCV and STAT_SND, since it is possible that only one of the two outputs will contain error information.

If one of the DIAG bits is set at output DIAG, also check whether the length and structure of the associated F-communication DB on both the sending and receiving ends match.

## Timing diagrams SENDS7 and RCVS7




## Output DIAG

Non-fail-safe information on the type of communication errors that have occurred is made available at output DIAG for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until you acknowledge them at input ACK_REI of the associated RCVS7 instruction.

## Structure of DIAG

| Bit no. | Assignment of SENDS7 and RCVS7 | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Reserved | — | — |
| Bit 1 | Reserved | — | — |
| Bit 2 | Reserved | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Timeout detected by SENDS7 and RCVS7 | Fault in bus connection to partner F-CPU | Check bus connection and ensure that no external fault sources are present. |
| | | Monitoring time setting for F-CPU and partner F-CPU is too low | Check assigned monitoring time TIMEOUT for SENDS7 and RCVS7 of both F-CPUs. If possible, set a higher value. Recompile safety program |
| | | CPs in STOP mode, or internal fault in CPs | • Switch CPs to RUN mode<br>• Check diagnostic buffer of CPs<br>• Replace CPs, if necessary |
| | | F-CPU/partner F-CPU in STOP mode, or internal fault in F-CPU/partner F-CPU | • Switch F-CPUs to RUN mode<br>• Check diagnostic buffer of F-CPUs<br>• Replace F-CPUs, if necessary |
| | | Communication was shut down with EN_SEND = 0. | Enable communication again at the associated SENDS7 with EN_SEND = 1 |
| | | S7 connection has changed, the IP address of the CP has changed, for example | Recompile the safety programs and download them to the F-CPUs |
| Bit 5 | Sequence number error detected by SENDS7 and RCVS7 | See description for bit 4 | See description for bit 4 |
| Bit 6 | CRC-error detected by SENDS7 and RCVS7 | See description for bit 4 | See description for bit 4 |
| Bit 7 | RCVS7:<br>Communication cannot be established | Configuration of the safety-related CPU-CPU communication is incorrect, parameter assignment of the SENDS7 and RCVS7 instructions is incorrect<br>See also description for Bit 4 | Check configuration of the safety-related CPU-CPU communication, the parameter assignment of the SENDS7 and the RCVS7 instructions is incorrect<br>See also description for Bit 4 |
| | SENDS7:<br>Reserved | — | — |

# 13.3 Instructions - FBD

## 13.3.1 General

### 13.3.1.1 New network (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Requirement

An F-block is open.

#### Procedure

To insert a new network, follow these steps:

1. Select the network after which you want to insert a new network.
2. Select the "Insert network" command in the shortcut menu.

**Note**

If you insert an element into the last empty network of the F-block in an FBD program, a new empty network is automatically inserted below it.

#### Result

A new empty network is inserted into the F-block.

### 13.3.1.2 Empty box (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Requirement

A network is available.

#### Procedure

To insert FBD elements into a network using an empty box, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to "Basic instructions > General > Empty box".
3. Use a drag-and-drop operation to move the "Empty box" element to the desired place in the network.
4. Hover the cursor over the yellow triangle in the top right corner of the empty box.

   A drop-down list is displayed.
5. Select the desired FBD element from the drop-down list.

If the instruction acts as a function block (FB) within the system, the "Call options" dialog opens. In this dialog, you can create an instance data block for the function block, either as a single instance or, if necessary, multi-instance, in which data of the inserted instruction are stored. Once it is created, the new instance data block can be found in the "Program resources" folder in the project tree under "Program blocks > System blocks". If you have selected "multi-instance", you can find it in the block interface in the "Static" section.

#### Result

The empty box is changed to the appropriate instruction. Placeholders are inserted for the parameters.

### 13.3.1.3 Open branch (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You use branches to program parallel connections with the Function Block Diagram (FBD) programming language. For this purpose, you use branches that you insert between the boxes. You can insert additional boxes in the branch, thereby programming complex function block diagrams.

#### Requirement

A network is available.

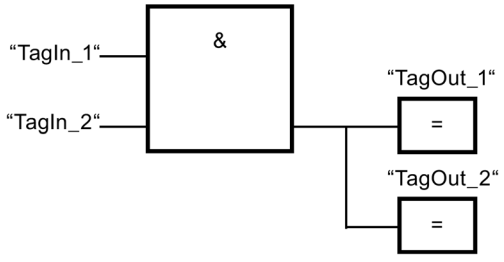
#### Procedure

To insert a new branch in a network, follow these steps:

1. Open the "Instructions" task card.
2. Navigate to "Basic instructions > General > Branch".
3. Use a drag-and-drop operation to move the element to the desired place in the network.

#### Example

The following figure provides an example of how to use branches:



### 13.3.1.4 Insert binary input (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

Use the "Insert binary input" instruction to expand the box of one of the following instructions by a binary input:

- "AND logic operation"
- "OR logic operation"
- "EXCLUSIVE OR logic operation"

You can query the signal state of several operands by expanding the instruction box.

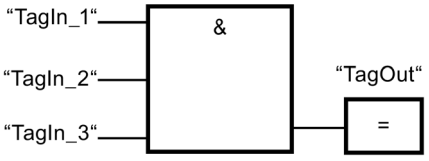
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Input | BOOL | The operand indicates the bit whose signal state will be queried. |

## Example

The following example shows how the instruction works:




The box of instruction "AND logic operation" has been expanded by an additional binary input at which the signal state of operand "TagIn_3" is queried. Output "TagOut" is set when the signal state of operands "TagIn_1", "TagIn_2", and "TagIn_3" is "1".

## See also

AND logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 499)

OR logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 500)

X: EXCLUSIVE OR logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 501)

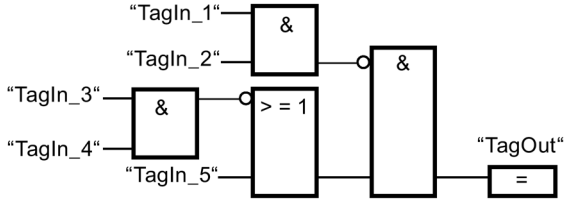
### 13.3.1.5 Invert RLO (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Invert RLO" instruction to invert the result of logic operation (RLO).

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Input "TagIn_1" or "TagIn_2" has signal state "0".
- Input "TagIn_3" or "TagIn_4" has the signal state "0" or input "TagIn_5" has the signal state "1".

## 13.3.2 Bit logic operations

### 13.3.2.1 AND logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "AND logic operation" instruction to query the signal states of two or more specified operands and evaluate them according to the AND truth table.

If the signal state of all the operands is "1", then the conditions are fulfilled and the instruction returns the result "1". If the signal state of one of the operands is "0", then the conditions are not fulfilled and the instruction generates the result "0".

If the "AND logic operation" instruction is the first instruction in a logic string, it saves the result of its signal state query in the RLO bit.

Each "AND logic operation" instruction that is not the first instruction in the logic string logically combines the result of its signal state query with the value saved in the RLO bit. This logical combination is performed according to the AND truth table.

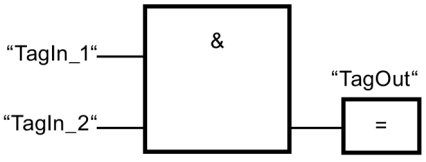
#### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Input | BOOL | The operand indicates the bit whose signal state will be queried. |

#### Example

The following example shows how the instruction works:




Output "TagOut" is set when the signal state of operands "TagIn_1" and "TagIn_2" is "1".

#### AND truth table

The following table shows the results when linking two operands by an AND logic operation:

| Signal state of the first operand | Signal state of the second operand | Result of logic operation |
|---|---|---|
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |

## See also

Insert binary input (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 497)

### 13.3.2.2 OR logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "OR logic operation" instruction to get the signal states of two or more specified operands and evaluate them according to the OR truth table.

If the signal state of at least one of the operands is "1", then the conditions are fulfilled and the instruction returns the result "1". If the signal state of all of the operands is "0", then the conditions are not fulfilled and the instruction generates the result "0".

If the "OR logic operation" instruction is the first instruction in a logic string, it saves the result of its signal state query in the RLO bit.

Each "OR logic operation" instruction that is not the first instruction in the logic string, logically combines the result of its signal state query with the value saved in the RLO bit. This logical combination is performed according to the OR truth table.

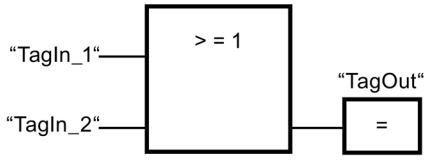
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Input | BOOL | The operand indicates the bit whose signal state will be queried. |

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the signal state of operand "TagIn_1" or "TagIn_2" is "1".

### OR truth table

The following table shows the results when linking two operands by an OR logic operation:

| Signal state of the first operand | Signal state of the second operand | Result of logic operation |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

### See also

Insert binary input (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 497)

## 13.3.2.3 X: EXCLUSIVE OR logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

### Description

You can use the "EXCLUSIVE OR logic operation" instruction to get the result of a signal state query according to the the EXCLUSIVE OR truth table.

With an "EXCLUSIVE OR logic operation" instruction, the signal state is "1" when the signal state of one of the two specified operands is "1". When more than two operands are queried, the overall result of logic operation is "1" if an odd-numbered quantity of queried operands returns the result "1".

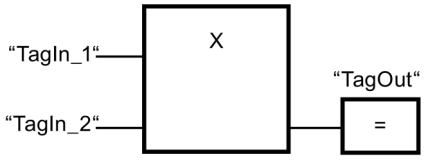
### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Input | BOOL | The operand indicates the bit whose signal state will be queried. |

### Example

The following example shows how the instruction works:




Output "TagOut" is set when the signal state of one of the two operands "TagIn_1" and "TagIn_2" is "1". When both operands have signal state "1" or "0" then output "TagOut" is reset.

## EXCLUSIVE OR truth table

The following table shows the results when two operands are linked by an EXCLUSIVE OR:

| Signal state of the first operand | Signal state of the second operand | Result of logic operation |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |
| 0 | 0 | 0 |

The following table shows the results when three operands are linked by an EXCLUSIVE OR:

| Signal state of the first operand | Signal state of the second operand | Signal state of the third operand | Result of logic operation |
|---|---|---|---|
| 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 |

## See also

Insert binary input (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 497)

### 13.3.2.4 =: Assignment (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Assignment" instruction to set the bit of a specified operand. If the result of logic operation (RLO) at the box input has signal state "1" or the box input for S7-1200/1500 F-CPUs is not connected, the specified operand is set to signal state "1". If the signal state at the box input is "0", the bit of the specified operand is reset to "0".

The instruction does not influence the RLO. The RLO at the box input is assigned directly to the operand located above the Assign box.

The "Assignment" instruction can be placed at any position in the logic operation sequence.

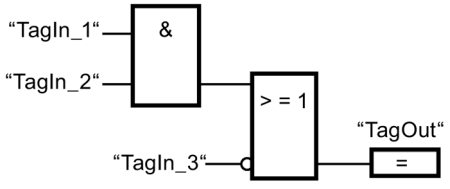
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Output | BOOL | Operand to which the RLO is assigned. |

## Example

The following example shows how the instruction works:




Operand "TagOut" at the output of the "Assignment" instruction is set when one of the following conditions is fulfilled:

- Inputs "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state at input "TagIn_3" is "0".

### 13.3.2.5 R: Reset output (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Reset output" instruction to reset the signal state of a specified operand to "0".

If the box input has signal state "1" or the box input for S7-1200/1500 F-CPUs is not connected, the specified operand is reset to "0". If there is a result of logic operation of "0" at the box input, the signal state of the specified operand remains unchanged.

The instruction does not influence the RLO. The RLO at the box input is transferred directly to the box output.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the operands of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

If the operand area "local data (temp)" is used for the operands of the instruction, the local data bit used must be initialized beforehand.

**Note**

You cannot use the "process image of the inputs", "process image of the outputs" from standard I/O and "standard DB" and "bit memory" operand areas for the operands of the instruction.

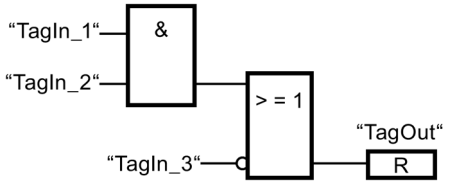
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Output | BOOL | Operand that is reset with RLO = "1". |

## Example

The following example shows how the instruction works:




Operand "TagOut" is reset when one of the following conditions is fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state of operand "TagIn_3" is "0".

### 13.3.2.6 S: Set output (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Set output" instruction to set the signal state of a specified operand to "1".

If the box input has signal state "1" or the box input for S7-1200/1500 F-CPUs is not connected, the specified operand is set to "1". If there is a result of logic operation of "0" at the box input, the signal state of the specified operand remains unchanged.

The instruction does not influence the RLO. The RLO at the box input is transferred directly to the box output.

**Note**

The instruction is not executed if it is applied to an output of an F-I/O that is passivated (e.g., during startup of the F-system). Therefore, it is preferable to access outputs of the F-I/O using only the "Assignment" instruction.

An F-I/O output is passivated if QBAD or QBAD_O_xx = 1 or value status = 0 is set in the corresponding F-I/O DB.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the operands of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

If the operand area "local data (temp)" is used for the operands of the instruction, the local data bit used must be initialized beforehand.

**Note**

You cannot use the "process image of the inputs", "process image of the outputs" from standard I/O and "standard DB" and "bit memory" operand areas for the operands of the instruction.

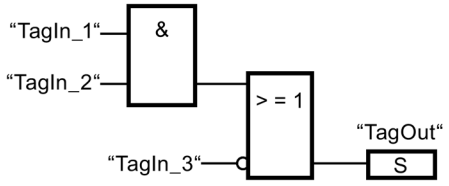
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand> | Output | BOOL | Operand that is set with RLO = "1". |

## Example

The following example shows how the instruction works:




Operand "TagOut" is set when one of the following conditions is fulfilled:

- Operands "TagIn_1" and "TagIn_2" have signal state "1".
- The signal state of operand "TagIn_3" is "0".

### 13.3.2.7 SR: Set/reset flip-flop (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Set/reset flip-flop" instruction to set or reset the bit of the specified operand based on the signal state of inputs S and R1. If the signal state at input S is "1" and the signal state at input R1 is "0", the specified operand is set to "1". If the signal state at input S is "0" and the signal state at input R1 is "1", the specified operand is reset to "0".

Input R1 takes priority over input S. If the signal state is "1" at the two inputs S and R1, the signal state of the specified operand is reset to "0".

The instruction is not executed if the signal state at the two inputs S and R1 is "0". The signal state of the operand then remains unchanged.

The current signal state of the operand is transferred to output Q and can be queried there.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the operands of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the operands of the instruction.

If the operand area "local data (temp)" is used for the edge bit memory of the instruction, the local data bit used must be initialized beforehand.

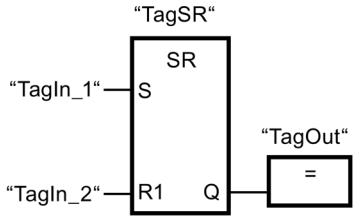
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| S | Input | BOOL | Enable setting |
| R1 | Input | BOOL | Enable resetting |
| <Operand> | Output | BOOL | Operand that is set or reset. |
| Q | Output | BOOL | Signal state of the operand |

## Example

The following example shows how the instruction works:




Operands "TagSR" and "TagOut" are set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- Operand "TagIn_2" has signal state "0".

Operands "TagSR" and "TagOut" are reset when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "0" and operand "TagIn_2" has signal state "1".
- Both operands "TagIn_1" and "TagIn_2" have signal state "1".

### 13.3.2.8 RS: Reset/set flip-flop (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Reset/set flip-flop" instruction to reset or set the bit of the specified operand based on the signal state of inputs R and S1. When the signal state is "1" at input R and "0" at input S1, the specified operand is reset to "0". When the signal state is "0" at input R and "1" at input S1, the specified operand is set to "1".

Input S1 takes priority over input R. If the signal state is "1" at the two inputs R and S1, the signal state of the specified operand is set to "1".

The instruction is not executed if the signal state at the two inputs R and S1 is "0". The signal state of the operand then remains unchanged.

The current signal state of the operand is transferred to output Q and can be queried there.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the operands of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the operands of the instruction.

If the operand area "local data (temp)" is used for the edge bit memory of the instruction, the local data bit used must be initialized beforehand.

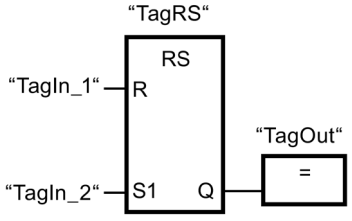
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| R | Input | BOOL | Enable resetting |
| S1 | Input | BOOL | Enable setting |
| <Operand> | Output | BOOL | Operand that is reset or set. |
| Q | Output | BOOL | Signal state of the operand |

## Example

The following example shows how the instruction works:




Operands "TagRS" and "TagOut" are reset when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- Operand "TagIn_2" has signal state "0".

Operands "TagRS" and "TagOut" are set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "0" and operand "TagIn_2" has signal state "1".
- Operands "TagIn_1" and "TagIn_2" have signal state "1".

### 13.3.2.9 P: Scan operand for positive signal edge (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Scan operand for positive signal edge" instruction to determine if there is a change from "0" to "1" in the signal state of a specified operand (<Operand1>). The instruction compares the current signal state of <Operand1> with the signal state of the previous query saved in <Operand2>. If the instruction detects a change in the result of logic operation from "0" to "1", there is a positive, rising edge.

If a rising edge is detected, the output of the instruction has signal state "1". In all other cases, the signal state at the output of the instruction is "0".

Enter the operand to be queried (<Operand1>) in the operand placeholder above the instruction. Enter the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

**Note**

The address of the edge memory bit must not be used more than once in the program, otherwise the edge memory bit would be overwritten. This would influence edge evaluation and the result would no longer be unequivocal.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the edge memory bit <Operand2> of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the edge memory bit <Operand2> of the instruction.

If operand area "local data (temp)" is used for the edge memory bit <Operand2> of the instruction, the local data bit used must be initialized beforehand.

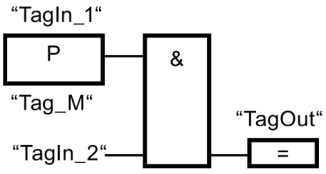
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | BOOL | Signal to be queried |
| <Operand2> | InOut | BOOL | Edge memory bit in which the signal state of the previous query is saved. |

## Example

The following example shows how the instruction works:




"TagOut" is set when the following conditions are fulfilled:

- There is a rising edge at input "TagIn_1".
- The signal state of operand "TagIn_2" is "1".

### 13.3.2.10 N: Scan operand for negative signal edge (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Scan operand for negative signal edge" instruction to determine if there is a change form "1" to "0" in the signal state of a specified operand. The instruction compares the current signal state of <Operand1> with the signal state of the previous query saved in <Operand2>. If the instruction detects a change in the result of logic operation from "1" to "0", there is a negative, falling edge.

If a falling edge is detected, the output of the instruction has signal state "1". In all other cases, the signal state at the output of the instruction is "0".

Enter the operand to be queried (<Operand1>) in the operand placeholder above the instruction. Enter the edge memory bit (<Operand2>) in the operand placeholder below the instruction.

**Note**

The address of the edge memory bit must not be used more than once in the program, otherwise the edge memory bit would be overwritten. This would influence edge evaluation and the result would no longer be unequivocal.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the edge memory bit <Operand2> of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the edge memory bit <Operand2> of the instruction.

If operand area "local data (temp)" is used for the edge memory bit <Operand2> of the instruction, the local data bit used must be initialized beforehand.

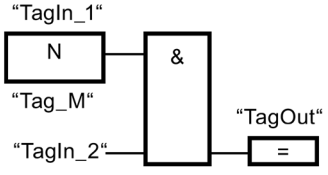
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Operand1> | Input | BOOL | Signal to be queried |
| <Operand2> | InOut | BOOL | Edge memory bit in which the signal state of the previous query is saved. |

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- There is a falling edge at input "TagIn_1".
- The signal state of operand "TagIn_2" is "1".

### 13.3.2.11 P_TRIG: Scan RLO for positive signal edge (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Scan RLO for positive signal edge" instruction to query a change in the signal state of the result of logic operation from "0" to "1". The instruction compares the current signal state of the result of logic operation with the signal state from the previous query, which is saved in the edge memory bit (<Operand>). If the instruction detects a change in the RLO from "0" to "1", there is a positive, rising edge.

If a rising edge is detected, the output of the instruction has signal state "1". In all other cases, the signal state at the output of the instruction is "0".

**Note**

The address of the edge memory bit must not be used more than once in the program, otherwise the edge memory bit would be overwritten. This would influence edge evaluation and the result would no longer be unequivocal.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the edge memory bit <Operand> of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the edge memory bit <Operand> of the instruction.

If operand area "local data (temp)" is used for the edge memory bit <Operand> of the instruction, the local data bit used must be initialized beforehand.

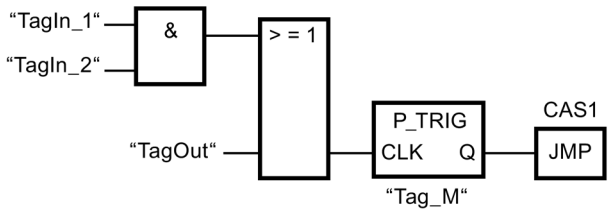
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CLK | Input | BOOL | Current RLO |
| <Operand> | InOut | BOOL | Edge memory bit in which the RLO of the previous query is saved. |
| Q | Output | BOOL | Result of edge evaluation |

## Example

The following example shows how the instruction works:




The RLO from the previous bit logic operation is saved in edge memory bit "Tag_M". If a change in the RLO signal state from "0" to "1" is detected, the program jumps to jump label CAS1.

## 13.3.2.12 N_TRIG: Scan RLO for negative signal edge (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Scan RLO for negative signal edge" instruction to query a change in the signal state of the result of logic operation from "1" to "0". The instruction compares the current signal state of the result of logic operation with the signal state from the previous query, which is saved in the edge memory bit (<Operand>). If the instruction detects a change in the RLO from "1" to "0", there is a negative, falling edge.

If a falling edge is detected, the output of the instruction has signal state "1". In all other cases, the signal state at the output of the instruction is "0".

**Note**

The address of the edge memory bit must not be used more than once in the program, otherwise the edge memory bit would be overwritten. This would influence edge evaluation and the result would no longer be unequivocal.

**Note**

If you want to use a formal parameter of an F-FB/F-FC for the edge memory bit <Operand> of the instruction, it must be declared as an input/output parameter.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You cannot use the "process image", "standard DB" and "bit memory" operand areas for the edge memory bit <Operand> of the instruction.

If operand area "local data (temp)" is used for the edge memory bit <Operand> of the instruction, the local data bit used must be initialized beforehand.

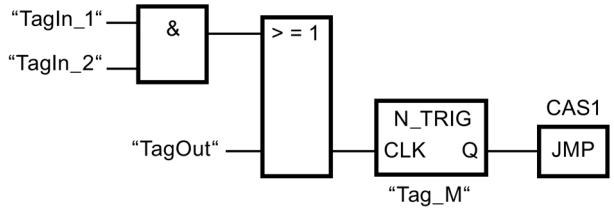
## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CLK | Input | BOOL | Current RLO |
| <Operand> | InOut | BOOL | Edge memory bit in which the RLO of the previous query is saved. |
| Q | Output | BOOL | Result of edge evaluation |

## Example

The following example shows how the instruction works:




The RLO of the previous bit logic operation is saved in edge bit memory "Tag_M". If a change in the RLO signal state from "1" to "0" is detected, the program jumps to jump label CAS1.

## 13.3.3 Safety functions

### 13.3.3.1 ESTOP1: Emergency STOP up to stop category 1 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

This instruction implements an emergency STOP shutdown with acknowledgment for Stop Categories 0 and 1.

Enable signal Q is reset to 0, as soon as input E_STOP takes a signal state of 0 (Stop category 0). Enable signal Q_DELAY is reset to 0 after the time delay set at input TIME_DEL (Stop Category 1).

Enable signal Q is reset to 1 not before input E_STOP takes a signal state of 1 and an acknowledgment occurs. The acknowledgment for the enable takes place according to the parameter assignment at input ACK_NEC:

- If ACK_NEC = 0, the acknowledgment is automatic.
- If ACK_NEC = 1, you must use a rising edge at input ACK for acknowledging the enable.

Output ACK_REQ is used to signal that a user acknowledgment is required at input ACK for the acknowledgment. The instruction sets output ACK_REQ to 1, as soon as input E_STOP = 1.

Following an acknowledgment, the instruction resets ACK_REQ to 0.

Every call of the "Emergency STOP up to Stop Category 1" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., ESTOP1_DB_1) or a multi-instance (e.g., ESTOP1_Instance_1) for the "Emergency STOP up to Stop Category 1" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

The ACK_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S033)*

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

**Note:** Only one emergency STOP signal (E_STOP) can be evaluated for the instruction. Discrepancy monitoring of the two NC contacts (when two channels are involved) in accordance with Categories 3 and 4 as defined in ISO 13849-1:2006 / EN ISO 13849-1:2008 is performed with suitable configuration (type of sensor interconnection: 2-channel equivalent) directly through the F-I/O with inputs. In order to keep the discrepancy time from influencing the response time, during the configuration of discrepancy behavior, you must configure "Supply value 0".

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| E_STOP | Input | BOOL | Emergency STOP |
| ACK_NEC | Input | BOOL | 1=Acknowledgment necessary |
| ACK | Input | BOOL | 1=Acknowledgment |
| TIME_DEL | Input | TIME | Time delay |
| Q | Output | BOOL | 1=Enable |
| Q_DELAY | Output | BOOL | Enable is OFF delayed |
| ACK_REQ | Output | BOOL | 1=Acknowledgment necessary |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | Version 1.0 requires that the F_TOF block with the number FB 186 is available in the project tree in the "Program blocks/System blocks/STEP 7 Safety" folder.<br>When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically. If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. You will then avoid number conflicts. |
| 1.1 | x | — | -— | These versions are functionally identical to version V1.0, but do not require the F_TOF block to have a particular number. |
| 1.2 | x | — | x | |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Startup characteristics

After an F-system startup, when ACK_NEC = 1, you must acknowledge the instruction using a rising edge at input ACK.

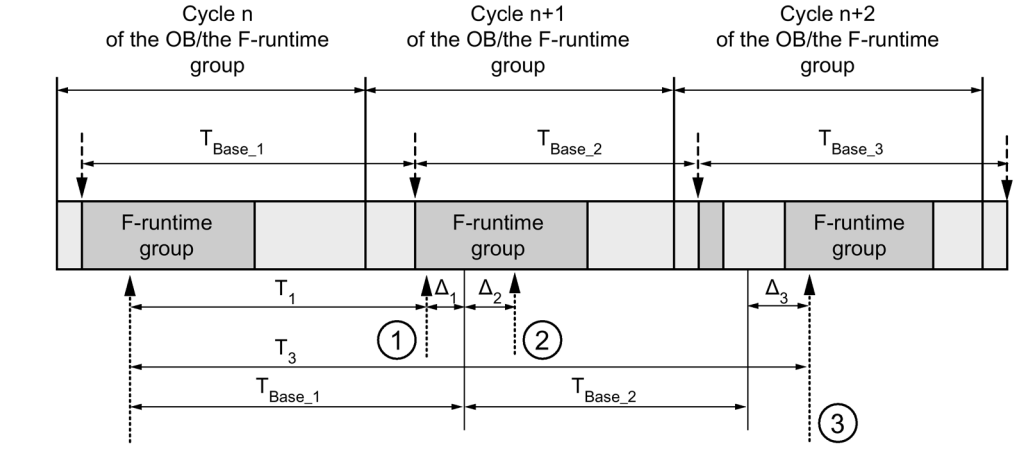
## Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits 1 to 5 are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Incorrect TIM_DEL setting | Time delay setting < 0 | Set time delay > 0 |
| Bit 1 | Reserved | — | — |
| Bit 2 | Reserved | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Acknowledgment not possible because emergency STOP is still active | Emergency STOP switch is locked | Release interlocking of emergency STOP switch |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of emergency STOP switch | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Emergency STOP switch is defective | Check emergency STOP switch |
| | | Wiring fault | Check wiring of emergency STOP switch |
| Bit 5 | If enable is missing: input ACK has a permanent signal state of 1 | Acknowledgment button defective | Check acknowledgment button |
| | | Wiring fault | Check wiring of acknowledgment button |
| Bit 6 | Acknowledgment required (= state of ACK_REQ) | — | — |
| Bit 7 | State of output Q | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




----► = Time base update

·····► = Call time of an instruction with time processing

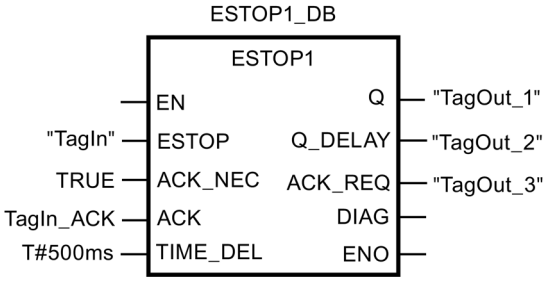
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

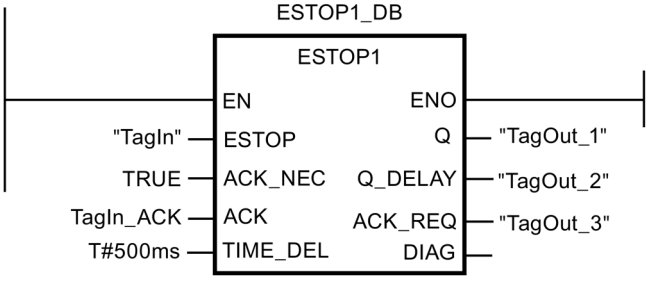
② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.3.3.2 TWO_HAND: Two-hand monitoring (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

This instruction implements two-hand monitoring.

**Note**

This instruction is only available for S7-300 and S7-400 F-CPUs. For S7-1200/1500 F-CPUs, you use the instruction "Two-hand monitoring with enable". The application "Two-hand monitoring with enable" replaces the instruction "Two-hand monitoring" with compatible functions.

If pushbuttons IN1 and IN2 are activated within the permitted discrepancy time DISCTIME ≤ 500 ms (IN1/IN2 = 1) (synchronous activation), output signal Q is set to 1. If the time difference between activation of pushbutton IN1 and pushbutton IN2 is greater than DISCTIME, then the pushbuttons must be released and reactivated.

Q is reset to 0 as soon as one of the pushbuttons is released (IN1/IN2 = 0). Enable signal Q can be reset to 1 only if the other pushbutton has been released, and if both pushbuttons are then reactivated within the discrepancy time. Enable signal Q can never be set to 1 if the discrepancy time is set to values less than 0 or greater than 500 ms.

Every call of the "Two-hand monitoring" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., TWO_HAND_DB_1) or a multi-instance (e.g., TWO_HAND_Instance_1) for the "Two-hand monitoring" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

The instruction supports the requirements in accordance with EN 574:1996 + A1:2008.

**Note:** Only one signal per pushbutton can be evaluated in the instruction. Discrepancy monitoring of the NC and NO contacts of pushbuttons IN1 and IN2 is performed directly during suitable configuration (type of sensor interconnection: 2-channel non equivalent) directly through the F-I/O with inputs. The normally open contact must be wired in such a way that it supplies the useful signal (see manual for the F-I/O you are using). In order to keep the discrepancy time from influencing the response time, you must assign "Supply value 0" for the behavior of discrepancy during configuration. If a discrepancy is detected, a fail-safe value of 0 is entered in the process image of the inputs (PII) for the pushbutton and QBAD or QBAD_I_xx = 1 is set in the relevant F-I/O DB. (See also F-I/O access (Page 122))

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

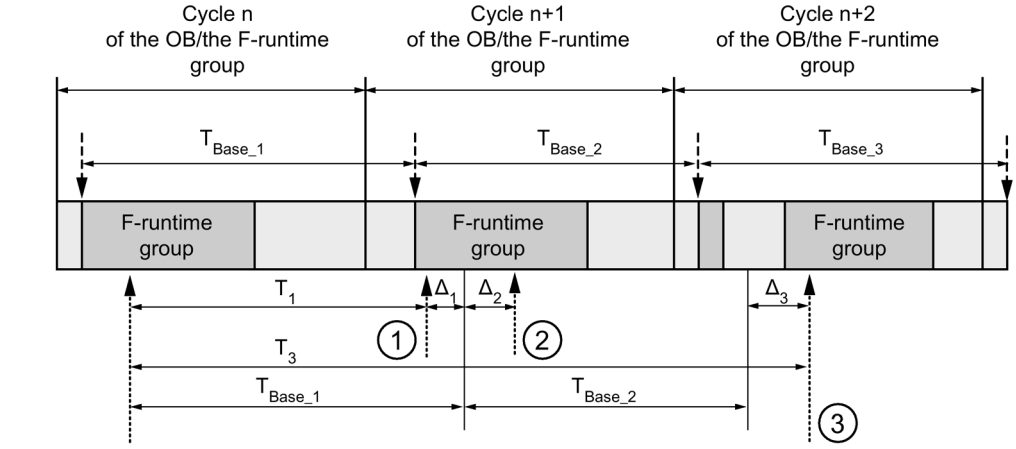
You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | BOOL | Pushbutton 1 |
| IN2 | Input | BOOL | Pushbutton 2 |
| DISCTIME | Input | TIME | Discrepancy time (0 to 500 ms) |
| Q | Output | BOOL | 1=Enable |

## Timing imprecision resulting from the update time of the time base used in the instruction:




----► = Time base update

·····► = Call time of an instruction with time processing

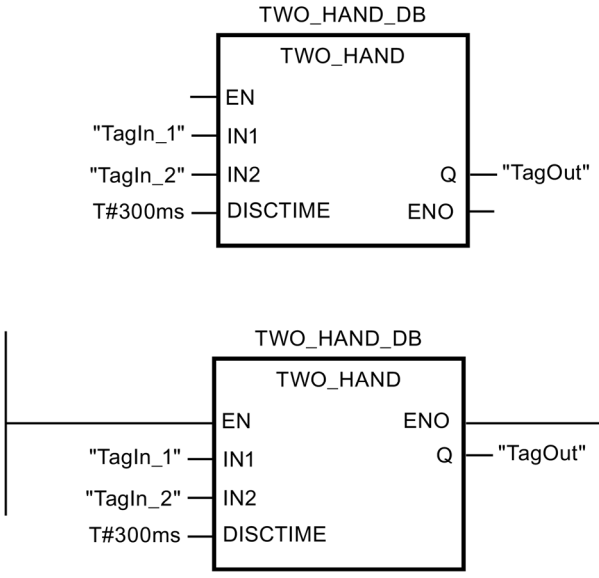
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.3.3.3 TWO_H_EN: Two-hand monitoring with enable (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction implements two-hand monitoring with enable.

If pushbuttons IN1 and IN2 are activated within the permitted discrepancy time DISCTIME ≤ 500 ms (IN1/IN2 = 1) (synchronous activation), output signal Q is set to 1 when existing ENABLE = 1. If the time difference between activation of pushbutton IN1 and pushbutton IN2 is greater than DISCTIME, then the pushbuttons must be released and reactivated.

Q is reset to 0 as soon as one of the pushbuttons is released (IN1/IN 2 = 0) or ENABLE = 0. Enable signal Q can be reset to 1 only if the other pushbutton has been released, and if both pushbuttons are then reactivated within the discrepancy time when existing ENABLE = 1.

Every call of the "Two-hand monitoring with enable" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., TWO_H_EN_DB_1) or a multi-instance (e.g., TWO_H_EN_Instance_1) for the "Two-hand monitoring with enable" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

The instruction supports the requirements in accordance with EN 574:1996 + A1:2008.

**Note:** Only one signal per pushbutton can be evaluated in the instruction. Discrepancy monitoring of the NC and NO contacts of pushbuttons IN1 and IN2 is performed directly during suitable configuration (type of sensor interconnection: 2-channel non equivalent) directly through the F-I/O with inputs. The normally open contact must be wired in such a way that it supplies the useful signal (see manual for the F-I/O you are using). In order to keep the discrepancy time from influencing the response time, during the configuration of discrepancy behavior, you must configure "Supply value 0".

If a discrepancy is detected, a fail-safe value of 0 is entered in the process image input (PII) for the pushbutton and QBAD or QBAD_I_xx = 1 or value status = 0 is set in the corresponding F-I/O DB.

**WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | BOOL | Pushbutton 1 |
| IN2 | Input | BOOL | Pushbutton 2 |
| ENABLE | Input | BOOL | Enable input |
| DISCTIME | Input | TIME | Discrepancy time (0 to 500 ms) |
| Q | Output | BOOL | 1=Enable |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

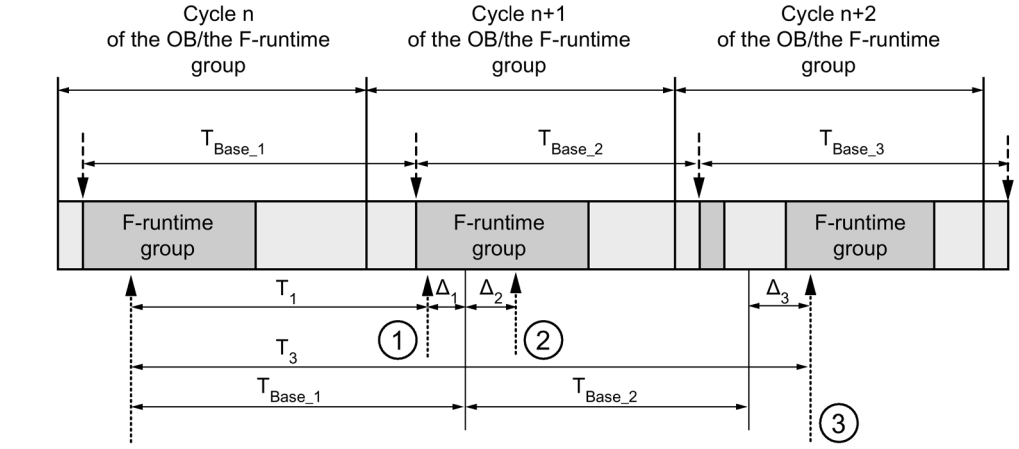
## Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits 0 to 5 are saved until the cause of the error has been eliminated.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Incorrect discrepancy time DISCTIME setting | Discrepancy time setting is <0 or > 500 ms | Set discrepancy time in range of 0 to 500 ms |
| Bit 1 | Discrepancy time elapsed | Discrepancy time setting is too low | If necessary, set a higher discrepancy time |
| | | Pushbuttons were not activated within the discrepancy time | Release pushbuttons and activate them within the discrepancy time |
| | | Wiring fault | Check wiring of pushbuttons |
| | | Pushbuttons defective | Check pushbuttons |
| | | Pushbuttons are wired to different F-I/O, and F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON on an F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 2 | Reserved | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Incorrect activation sequence | One pushbutton was not released | Release pushbuttons and activate them within the discrepancy time |
| | | Pushbuttons defective | Check pushbuttons |
| Bit 5 | ENABLE does not exist | ENABLE = 0 | Set ENABLE = 1, release pushbutton and activate it within the discrepancy time |
| Bit 6 | Reserved | — | — |
| Bit 7 | State of output Q | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




⇢ = Time base update

⇢ = Call time of an instruction with time processing

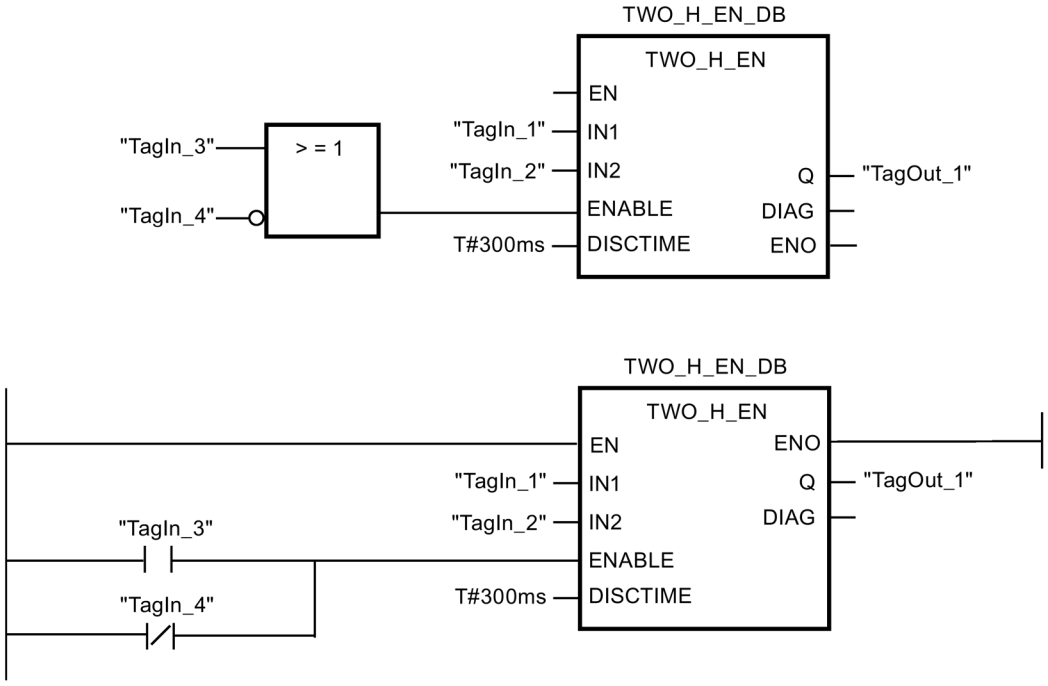
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.3.3.4 MUTING: Muting (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

This instruction performs parallel muting with two or four muting sensors.

**Note**

This instruction is only available for S7-300 and S7-400 F-CPUs. For S7-1200/1500 F-CPUs, you use the instruction "Parallel muting (Page 542)". The instruction "Parallel muting" replaces the instruction "Muting" with compatible functions.

Muting is a defined suppression of the protective function of light curtains. Light curtain muting can be used to introduce goods or objects into the danger area monitored by the light curtain without causing the machine to stop.

To utilize the muting function, at least two independently wired muting sensors must be present. The use of two or four muting sensors and correct integration into the production sequence must ensure that no persons enter the danger area while the light curtain is muted.

Every call of the "Muting" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., MUTING_DB_1) or a multi-instance (e.g., MUTING_Instance_1) for the "Muting" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

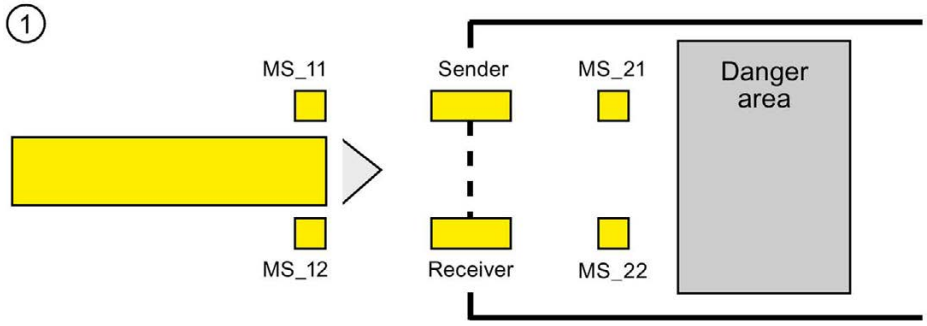
You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| MS_11 | Input | BOOL | Muting sensor 1 of sensor pair 1 |
| MS_12 | Input | BOOL | Muting sensor 2 of sensor pair 1 |
| MS_21 | Input | BOOL | Muting sensor 1 of sensor pair 2 |
| MS_22 | Input | BOOL | Muting sensor 2 of sensor pair 2 |
| STOP | Input | BOOL | 1=Conveyor system stopped |
| FREE | Input | BOOL | 1=Light curtain uninterrupted |
| QBAD_MUT | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal of the muting lamp channel |
| DISCTIM1 | Input | TIME | Discrepancy time of sensor pair 1 (0 to 3 s) |
| DISCTIM2 | Input | TIME | Discrepancy time of sensor pair 2 (0 to 3 s) |
| TIME_MAX | Input | TIME | Maximum muting time (0 to 10 min) |
| ACK | Input | BOOL | Acknowledgment of restart inhibit |
| Q | Output | BOOL | 1= Enable, not off |
| MUTING | Output | BOOL | Display of muting is active |
| ACK_REQ | Output | BOOL | Acknowledgment necessary |
| FAULT | Output | BOOL | Group error |
| DIAG | Output | BYTE | Service information |

### Schematic sequence of error-free muting procedure with 4 muting sensors (MS_11, MS_12, MS_21, MS_22)

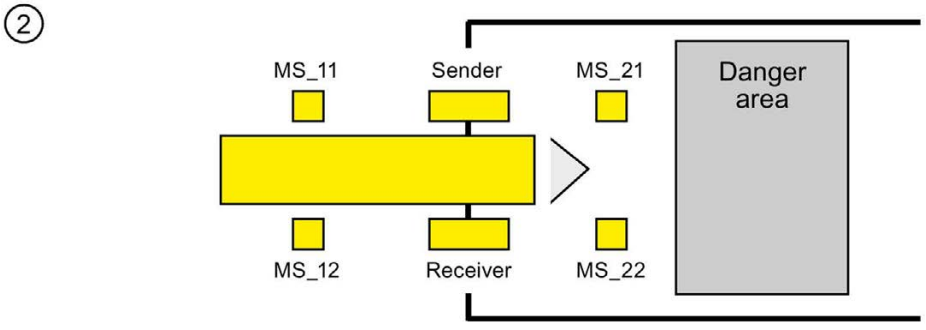



- If both muting sensors MS_11 and MS_12 are activated by the product within DISCTIM1 (apply signal state = 1), the instruction starts the MUTING function. Enable signal Q remains 1, even when input FREE = 0 (light curtain interrupted by product). The MUTING output for setting the muting lamp switches to 1.

**Note**

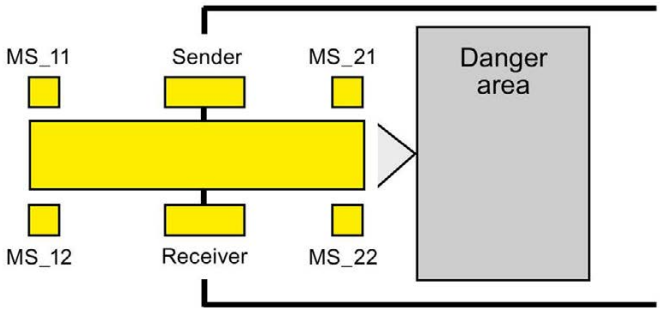
The muting lamp can be monitored using the QBAD_MUT input. To do this, you must wire the muting lamp to an output with wire break monitoring of an F-I/O and supply the QBAD_MUT input with the QBAD signal of the associated F-I/O or the QBAD_O_xx signal of the associated channel. If QBAD_MUT = 1, muting is terminated by the instruction. If monitoring of the muting lamp is not necessary, you do not have to supply input QBAD_MUT.

F-I/O that can promptly detect a wire break after activation of the muting operation must be used (*see manual for specific F-I/O*).



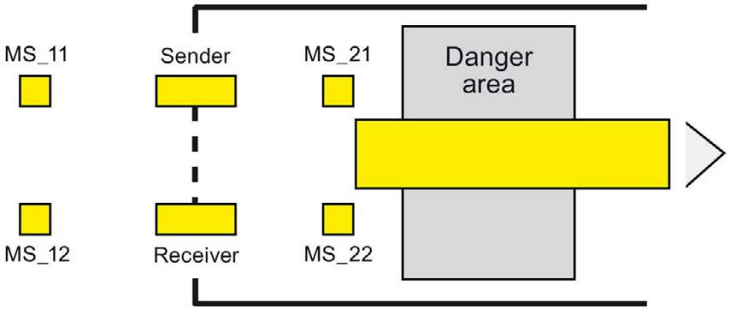

- As long as both muting sensors MS_11 and MS_12 continue to be activated, the MUTING function of the instruction causes Q to remain 1 and MUTING to remain 1 (so that the product can pass through the light curtain without causing the machine to stop).

③




- The two muting sensors MS_21 and MS_22 must be activated (within DISCTIM2) before muting sensors MS_11 and MS_12 are switched to inactive (apply signal state 0). In this way, the instruction retains the MUTING function. (Q = 1, MUTING = 1).

④

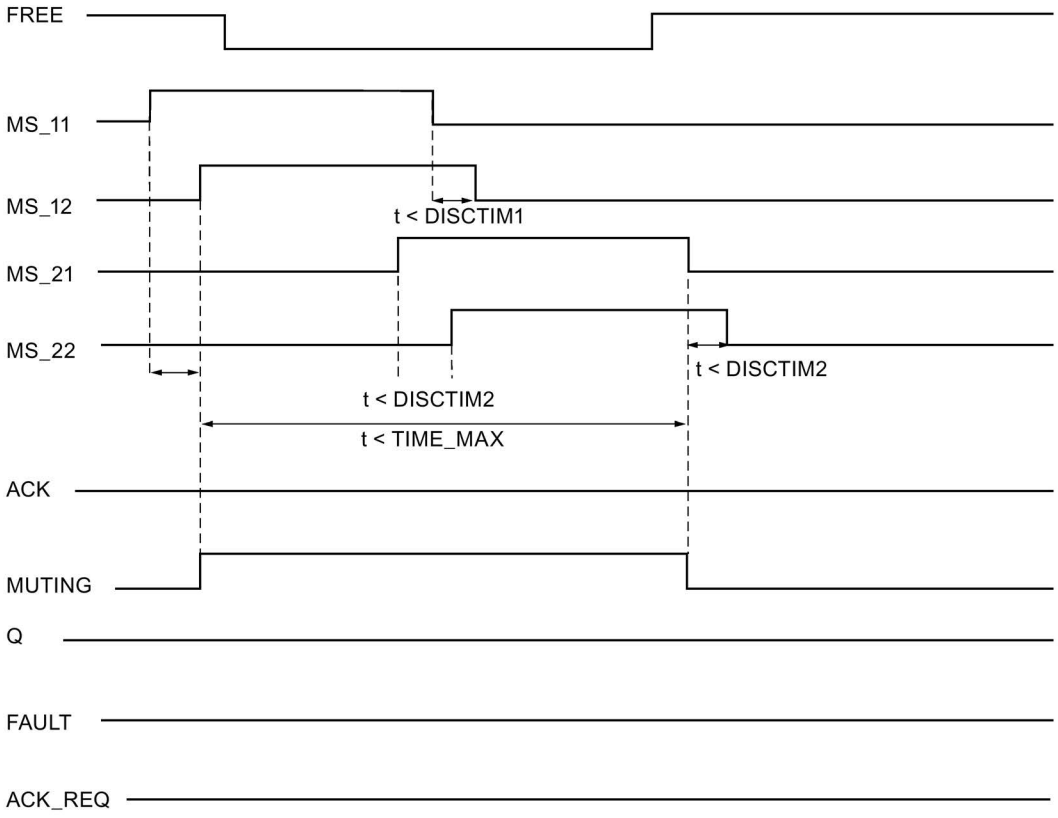



- Only if one of the two muting sensors MS_21 and MS_22 is switched to inactive (product enables sensors) is the MUTING function terminated (Q = 1, MUTING = 0). The maximum activation time for the MUTING function is the time set at input TIME_MAX.

**Note**

The MUTING function is also started if the product passes the light curtain in the reverse direction and the muting sensors are thus activated by the product in reverse order.

## Timing diagrams for error-free muting procedure with 4 muting sensors

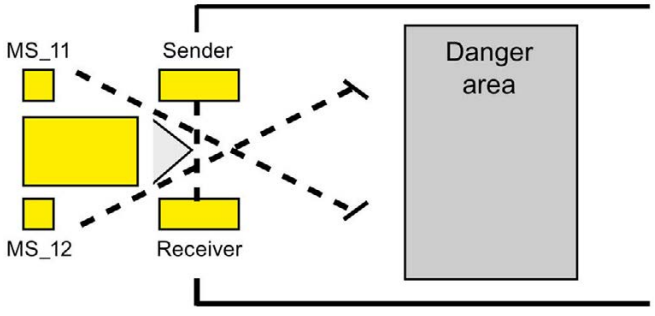



## Schematic sequence of muting procedure with reflection light barriers

If reflection light barriers are used as muting sensors, they are generally arranged diagonally.

In general, this arrangement of reflection light barriers as muting sensors requires only two light barriers, and only MS_11 and MS_12 are interconnected.

The sequence is similar to that of the muting procedure with 4 muting sensors. Step 3 is omitted. In step 4, replace MS_21 and MS_22 with MS_11 and MS_12, respectively.



### Restart inhibit upon interruption of light curtain (if MUTING is not active), when errors occur, and during F-system startup

Enable signal Q cannot be set to 1 or becomes 0, if:

- Light curtain is interrupted (e.g., by a person or material transport) while the MUTING function is not active
- The muting lamp monitoring function responds at input QBAD_MUT
- Sensor pair 1 (MS_11 and MS_12) or sensor pair 2 (MS_21 and MS_22) is not activated or deactivated during discrepancy time DISCTIM1 or DISCTIM2, respectively
- The MUTING function is active longer than the maximum muting time TIME_MAX
- Discrepancy times DISCTIM1 and DISCTIM2 have been set to values < 0 or > 3 s
- Maximum muting time TIME_MAX has been set to a value< 0 or > 10 min

In the identified cases, output FAULT (group error) is set to 1 (restart inhibit). If the MUTING function is started, it will be terminated and the Muting output becomes 0.

**⚠ WARNING**

When a valid combination of muting sensors is immediately detected at startup of the F-system (for example, because the muting sensors are interconnected to inputs of a standard I/O that immediately provide process values during the F-system startup), the MUTING function is immediately started and the MUTING output and enable signal Q are set to 1. The FAULT output (group error) is not set to 1 (no restart inhibit!). *(S035)*

### Acknowledgment of restart inhibit

Enable signal Q becomes 1 again, when:

- The light curtain is no longer interrupted
- Errors, if present, are eliminated (see output DIAG)

  and

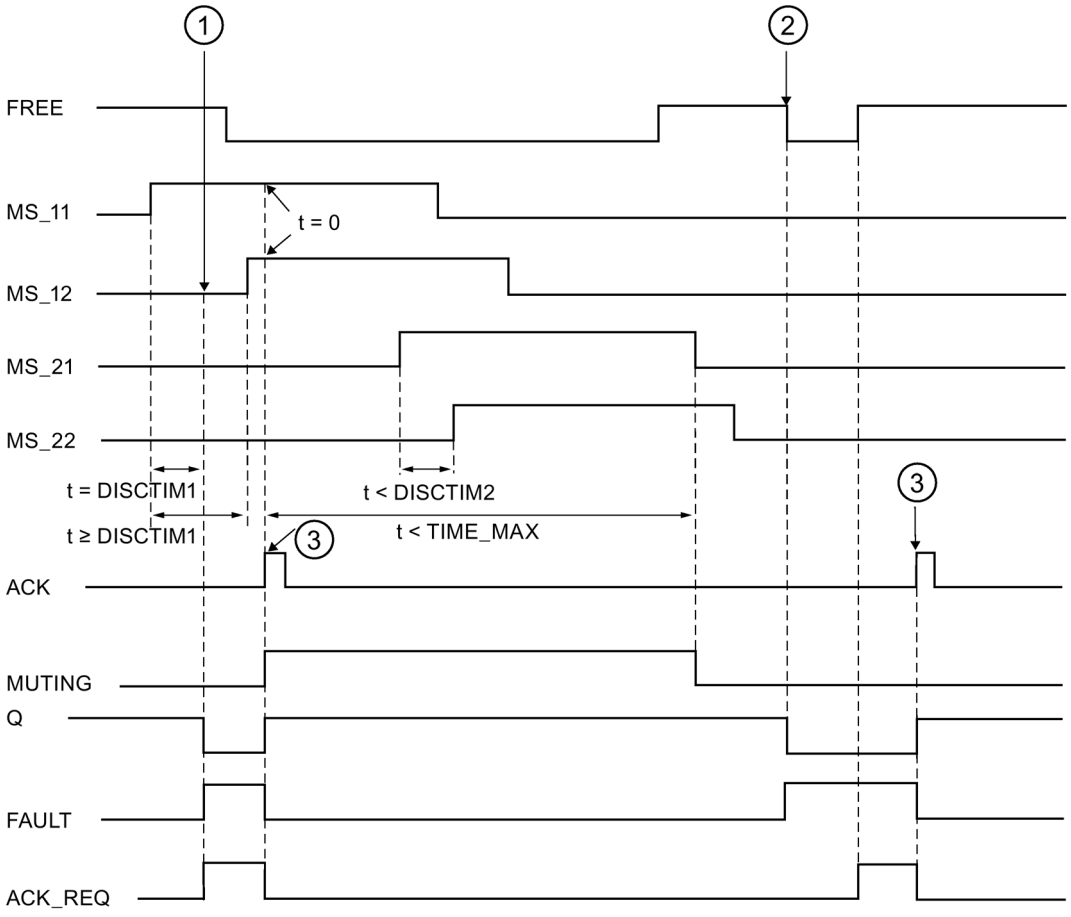
- A user acknowledgment with positive edge occurs at input ACK (see also Implementation of user acknowledgment (Page 151)).

The FAULT output is set to 0. Output ACK_REQ = 1 signals that user acknowledgment at input ACK is required to eliminate the restart inhibit. The instruction sets ACK-REQ = 1 as soon as the light curtain is no longer interrupted or errors have been eliminated. Once acknowledgment has occurred, the instruction resets ACK_REQ to 0.

**Note**

Following discrepancy errors and once the maximum muting time has been exceeded, ACK_REQ is immediately set to 1. As soon as a user acknowledgment has taken place at input ACK, discrepancy times DISCTIM1 and DISCTIM2 and maximum muting time TIME_MAX are reset.

### Timing diagrams for discrepancy errors at sensor pair 1 or interruption of the light curtain (if MUTING is not active)




① Sensor pair 1 (MS_11 and MS_12) is not activated within discrepancy time DISCTIM1.

② The light curtain is interrupted even though the MUTING function is not active.

③ Acknowledgment

### Behavior with stopped conveyor equipment

If, while the conveyor equipment has stopped, the monitoring for one of the following reasons is to be deactivated:

- To comply with discrepancy time DISCTIM1 or DISCTIM2
- To comply with maximum muting time TIME_MAX

you must supply input STOP with a "1" signal for as long as the conveyor equipment is stopped. As soon as the conveyor equipment is running again (STOP = 0), discrepancy times DISCTIM1 and DISCTIM2 and maximum muting time TIME_MAX are reset.

**WARNING**

When STOP = 1, the discrepancy monitoring is shut down. During this time, if inputs MSx1/MSx2 of a sensor pair both take a signal state of 1 due to an undetected error, e.g., because both muting sensors fail to 1, the error is not detected and the MUTING function can be started unintentionally. *(S036)*

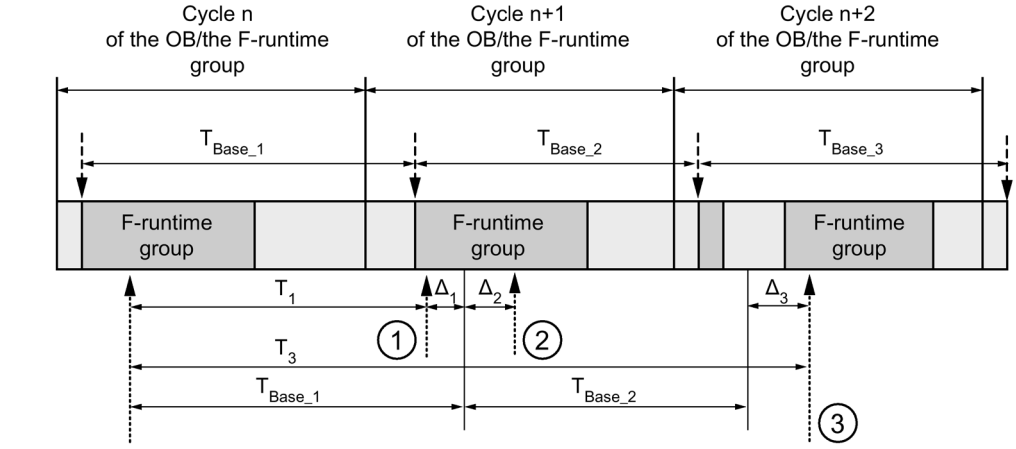
### Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Discrepancy error or incorrect discrepancy time DISCTIM 1 setting for sensor pair 1 | Malfunction in production sequence | Malfunction in production sequence eliminated |
| | | Sensor defective | Check sensors |
| | | Wiring fault | Check wiring of sensors |
| | | Sensors are wired to different F-I/O, and F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON on an F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Discrepancy time setting is too low | If necessary, set a higher discrepancy time |
| | | Discrepancy time setting is < 0 s or > 3 s | Set discrepancy time in range between 0 s and 3 s |
| Bit 1 | Discrepancy error or incorrect discrepancy time DISCTIM 2 setting for sensor pair 2 | Same as Bit 0 | Same as Bit 0 |
| Bit 2 | Maximum muting time exceeded or incorrect muting time TIME_MAX setting | Malfunction in production sequence | Malfunction in production sequence eliminated |
| | | Maximum muting time setting is too low | If necessary, set a higher maximum muting time |
| | | Muting time setting is < 0 s or > 10 min | Set muting time in range from 0 s to 10 min |
| Bit 3 | Light curtain interrupted and muting not active | Light curtain is defective | Check light curtain |
| | | Wiring fault | Check wiring of light curtain (FREE input) |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of light curtain (FREE input) | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | See other DIAG bits | |
| Bit 4 | Muting lamp is defective or cannot be set | Muting lamp is defective | Replace muting lamp |
| | | Wiring fault | Check wiring of muting lamp |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of muting lamp | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 5 | Reserved | — | — |
| Bit 6 | Reserved | — | — |
| Bit 7 | Reserved | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




----► = Time base update

·····► = Call time of an instruction with time processing

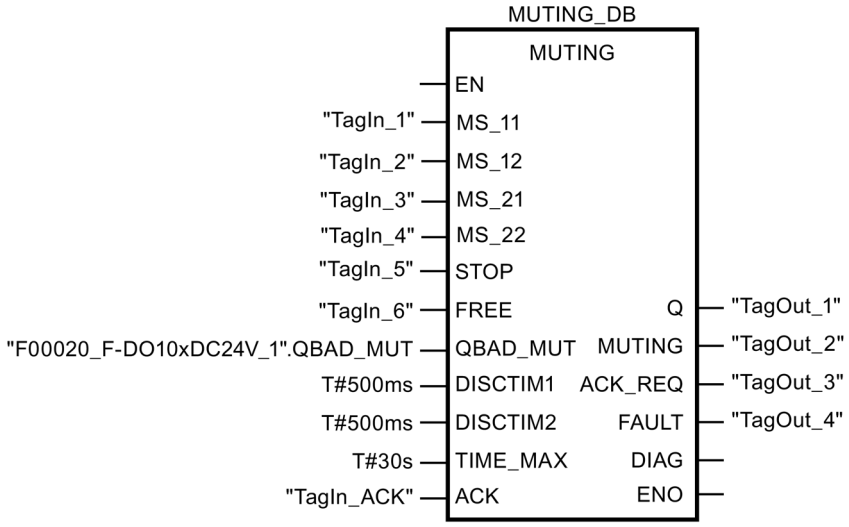
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

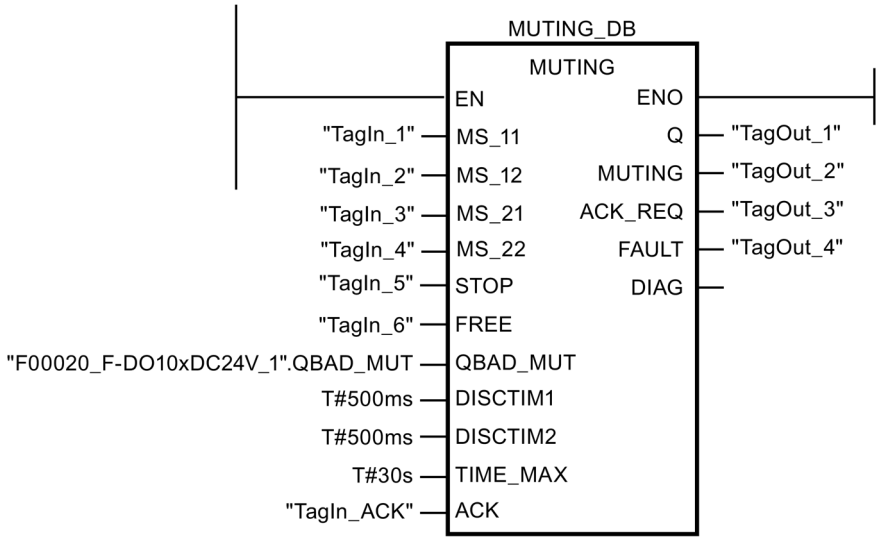
② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.3.3.5 MUT_P: Parallel muting (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction performs parallel muting with two or four muting sensors.

Muting is a defined suppression of the protective function of light curtains. Light curtain muting can be used to introduce goods or objects into the danger area monitored by the light curtain without causing the machine to stop.

To utilize the muting function, at least two independently wired muting sensors must be present. The use of two or four muting sensors and correct integration into the production sequence must ensure that no persons enter the danger area while the light curtain is muted.

Every call of the "Parallel muting" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., MUT_P_DB_1) or a multi-instance (e.g., MUT_P_Instance_1) for the "Parallel muting" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| MS_11 | Input | BOOL | Muting sensor 11 |
| MS_12 | Input | BOOL | Muting sensor 12 |
| MS_21 | Input | BOOL | Muting sensor 21 |
| MS_22 | Input | BOOL | Muting sensor 22 |
| STOP | Input | BOOL | 1=Conveyor system stopped |
| FREE | Input | BOOL | 1=Light curtain uninterrupted |
| ENABLE | Input | BOOL | 1=Enable MUTING |
| QBAD_MUT | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal / inverted value status of the muting lamp channel |
| ACK | Input | BOOL | Acknowledgment of restart inhibit |
| DISCTIM1 | Input | TIME | Discrepancy time of sensor pair 1 (0 to 3 s) |
| DISCTIM2 | Input | TIME | Discrepancy time of sensor pair 2 (0 to 3 s) |
| TIME_MAX | Input | TIME | Maximum muting time (0 to 10 min) |
| Q | Output | BOOL | 1= Enable, not off |
| MUTING | Output | BOOL | Display of muting is active |
| ACK_REQ | Output | BOOL | Acknowledgment necessary |
| FAULT | Output | BOOL | Group error |
| DIAG | Output | WORD | Service information |

## Instruction versions

A number of versions are available for this instruction:

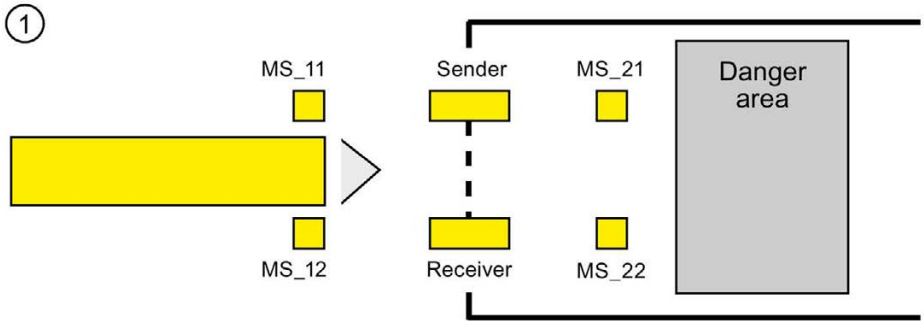
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x* | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x* | — | — | These versions have identical functions to version V1.0.<br>The output DIAG can now be correctly interconnected with the operand of data type WORD. |
| 1.2 | x* | — | x | |
| 1.3 | x* | x | x | |

* S7-300/400: When a restart inhibit (output FAULT = 1) and ENABLE = 1 is present, output ACK_REQ is set to 1 even if not at least one muting sensor is activated. Use the DIAG bits 5 and 6 for additional information.

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

### Schematic sequence of error-free muting procedure with 4 muting sensors (MS_11, MS_12, MS_21, MS_22)

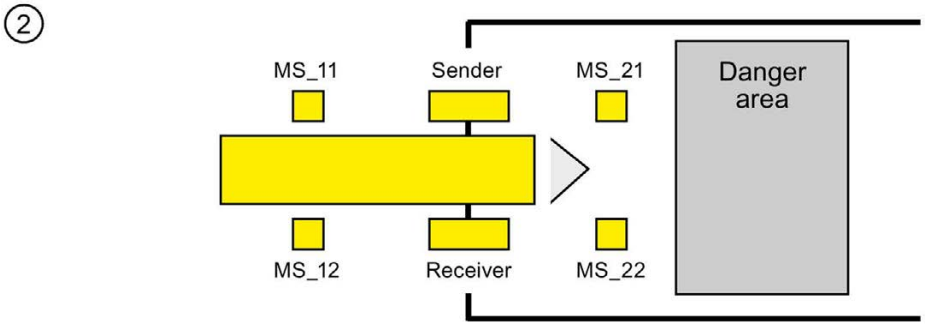



- If muting sensors MS_11 and MS_12 are both activated by the product within DISCTIM1 (apply signal state = 1) and MUTING is enabled by setting the ENABLE input to 1, the instruction starts the MUTING function. Enable signal Q remains 1, even when input FREE = 0 (light curtain interrupted by product). The MUTING output for setting the muting lamp switches to 1.

**Note**

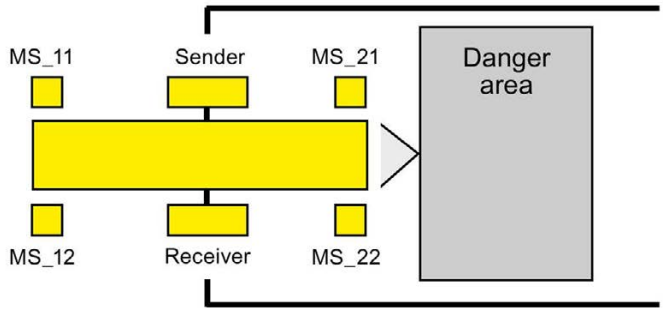
The muting lamp can be monitored using the QBAD_MUT input. To do this, you must wire the muting lamp to an output with wire break monitoring of an F-I/O and supply the QBAD_MUT input with the QBAD signal of the associated F-I/O or the QBAD_O_xx signal / with inverted value statues of the associated channel. If QBAD_MUT = 1, muting is terminated by the instruction. If monitoring of the muting lamp is not necessary, you do not have to supply input QBAD_MUT.

F-I/O that can promptly detect a wire break after activation of the muting operation must be used (*see manual for specific F-I/O*).



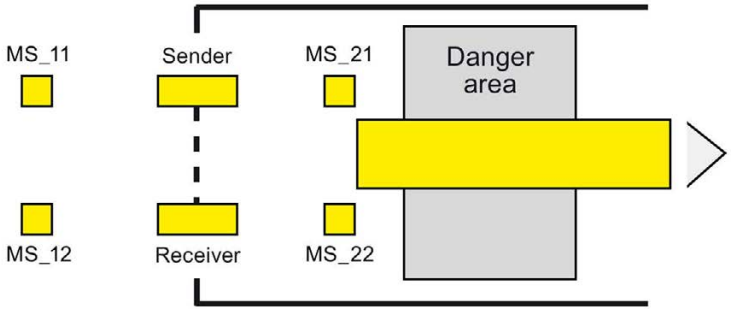

- As long as both muting sensors MS_11 and MS_12 continue to be activated, the MUTING function of the instruction causes Q to remain 1 and MUTING to remain 1 (so that the product can pass through the light curtain without causing the machine to stop). Each of the two muting sensors MS_11 and MS_12 may be switched to inactive (t < DISCTIM1) for a short time (apply signal state 0).

③




- Muting sensors MS_21 and MS_22 must both be activated (within DISCTIM2) before muting sensors MS_11 and MS_12 are switched to inactive (apply signal state 0). In this way, the instruction retains the MUTING function. (Q = 1, MUTING = 1).

④

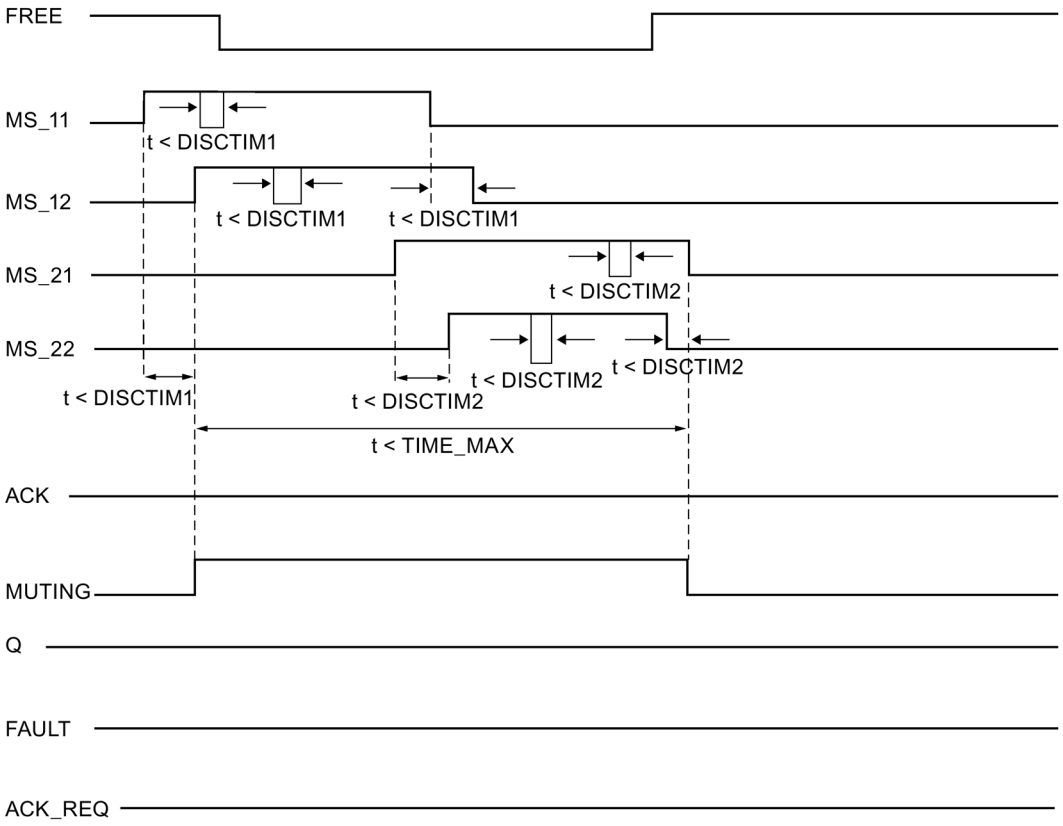



Only if muting sensors MS_21 and MS_22 are both switched to inactive (product enables sensors) is the MUTING function terminated (Q = 1, MUTING = 0). The maximum activation time for the MUTING function is the time set at input TIME_MAX.

**Note**

The MUTING function is also started if the product passes the light curtain in the reverse direction and the muting sensors are thus activated by the product in reverse order.

## Timing diagrams for error-free muting procedure with 4 muting sensors

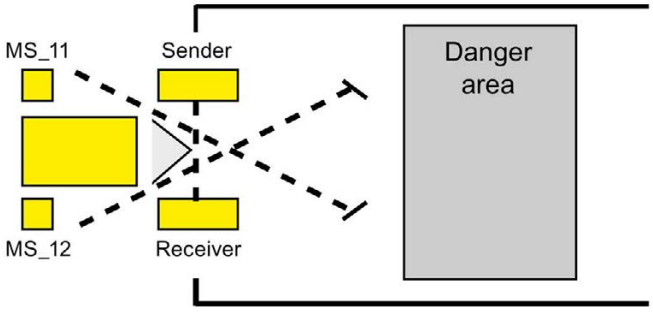



## Schematic sequence of muting procedure with reflection light barriers

If reflection light barriers are used as muting sensors, they are generally arranged diagonally.

In general, this arrangement of reflection light barriers as muting sensors requires only two light barriers, and only MS_11 and MS_12 are interconnected.

The sequence is similar to that of the muting procedure with 4 muting sensors. Step 3 is omitted. In step 4, replace MS_21 and MS_22 with MS_11 and MS_12, respectively.



### Restart inhibit upon interruption of light curtain (MUTING is not active), as well as when errors occur and during F-system startup

Enable signal Q cannot be set to 1 or becomes 0, if:

- Light curtain is interrupted (e.g., by a person or material transport) while the MUTING function is not active
- Light curtain is (being) interrupted and the muting lamp monitoring at input QBAD_MUT is set to 1
- Light curtain is (being) interrupted and the MUTING function is not enabled by setting input ENABLE to 1
- Sensor pair 1 (MS_11 and MS_12) or sensor pair 2 (MS_21 and MS_22) is not activated or deactivated during discrepancy time DISCTIM1 or DISCTIM2, respectively
- The MUTING function is active longer than the maximum muting time TIME_MAX
- Discrepancy times DISCTIM1 and DISCTIM2 have been set to values < 0 or > 3 s
- Maximum muting time TIME_MAX has been set to a value< 0 or > 10 min
- The F-system starts up (regardless of whether or not the light curtain is interrupted, because the F-I/O is passivated after F-system startup and, thus, the FREE input is initially supplied with 0)

In the identified cases, output FAULT (group error) is set to 1 (restart inhibit). If the MUTING function is started, it will be terminated and the Muting output becomes 0.

### User acknowledgment of restart inhibit (no muting sensor is activated or ENABLE = 0)

Enable signal Q becomes 1 again, when:

- The light curtain is no longer interrupted
- Errors, if present, are eliminated (see output DIAG)

  and

- A user acknowledgment with positive edge occurs at input ACK (see also Implementation of user acknowledgment (Page 151)).

The FAULT output is set to 0. Output ACK_REQ = 1 (and DIAG bit 6) signals that user acknowledgment at input ACK is required to eliminate the restart inhibit. The instruction sets ACK_REQ = 1 as soon as the light curtain is no longer interrupted or the errors have been eliminated. Once acknowledgment has occurred, the instruction resets ACK_REQ to 0.

### User Acknowledgment of restart inhibit (at least one muting sensor is activated and ENABLE = 1)

Enable signal Q becomes 1 again, when:

- Errors, if present, are eliminated (see output DIAG)
- FREE occurs until a valid combination of muting sensors is detected

The FAULT output is set to 0. The MUTING function is restarted, if necessary, and the MUTING output becomes 1 if a valid combination of muting sensors is detected. When ENABLE = 1, output ACK_REQ = 1 (and DIAG bit 5) signals that FREE is necessary for error elimination and for removal of the restart inhibit.*After successful FREE, ACK_REQ is reset to 0 by the instruction.

**Note**

Once the maximum muting time is exceeded, TIME_MAX is reset as soon as the MUTING function is restarted.

### FREE function

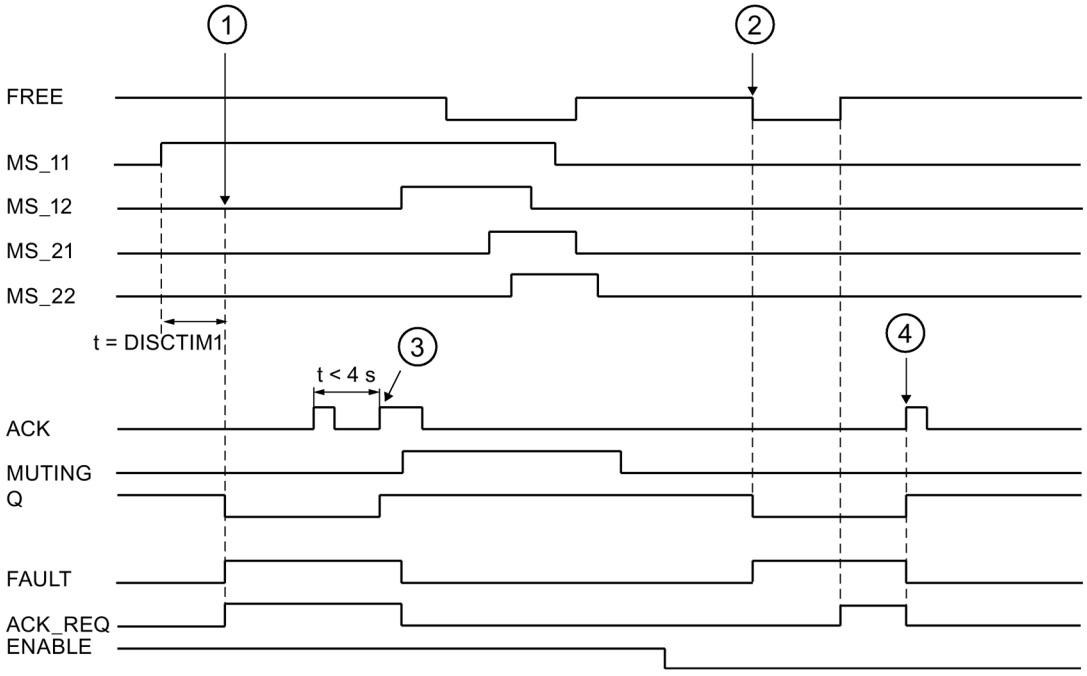
If an error cannot be corrected immediately, the FREE function can be used to free the muting range. Enable signal Q and output MUTING =1 temporarily. The FREE function can be used if:

- ENABLE = 1
- At least one muting sensor is activated
- A user acknowledgment with rising edge at input ACK occurs twice within 4 s, and the second user acknowledgment at input ACK remains at a signal state of 1 (acknowledgment button remains activated)

**⚠ WARNING**

When using the FREE function, the action must be observed. A dangerous situation must be able to be interrupted at any time by releasing the acknowledgment button. The acknowledgment button must be mounted in such a way the entire danger area can be observed. *(S037)*

### Timing diagrams for discrepancy errors at sensor pair 1 or interruption of the light curtain (MUTING is not active)




① Sensor pair 1 (MS_11 and MS_22) is not activated within discrepancy time DISCTIM1.

② The light curtain is interrupted even though there is no enable (ENABLE=0)

③ FREE function

④ Acknowledgment

### Behavior with stopped conveyor equipment

If, while the conveyor equipment has stopped, the monitoring for one of the following reasons is to be deactivated:

- To comply with discrepancy time DISCTIM1 or DISCTIM2
- To comply with maximum muting time TIME_MAX

You must supply input STOP with a "1" signal for as long as the conveyor equipment is stopped. As soon as the conveyor equipment is running again (STOP = 0), discrepancy times DISCTIM1 and DISCTIM2 and maximum muting time TIME_MAX are reset.

**⚠ WARNING**

When STOP = 1 or ENABLE = 0, discrepancy monitoring is shut down. During this time, if inputs MSx1/MSx2 of a sensor pair both take a signal state of 1 due to an undetected error, e.g., because both muting sensors fail to 1, the fault is not detected and the MUTING function can be started unintentionally (when ENABLE =1). *(S038)*

## Output DIAG

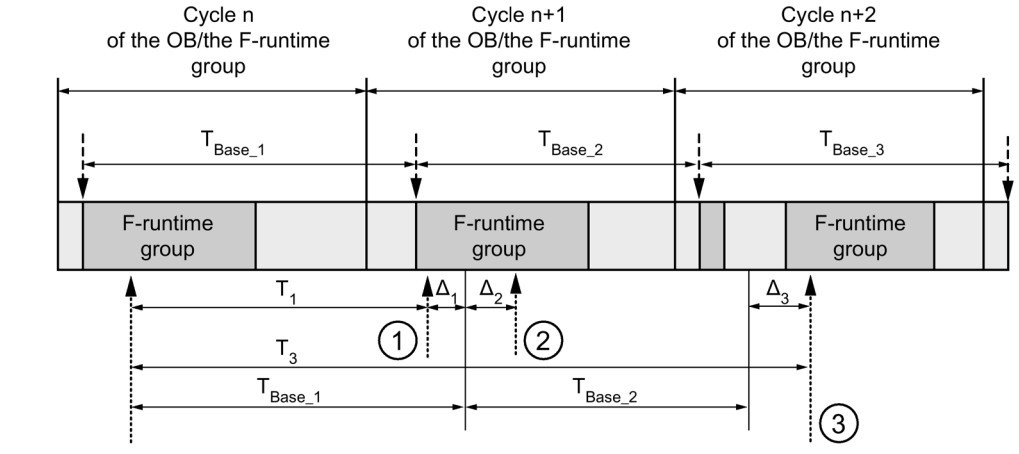
The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits 0 to 6 are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Discrepancy error or incorrect discrepancy time DISCTIM 1 setting for sensor pair 1 | Malfunction in production sequence | Malfunction in production sequence eliminated |
| | | Sensor defective | Check sensors |
| | | Wiring fault | Check wiring of sensors |
| | | Sensors are wired to different F-I/O, and F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON on an F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Discrepancy time setting is too low | If necessary, set a higher discrepancy time |
| | | Discrepancy time setting is < 0 s or > 3 s | Set discrepancy time in range between 0 s and 3 s |
| Bit 1 | Discrepancy error or incorrect discrepancy time DISCTIM 2 setting for sensor pair 2 | Same as Bit 0 | Same as Bit 0 |
| Bit 2 | Maximum muting time exceeded or incorrect muting time TIME_MAX setting | Malfunction in production sequence | Malfunction in production sequence eliminated |
| | | Maximum muting time setting is too low | If necessary, set a higher maximum muting time |
| | | Muting time setting is < 0 s or > 10 min | Set muting time in range from 0 s to 10 min |
| Bit 3 | Light curtain interrupted and muting not active | ENABLE = 0 | Set ENABLE = 1 |
| | | Light curtain is defective | Check light curtain |
| | | Wiring fault | Check wiring of light curtain (FREE input) |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of light curtain<br>(FREE input) | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Startup of F-system | For FREE, see DIAG Bit 5 |
| | | See other DIAG bits | |
| Bit 4 | Muting lamp is defective or cannot be set | Muting lamp is defective | Replace muting lamp |
| | | Wiring fault | Check wiring of muting lamp |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of muting lamp | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 5 | FREE is necessary | See other DIAG bits | Two rising edges at ACK within 4 s, and activate acknowledgment button until ACK_REQ = 0 |
| Bit 6 | Acknowledgment necessary | — | — |
| Bit 7 | State of output Q | — | — |
| Bit 8 | State of output MUTING | — | — |
| Bit 9 | FREE active | — | — |
| Bit 10 | Reserved | — | — |
| ... | | | |
| Bit 15 | Reserved | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:



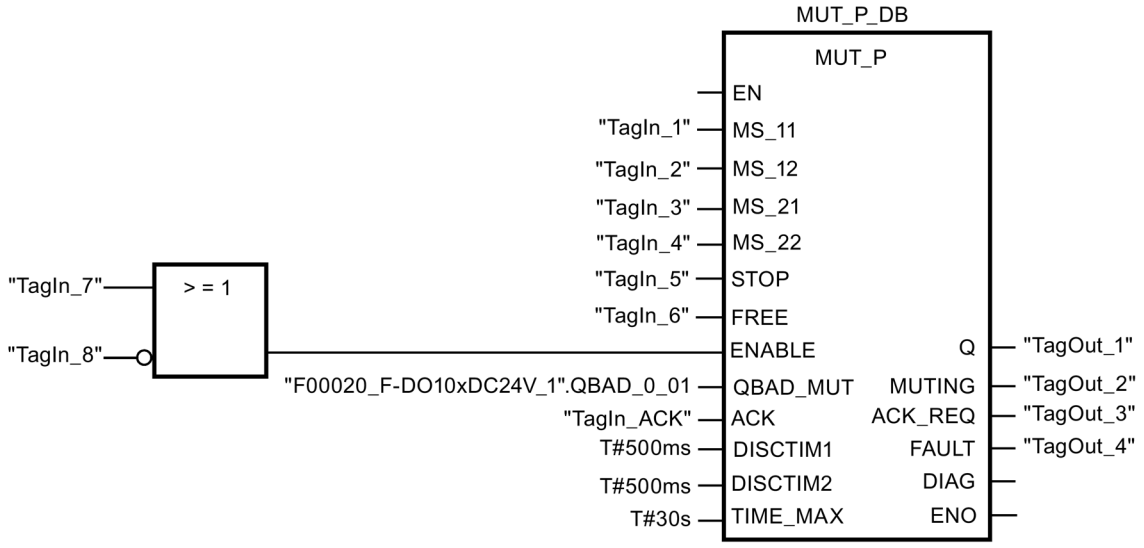

① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

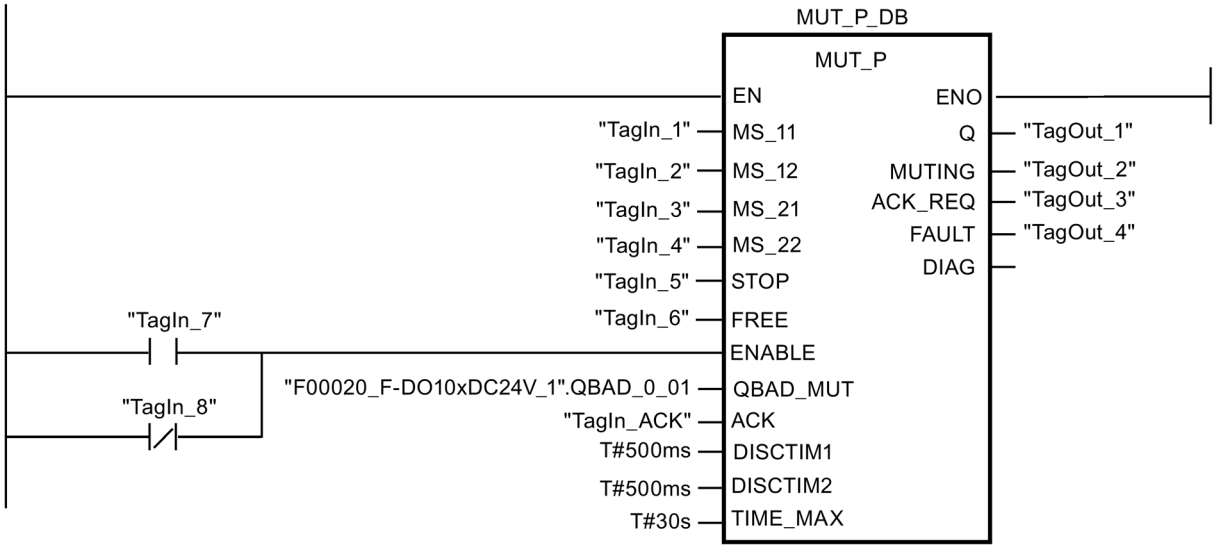
② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.3.3.6 EV1oo2DI: 1oo2 evaluation with discrepancy analysis (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction implements a 1oo2 evaluation of two single-channel sensors combined with a discrepancy analysis.

Output Q is set to 1, if the signal states of inputs IN1 and IN2 both equal 1 and no discrepancy error DISC_FLT is stored. if the signal state of one or both inputs is 0, output Q is set to 0.

As soon as the signal states of inputs IN1 and IN2 are different, the discrepancy time DISCTIME is started. If the signal states of the two inputs are still different once the discrepancy time expires, a discrepancy error is detected and DISC_FLT is set to 1 (restart inhibit).

If the discrepancy between inputs IN1 and IN2 is no longer detected, the discrepancy error is acknowledged according to the parameter assignment of ACK_NEC:

- If ACK_NEC = 0, the acknowledgment is automatic.
- If ACK_NEC = 1, you must use a rising edge at input ACK to acknowledge the discrepancy error.

The output ACK_REQ = 1 signals that a user acknowledgment at input ACK is necessary to acknowledge the discrepancy error (cancel the restart inhibit). The instruction sets ACK_REQ = 1 as soon as discrepancy is no longer detected. After acknowledgment or if, prior to acknowledgment, there is once again a discrepancy between inputs IN1 and IN2, the instruction resets ACK_REQ to 0.

Output Q can never be set to 1 if the discrepancy time setting is < 0 or > 60 s. In this case, output DISC_FLT is also set to 1 (restart inhibit). The call interval of the safety program (e.g., OB35) must be less than the discrepancy time setting.

Every call of the "1oo2 evaluation with discrepancy analysis" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., EV1oo2DI_DB_1) or a multi-instance (e.g., EV1oo2DI_Instance_1) for the "1oo2 evaluation with discrepancy analysis" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

| ⚠ WARNING |
| --- |
| The ACK_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S033)* |

**WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | BOOL | Sensor 1 |
| IN2 | Input | BOOL | Sensor 2 |
| DISCTIME | Input | TIME | Discrepancy time (0 to 60 s) |
| ACK_NEC | Input | BOOL | 1 = acknowledgment necessary for discrepancy error |
| ACK | Input | BOOL | Acknowledgment of discrepancy error |
| Q | Output | BOOL | Output |
| ACK_REQ | Output | BOOL | 1 = acknowledgment required |
| DISC_FLT | Output | BOOL | 1 = discrepancy error |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

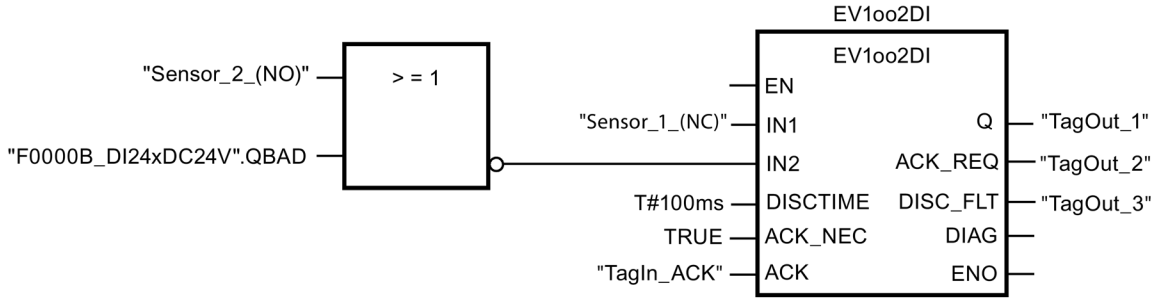
## Activating inputs IN1 and IN2

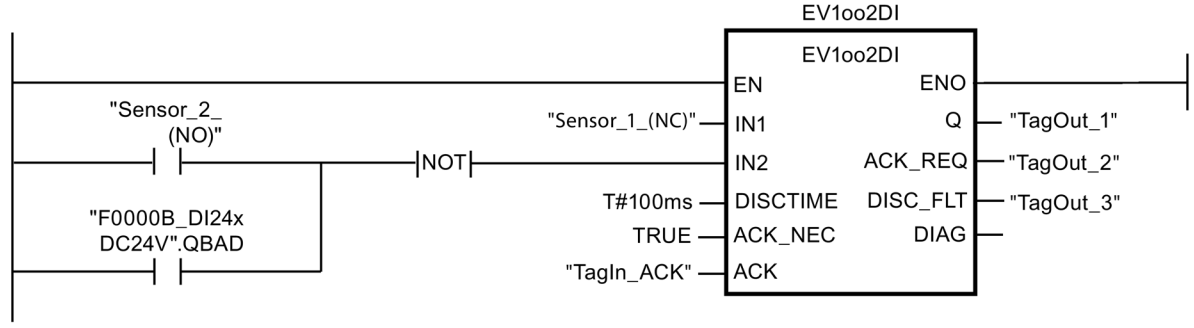
Inputs IN1 and IN2 must both be activated in such a way that their safe state is 0.

## Example with QBAD or QBAD_I_xx signal

For non-equivalent signals you need to OR the input (IN1 and IN2) with which you assign the encoder signal to the safe state 1, with the QBAD signal of the associated F-I/O or the QBAD_I_xx signal of the associated channel (with S7-300/400 F-CPUs) and negate the result. Signal state 0 is then at input IN1 or IN2 when fail-safe values are output.

Network1: EV1oo2DI with non-equivalent signals

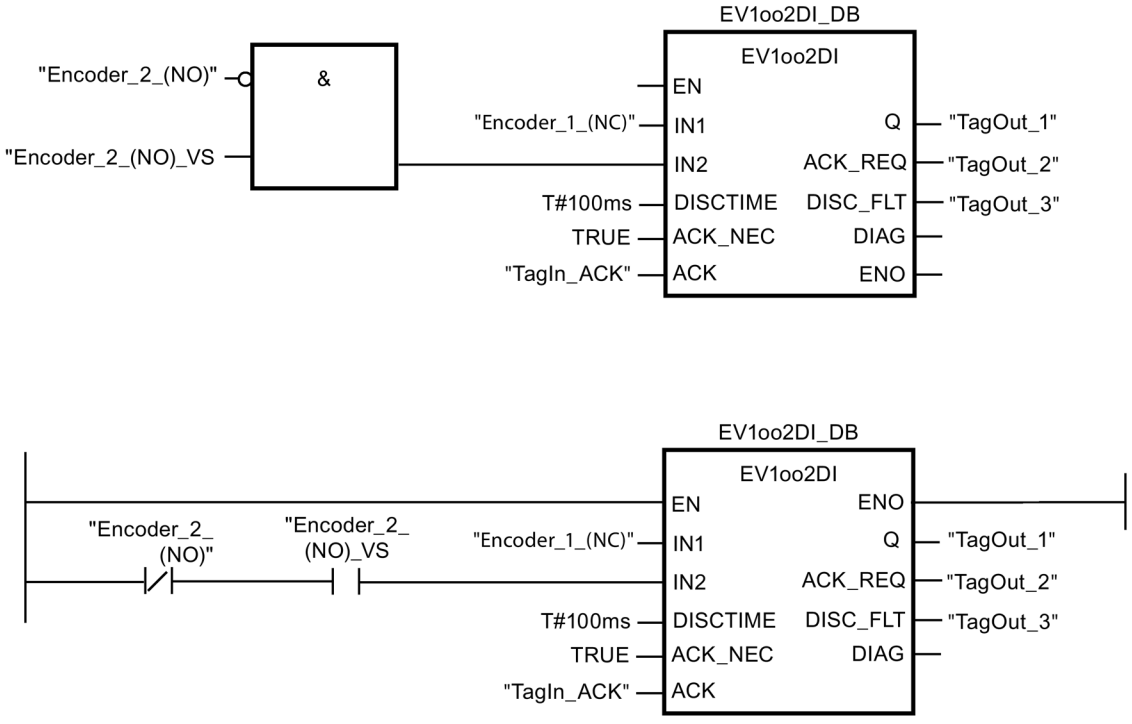


## Example with value status

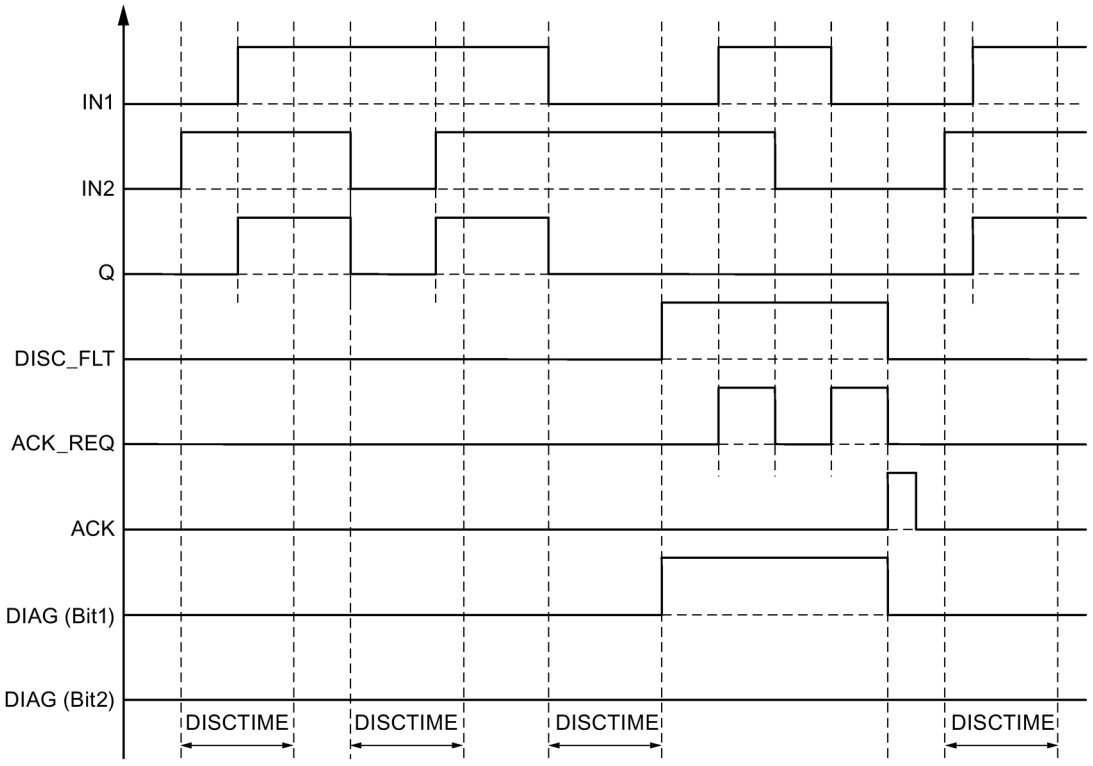
For nonequivalent signals, you have to negate the input (IN1 or IN2) with which you have assigned the encoder signal to a safe state of 1 and AND it with the value status of the associated channel. Signal state 0 is then at input IN1 or IN2 when fail-safe values are output.

Network1: EV1oo2DI with non-equivalent signals



## Timing diagrams EV1oo2DI

If ACK_NEC = 1:




## Startup characteristics

**Note**

If the sensors at inputs IN1 and IN2 are assigned to different F-I/O, it is possible that the fail-safe values are output for different lengths of time following startup of the F-system due to different startup characteristics of the F-I/O. If the signal states of inputs IN1 and IN2 remain different after the discrepancy time DISCTIME has expired, a discrepancy error is detected after the F-system starts up.

If ACK_NEC = 1 you must acknowledge the discrepancy error with a rising edge at input ACK.

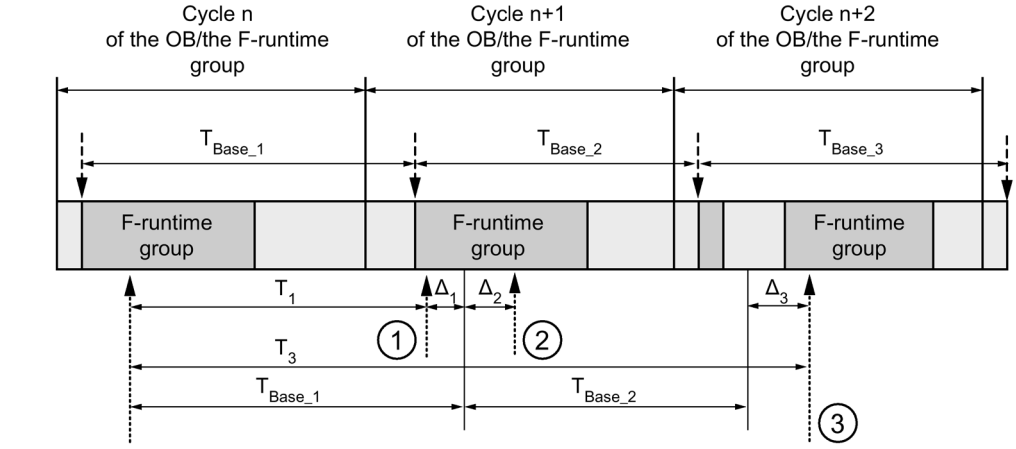
## Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Discrepancy error or incorrect discrepancy time setting (= status of DISC_FLT) | Sensor defective | Check sensors |
| | | Wiring fault | Check wiring of sensors |
| | | Sensors are wired to different F-I/O, and F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON on an F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| | | Discrepancy time setting is too low | If necessary, set a higher discrepancy time |
| | | Discrepancy time setting is < 0 s or > 60 s | Set discrepancy time in range between 0 s and 60 s |
| Bit 1 | For discrepancy errors: last signal state change was at input IN1 | — | — |
| Bit 2 | For discrepancy errors: last signal state change was at input IN2 | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Reserved | — | — |
| Bit 5 | For discrepancy errors: input ACK has a permanent signal state of 1 | Acknowledgment button defective | Replace acknowledgment button |
| | | Wiring fault | Check wiring of acknowledgment button |
| Bit 6 | Acknowledgment necessary | — | — |
| Bit 7 | State of output Q | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




----► = Time base update

·····► = Call time of an instruction with time processing

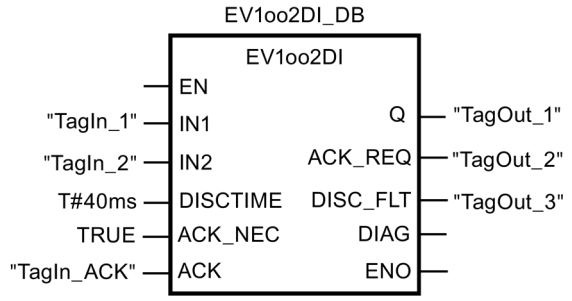
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

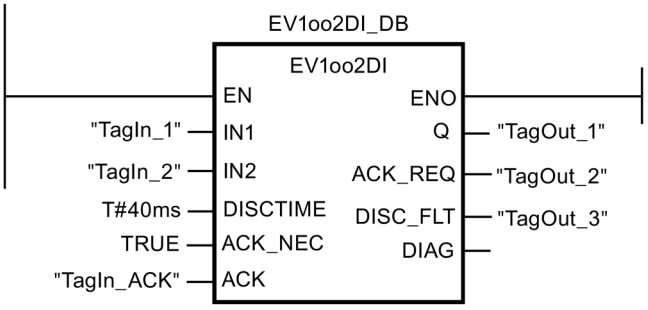
② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:



### 13.3.3.7 FDBACK: Feedback monitoring (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction implements feedback monitoring.

The signal state of output Q is checked to see whether it corresponds to the inverse signal state of the feedback input FEEDBACK.

Output Q is set to 1 as soon as input ON = 1. Requirement for this is that the feedback input FEEDBACK = 1 and no feedback error is saved.

Output Q is reset to 0, as soon as input ON = 0 or if a feedback error is detected.

A feedback error ERROR = 1 is detected if the inverse signal state of the feedback input FEEDBACK (to input Q) does not follow the signal state of output Q within the maximum tolerable feedback time. The feedback error is saved.

If a discrepancy is detected between the feedback input FEEDBACK and the output Q after a feedback error, the feedback error is acknowledged in accordance with the parameter assignment of ACK_NEC:

- If ACK_NEC = 0, the acknowledgment is automatic.
- If ACK_NEC = 1, you must acknowledge the feedback error with a rising edge at input ACK.

The ACK_REQ = 1 output then signals that a user acknowledgment is necessary at input ACK to acknowledge the feedback error. Following an acknowledgment, the instruction resets ACK_REQ to 0.

To avoid a feedback errors from being detected and acknowledgment from being required when the F-I/O controlled by the Q output are passivated, you need to supply input QBAD_FIO with the QBAD signal of the associated F-I/O or the QBAD_O_xx signal / inverted value status of the associated channel.

Every call of the "Feedback monitoring" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., FDBACK_DB_1) or a multi-instance (e.g., FDBACK_Instance_1) for the "Feedback monitoring" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

The ACK_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S033)*

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision *(S034)*.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ON | Input | BOOL | 1= Enable output |
| FEEDBACK | Input | BOOL | Feedback input |
| QBAD_FIO | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal / inverted value status of the Q output |
| ACK_NEC | Input | BOOL | 1=Acknowledgment necessary |
| ACK | Input | BOOL | Acknowledgment |
| FDB_TIME | Input | TIME | Feedback time |
| Q | Output | BOOL | Output |
| ERROR | Output | BOOL | Feedback error |
| ACK_REQ | Output | BOOL | Acknowledgment request |
| DIAG | Output | BYTE | Service information |

## Instruction versions

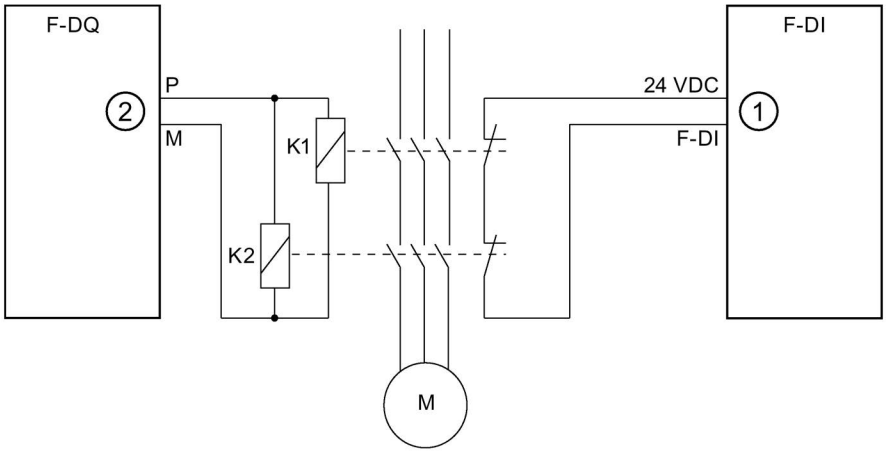
A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | Version 1.0 requires that the F_TOF block with the number FB 186 is available in the project tree in the "Program blocks/System blocks/STEP 7 Safety" folder.<br>When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically. If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. You will then avoid number conflicts. |
| 1.1 | x | — | — | These versions are functionally identical to version V1.0, but do not require the F_TOF block to have a particular number. |
| 1.2 | x | — | x | |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Interconnection example




① Sent to the FEEDBACK input of the instruction
② from output Q of the instruction

## Startup characteristics

After an F-system startup, the instruction does not have be acknowledged when no errors are present.

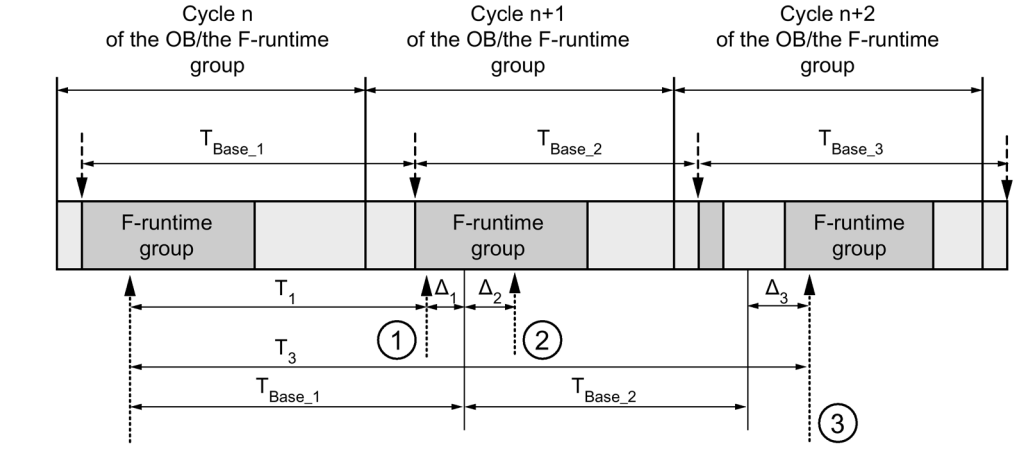
## Output DIAG

The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits 0, 2, and 5 are saved until acknowledgment at input ACK.

## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Feedback error or incorrect feedback time setting (= state of ERROR) | Feedback time setting < 0 | Set feedback time > 0 |
| | | Feedback time setting is too low | If necessary, set a higher feedback time |
| | | Wiring fault | Check wiring of actuator and feedback contact |
| | | Actuator or feedback contact is defective | Check actuator and feedback contact |
| | | I/O fault or channel fault of feedback input | Check I/O |
| Bit 1 | Passivation of F-I/O/channel controlled by output Q (= state of QBAD_FIO) | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 2 | After feedback error: feedback input has permanent signal state of 0 | F-I/O fault or channel fault of feedback input | Check I/O |
| | | Feedback contact is defective | Check feedback contact |
| | | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O of feedback input | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 3 | Reserved | — | — |
| Bit 4 | Reserved | — | — |
| Bit 5 | For feedback error: input ACK has a permanent signal state of 1 | Acknowledgment button defective | Check acknowledgment button |
| | | Wiring fault | Check wiring of acknowledgment button |
| Bit 6 | Acknowledgment required (= state of ACK_REQ) | — | — |
| Bit 7 | State of output Q | — | — |

## Timing imprecision resulting from the update time of the time base used in the instruction:




----► = Time base update

·····► = Call time of an instruction with time processing

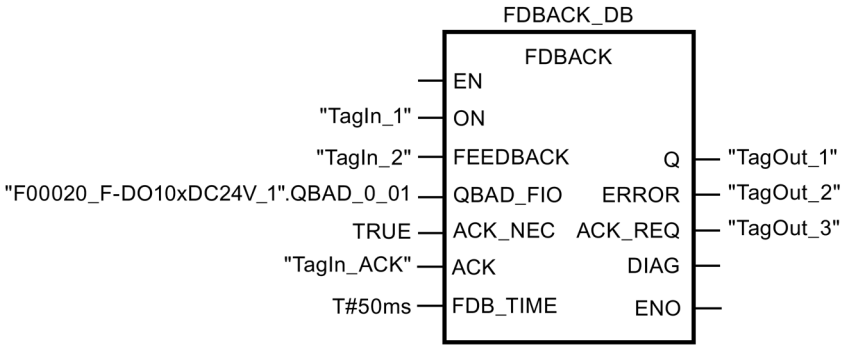
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

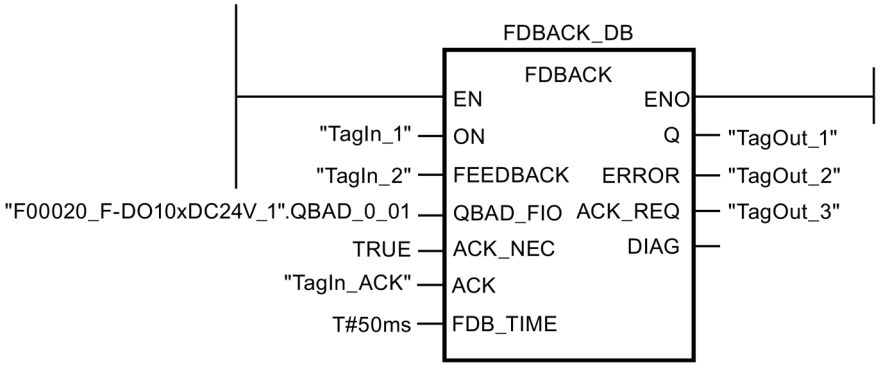
② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

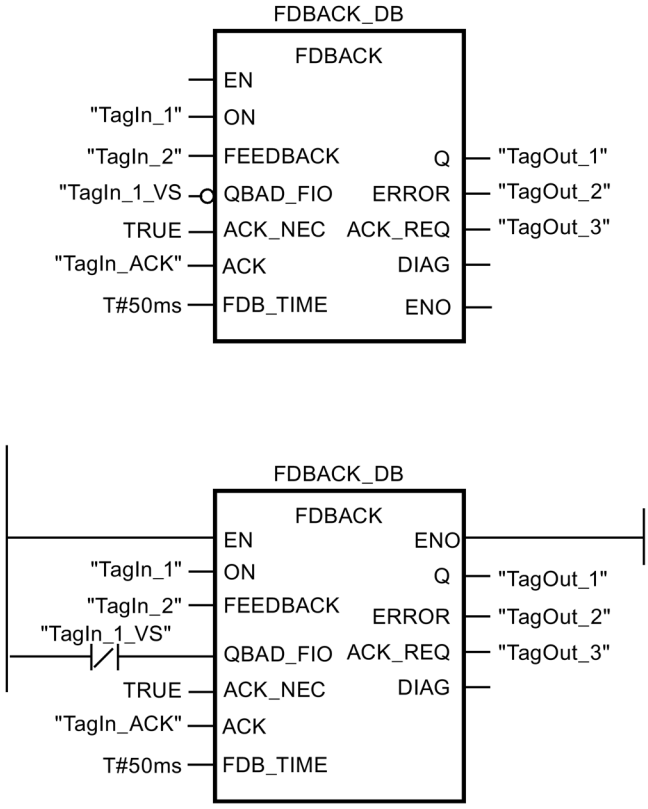
## Example

The following example shows how the instruction for S7-300/400 F-CPUs works:



The following example shows how the instruction for S7-1200/1500 F-CPUs works:



### 13.3.3.8 SFDOOR: Safety door monitoring (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction implements safety door monitoring.

Enable signal Q is reset to 0 as soon as one of the inputs IN1 or IN2 take a signal state of 0 (safety door is opened). The enable signal can be reset to 1, only if:

- Inputs IN1 and IN2 both take a signal state of 0 prior to opening the door (safety door has been completely opened)
- Inputs IN1 and IN2 then both take a signal state of 1 (safety door is closed)
- An acknowledgment occurs

The acknowledgment for the enable takes place according to the parameter assignment at input ACK_NEC:

- If ACK_NEC = 0, the acknowledgment is automatic.
- If ACK_NEC = 1, you must use a rising edge at input ACK for acknowledging the enable.

Output ACK_REQ = 1 is used to signal that a user acknowledgment is required at input ACK for the acknowledgment. The instruction sets ACK_REQ = 1 as soon as the door is closed. Following an acknowledgment, the instruction resets ACK_REQ to 0.

In order for the instruction to recognize whether inputs IN1 and IN2 are 0 merely due to passivation of the associated F-I/O, you need to supply inputs QBAD_IN1 or QBAD_IN2 with the QBAD signal of the associated F-I/O or QBAD_I_xx signal / inverted value status of the associated channel. Among other things, this will prevent you from having to open the safety door completely prior to an acknowledgment in the event the F-I/O are passivated.

Every call of the "Safety door monitoring" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., SFDOOR_DB_1) or a multi-instance (e.g., SFDOOR_Instance_1) for the "Safety door monitoring" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

The ACK_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S033)*

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | BOOL | Input 1 |
| IN2 | Input | BOOL | Input 2 |
| QBAD_IN1 | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal / inverted value status of the channel of input IN1 |
| QBAD_IN2 | Input | BOOL | QBAD signal of the F-I/O or QBAD_O_xx signal / inverted value status of the channel of input IN2 |
| OPEN_NEC | Input | BOOL | 1= Open necessary at startup |
| ACK_NEC | Input | BOOL | 1=Acknowledgment necessary |
| ACK | Input | BOOL | Acknowledgment |
| Q | Output | BOOL | 1= Enable, safety door closed |
| ACK_REQ | Output | BOOL | Acknowledgment request |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

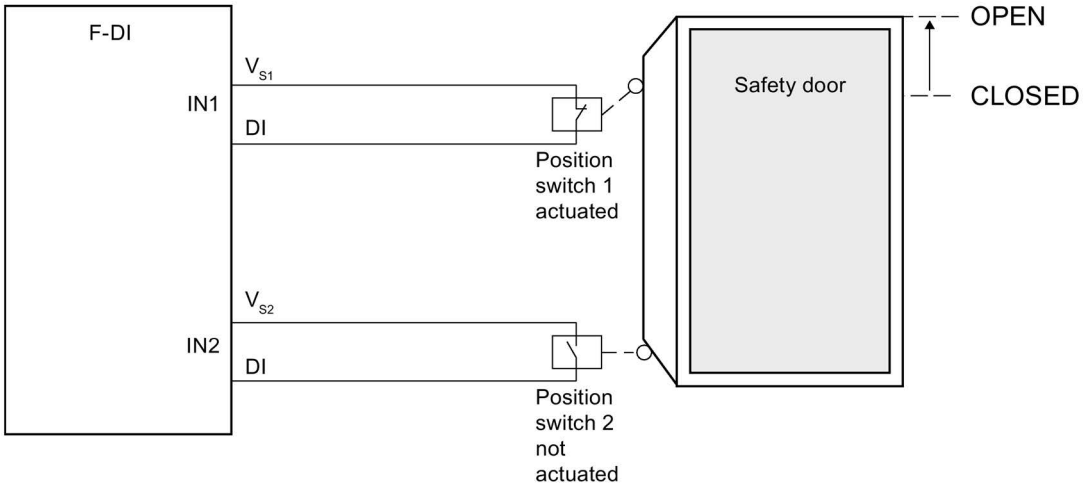
When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

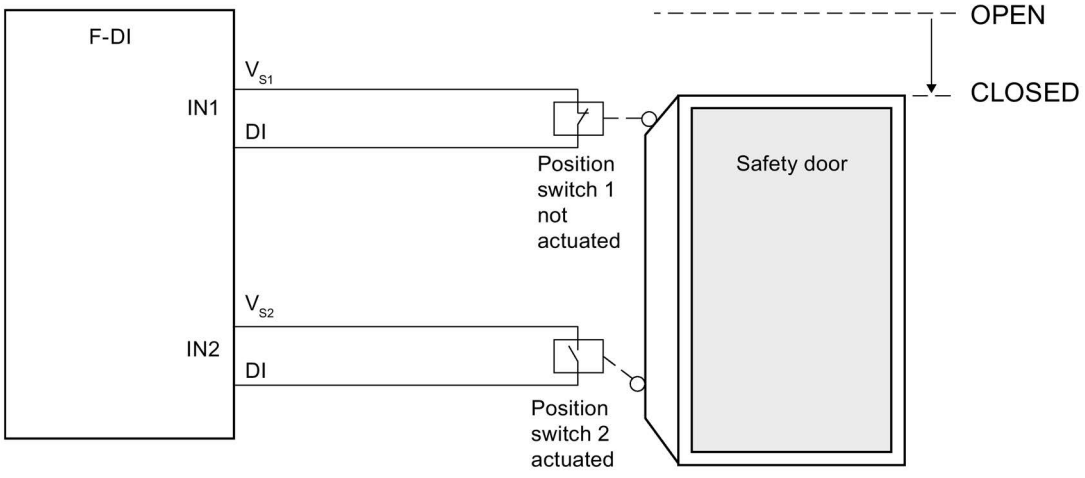
## Interconnection example

You must interconnect the NC contact of position switch 1 of the safety door at input IN1 and the NO contact of position switch 2 at input IN2. Position switch 1 must be mounted in such a way that it is positively operated when the safety door is open. Position switch 2 must be mounted in such a way that it is operated when the safety door is closed.

Safety door open:




Safety door closed:



## Startup characteristics

After an F-system startup, enable signal Q is reset to 0. The acknowledgment for the enable takes place according to the parameter assignment at inputs OPEN_NEC and ACK_NEC:

- When OPEN_NEC = 0, an automatic acknowledgment occurs **independently** of ACK_NEC, as soon as the two inputs IN1 and IN2 take signal state 1 for the first time following reintegration of the associated F-I/O (safety door is closed).
- When OPEN_NEC = 1 **or** if at least one of the IN1 and IN2 inputs still has a signal state of 0 after reintegration of the associated F-I/O, an automatic acknowledgment occurs **according** to ACK_NEC or you have to use a rising edge at input ACK for the enable. Prior to acknowledgment, inputs IN1 and IN2 both have to take a signal state of 0 (safety door has been completely opened) followed by a signal state of 1 (safety door is closed).

**⚠ WARNING**

The OPEN_NEC tag must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded. *(S039)*

## Output DIAG

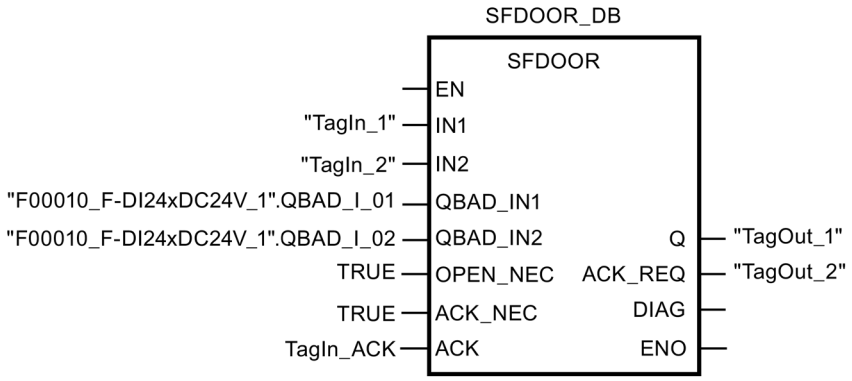
The DIAG output provides non-fail-safe information on errors for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program.

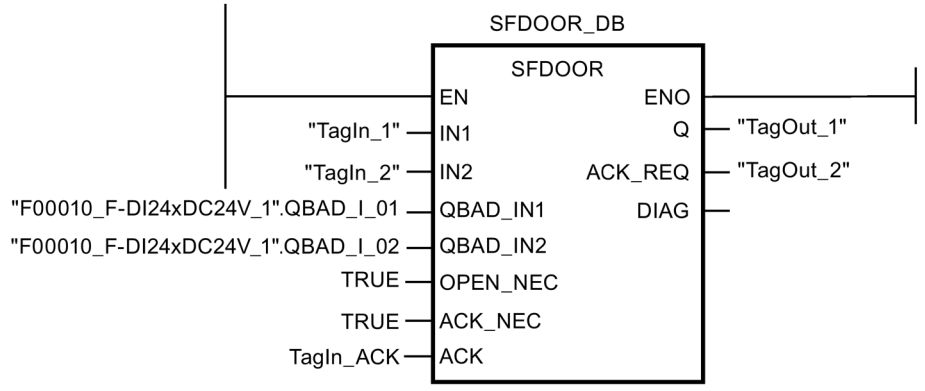
## Structure of DIAG

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Reserved | — | — |
| Bit 1 | Signal state 0 is missing at both IN1 and IN2 inputs | Safety door was not completely opened when OPEN_NEC = 1 after F-system startup | Open safety door completely |
| | | Open safety door was not completely opened | Open safety door completely |
| | | Wiring fault | Check wiring of position switch |
| | | Position switch is defective | Check position switch |
| | | Position switch is incorrectly adjusted | Adjust position switch properly |
| Bit 2 | Signal state 1 is missing at both IN1 and IN2 inputs | Safety door was not closed | Close safety door |
| | | Wiring fault | Check wiring of position switch |
| | | Position switch is defective | Check position switch |
| | | Position switch is incorrectly adjusted | Adjust position switch properly |
| Bit 3 | QBAD_IN1 and/or QBAD_IN2 = 1 | F-I/O fault, channel fault, or communication error, or passivation by means of PASS_ON of F-I/O or channel of IN1 and/or IN2 | For a solution, see the section "Structure of DIAG", bits 0 to 6 in DIAG (Page 137) |
| Bit 4 | Reserved | — | — |
| Bit 5 | If enable is missing: input ACK has a permanent signal state of 1 | Acknowledgment button defective | Check acknowledgment button |
| | | Wiring fault | Check wiring of acknowledgment button |
| Bit 6 | Acknowledgment required (= state of ACK_REQ) | — | — |
| Bit 7 | State of output Q | — | — |

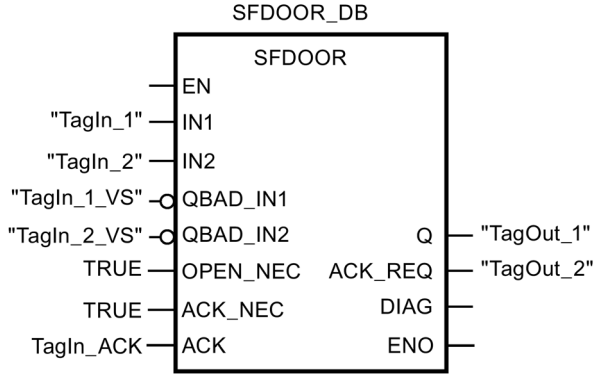
## Example

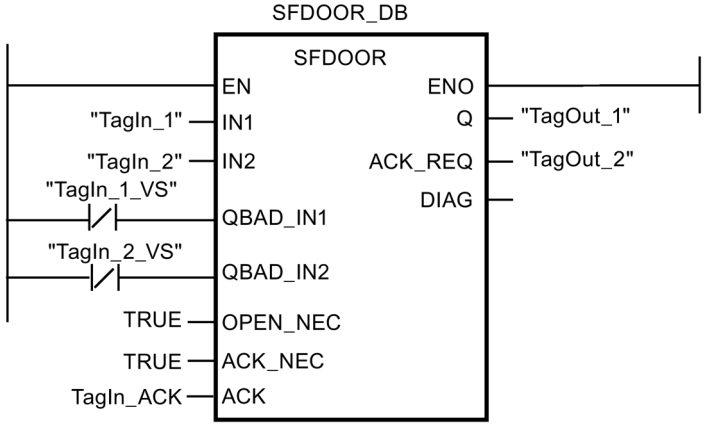
The following example shows how the instruction for S7-300/400 F-CPUs works:



The following example shows how the instruction for S7-1200/1500 F-CPUs works:



### 13.3.3.9 ACK_GL: Global acknowledgment of all F-I/O in an F-runtime group (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction creates an acknowledgment for the simultaneous reintegration of all F-I/O or channels of the F-I/O of an F-runtime group after communication errors, F-I/O errors, or channel faults.

A user acknowledgment (Page 151) with a positive edge at input ACK_GLOB is required for reintegration. The acknowledgment occurs analogously to the user acknowledgment via the ACK_REI tag of the F-I/O DB (Page 137), but it acts simultaneously on all F-I/O of the F-runtime group in which the instruction is called.

If you use the instruction ACK_GL, you do not have to provide a user acknowledgment for each F-I/O of the F-runtime group via the ACK_REI tag of the F-I/O DB.

Every call of the "Global acknowledgment of all F-I/O of a runtime group" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., ACK_GL_DB_1) or a multi-instance (e.g., ACK_GL_Instance_1) for the "Global acknowledgment of all F-I/O of a runtime group" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

An acknowledgment via the ACK_GL instruction is only possible if the tag ACK_REI of the F-I/O DB = 0. Accordingly, an acknowledgment via the tag ACK_REI of the F-I/O DB is only possible if the input ACK_GLOB of the instruction = 0.

The instruction is only allowed to be called once per F-runtime group.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ACK_GLOB | Input | BOOL | 1=acknowledgment for reintegration |

## Instruction versions

A number of versions are available for this instruction:

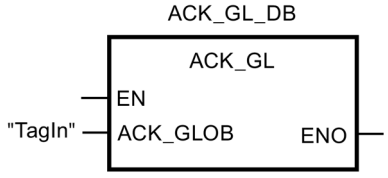
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

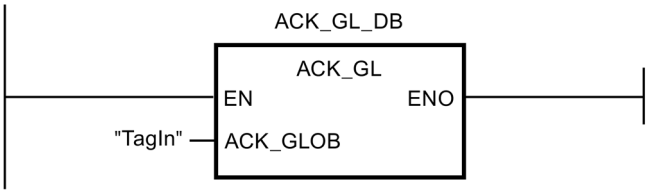
When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:



## 13.3.4 Timer operations

### 13.3.4.1 TP: Generate pulse (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Generate pulse" instruction to set output Q for an assigned period. The instruction is started if the result of logic operation (RLO) changes from "0" to "1" (positive edge) at input IN. The assigned period PT starts running when the instruction starts. Output Q is set for period PT, regardless of the subsequent sequence of the input signal. Also the detection of a new positive signal edge does not influence the signal state at output Q as long as period PT runs.

The current time value can be queried at output ET. The time value begins at T#0s and ends when the value of period PT is reached. If period PT is reached and the signal state at input IN is "0", output ET is reset.

Every call of the "Generate pulse" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Timer_DB_1) or a multi-instance (e.g., F_IEC_Timer_Instance_1) for the "Generate pulse" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction").
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

The operating system resets the instances of the "Generate pulse" instruction on a startup of the F-system.

**Note**

The functionality of this instruction differs from the corresponding standard TP instruction in the following points:

- When a call is made with PT = 0 ms, the TP instance is not reset completely (initialized). The instruction behaves in accordance with the timing diagrams: Only outputs Q and ET are reset. To restart the pulse, a new rising signal edge at input IN is required once PT is greater than 0 again.
- A call with PT < 0 ms resets outputs Q and ET. To restart the pulse, a new rising signal edge at input IN is required once PT is greater than 0 again.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | BOOL | Start input |
| PT | Input | TIME | Duration of pulse; must be positive. |
| Q | Output | BOOL | Pulse output |
| ET | Output | TIME | Current time value |

## Instruction versions

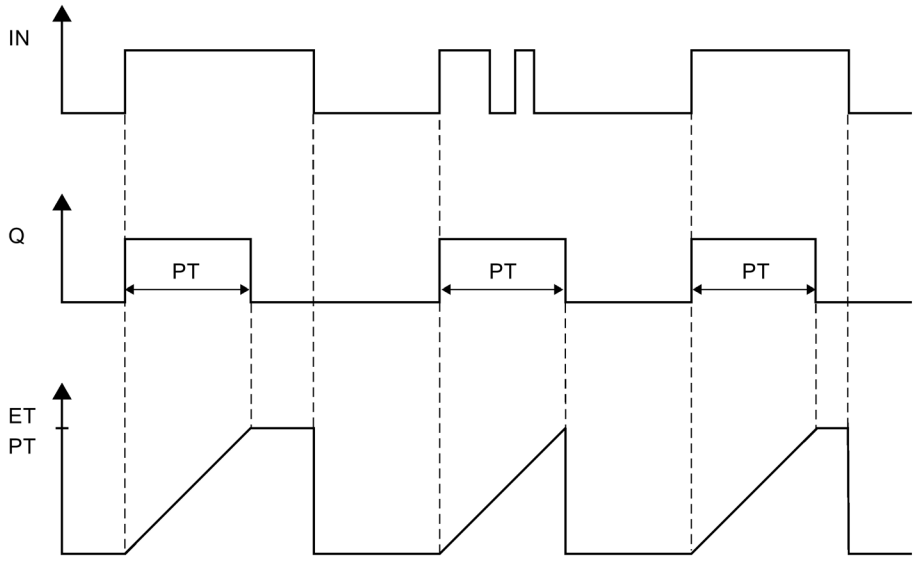
A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

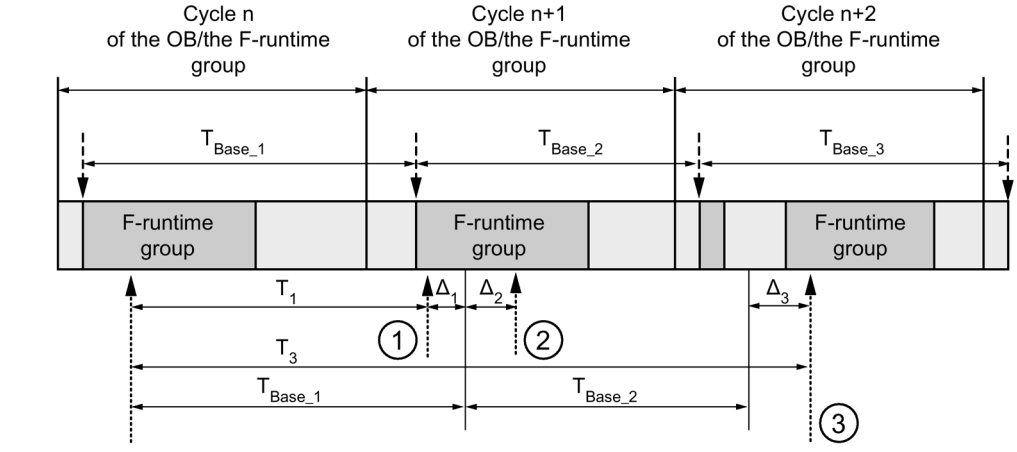
When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Timing diagrams TP



### Timing imprecision resulting from the update time of the time base used in the instruction:




---→ = Time base update

·····→ = Call time of an instruction with time processing

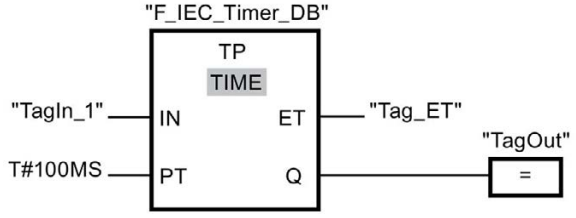
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:




If the signal state of operand "TagIn_1" changes from "0" to "1", the "Generate pulse" instruction is started and the period assigned at input PT (100 ms) runs, regardless of the further course of operand "TagIn_1".

Operand "TagOut" at output Q has signal state "1" as long as the period is running. Operand "Tag_ET" contains the current time value.

### 13.3.4.2 TON: Generate on-delay (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Generate on-delay" instruction to delay the setting of output Q by the assigned period PT. The "Generate on-delay" instruction is started when the result of logic operation (RLO) at input IN changes from "0" to "1" (positive signal edge). The assigned period PT starts running when the instruction starts. When period PT has expired, output Q is set to signal state "1". Output Q remains set as long as the start input is set to "1". When the signal state at the start input changes from "1" to "0", output Q is reset. The time function is restarted when a new positive signal edge is detected at the start input.

The current time value can be queried at output ET. The time value begins at T#0s and ends when the value of period PT is reached. Output ET is reset, as soon as the signal state at input IN changes to "0".

Every call of the "Generate on-delay" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Timer_DB_1) or a multi-instance (e.g., F_IEC_Timer_Instance_1) for the "Generate on-delay" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

The operating system resets the instances of the "Generate on-delay" instruction on a startup of the F-system.

**Note**

The functionality of this instruction differs from the corresponding standard TON instruction in the following points:

- When a call is made with PT = 0 ms, the instance of the TON is not reset completely (initialized). The instruction behaves in accordance with the timing diagrams: Only output ET is reset. To restart the on-delay, a new rising signal edge at input IN is required once PT is greater than 0 again.
- A call with PT < 0 ms resets outputs Q and ET. To restart the on-delay, a new rising signal edge at input IN is required once PT is greater than 0 again.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | BOOL | Start input |
| PT | Input | TIME | Duration of on-delay; must be positive. |
| Q | Output | BOOL | Output that is set after expiration of time PT. |
| ET | Output | TIME | Current time value |

## Instruction versions

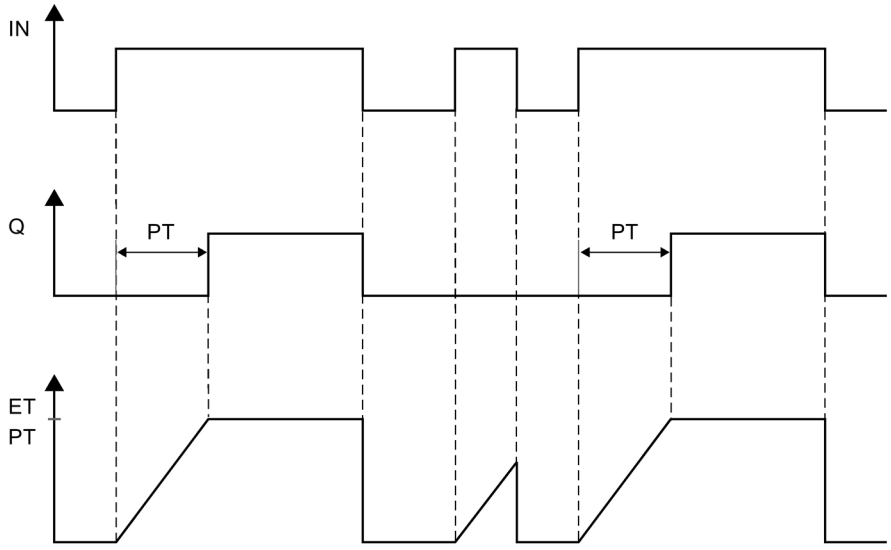
A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

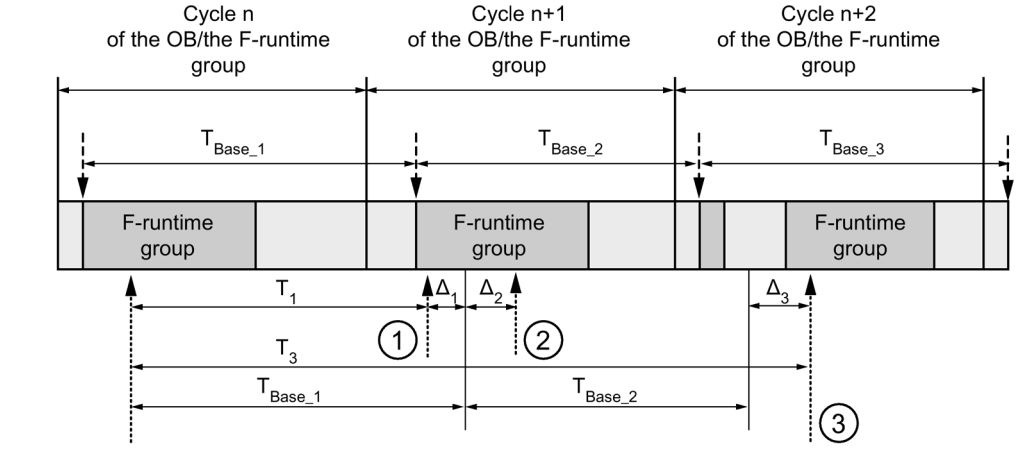
When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Pulse diagram



### Timing imprecision resulting from the update time of the time base used in the instruction:



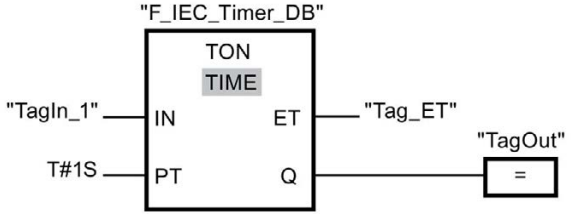

① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:




When the signal state of operand "TagIn_1" changes from "0" to "1", the

"Generate on-delay" instruction is started and the period assigned at input PT (1 s) runs.

Operand "TagOut" at output Q feeds signal state "1" when the period has elapsed and remains set as long as operand "TagIn_1" still feeds signal state "1". Operand "Tag_ET" contains the current time value.

### 13.3.4.3 TOF: Generate off-delay (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Generate off-delay" instruction to delay resetting output Q by the assigned period PT. Output Q is set if the result of logic operation (RLO) changes from "0" to "1" (positive edge) at input IN. The assigned period PT starts when the signal state at input IN changes back to "0". Output Q remains set as long as period PT runs. After period PT expires, output Q is reset. If the signal state at input IN changes to "1" before period PT has expired, then the time is reset. The signal state at output Q remains at "1".

The current time value can be queried at output ET. The time value begins at T#0s and ends when the value of period PT is reached.

Every call of the "Generate off-delay" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is in inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Timer_DB_1) or a multi-instance (e.g., F_IEC_Timer_Instance_1) for the "Generate off-delay" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

**⚠ WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

The operating system resets the instances of the "Generate off-delay" instruction on a startup of the F-system.

**Note**

The functionality of this instruction differs from the corresponding standard TOF instruction in the following points:

- When a call is made with PT = 0 ms, the instance of the TOF is not reset completely (initialized). The instruction behaves in accordance with the timing diagrams: Only outputs Q and ET are reset. To restart the off-delay, another falling signal edge at input IN is required once PT is greater than 0 again.
- A call with PT < 0 ms resets outputs Q and ET. To restart the off-delay, another falling signal edge at input IN is required once PT is greater than 0 again.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | BOOL | Start input |
| PT | Input | TIME | Duration of off-delay; PT must be positive. |
| Q | Output | BOOL | Output that is reset after expiration of time PT. |
| ET | Output | TIME | Current time value |

## Instruction versions

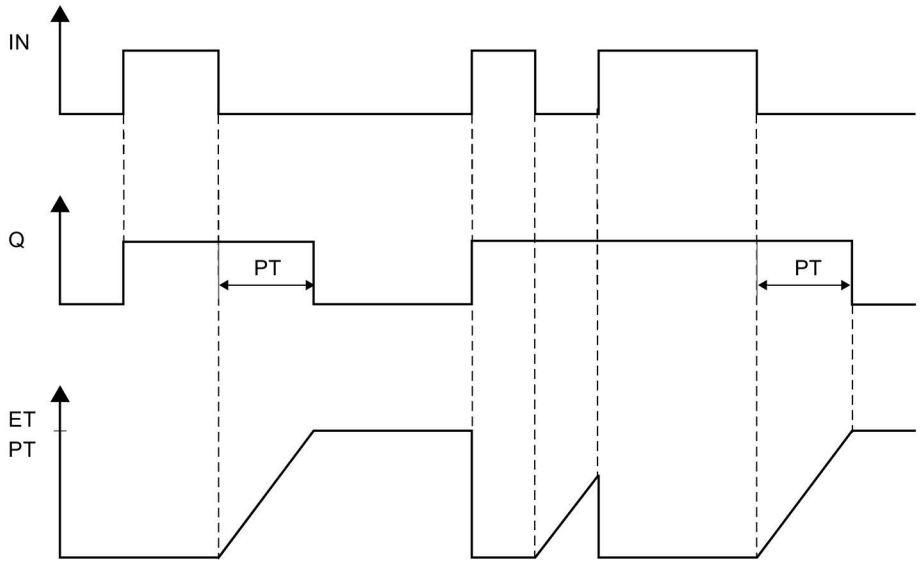
A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

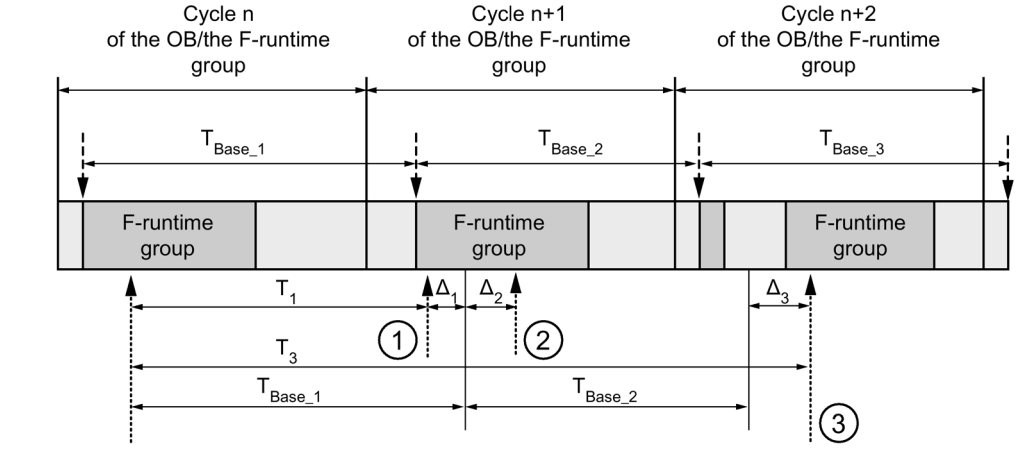
When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Pulse diagram



### Timing imprecision resulting from the update time of the time base used in the instruction:




---→ = Time base update

·····→ = Call time of an instruction with time processing

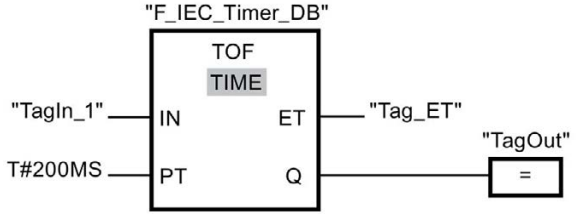
① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

The following example shows how the instruction works:




If the signal state of operand "TagIn_1" changes from "0" to "1", the signal state of operand"TagOut" at output Q is set to "1".

If the signal state of operand "TagIn_1" changes back to "0", the period assigned at input PT (200 ms) runs.

The "TagOut" operand at output Q is set back to "0" when the period expires. Operand "Tag_ET" contains the current time value.

## 13.3.5 Counter operations

### 13.3.5.1 CTU: Count up (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Count up" instruction to increment the value at output CV. When the signal state at input CU changes from "0" to "1" (positive signal edge), the instruction is executed and the current count value at output CV is increased by one. The count value is increased on each detection of a positive signal edge until it reaches the high limit of the data type specified at output CV. When the high limit is reached, the signal state at input CU no longer affects the instruction.

The counter status can be queried at output Q. The signal state at output Q is determined by parameter PV. When the current count value is greater than or equal to the value of parameter PV, output Q is set to signal state "1". In all other cases, the signal state at output Q is "0".

The value at output CV is reset to zero when the signal state at input R changes to "1". As long as signal state "1" exists at input R, the signal state at input CU has no effect on the instruction.

Every call of the "Count up" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Counter_DB_1) or a multi-instance (e.g., F_IEC_Counter_Instance_1) for the "Count up" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

The operating system resets the instances of the "Count up" instruction on a startup of the F-system.

#### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CU | Input | BOOL | Counter input |
| R | Input | BOOL | Reset input |
| PV | Input | INT | Value for which output Q is set |
| Q | Output | BOOL | Counter status |
| CV | Output | INT | Current count value |

## Instruction versions

A number of versions are available for this instruction:

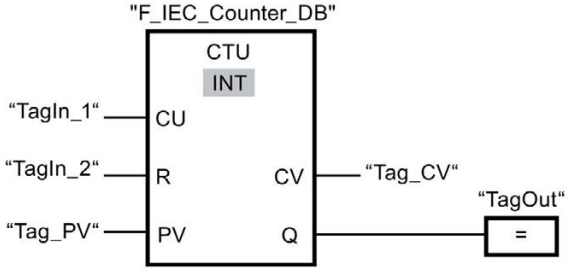
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




When the signal state of the "TagIn_1" operand changes from "0" to "1", the "Count up" instruction is executed and the current count value of the "Tag_CV" operand is increased by one. The count value is increased on every additional positive signal edge until the high limit of the specified data type (32767) is reached.

The value at parameter PV is taken as the limit for the determination of output "TagOut". Output "TagOut" delivers the signal state "1" as long as the current count value is greater than or equal to the value of operand "Tag_PV". In all other cases, output TagOut has signal state "0".

### 13.3.5.2 CTD: Count down (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Count down" instruction to decrement the value at output CV. When the signal state at input CD changes from "0" to "1" (positive signal edge), the instruction is executed and the current count value at output CV is decreased by one. The count value is decreased on each detection of a positive signal edge until it reaches the low limit of the specified data type. When the low limit is reached, the signal state at input CD no longer affects the instruction.

The counter status can be queried at output Q. When the current count value is less than or equal to zero, output Q is set to signal state "1". In all other cases, the signal state at output Q is "0".

The value at output CV is set to the value of parameter "PV" when the signal state at input LD changes to "1". As long as signal state "1" exists at input LD, the signal state at input CD has no effect on the instruction.

Every call of the "Count down" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Counter_DB_1) or a multi-instance (e.g., F_IEC_Counter_Instance_1) for the "Count down" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

The operating system resets the instances of the "Count down" instruction on a startup of the F-system.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CD | Input | BOOL | Counter input |
| LD | Input | BOOL | Load input |
| PV | Input | INT | Value at the output CV when LD = 1 is set |
| Q | Output | BOOL | Counter status |
| CV | Output | INT | Current count value |

## Instruction versions

A number of versions are available for this instruction:

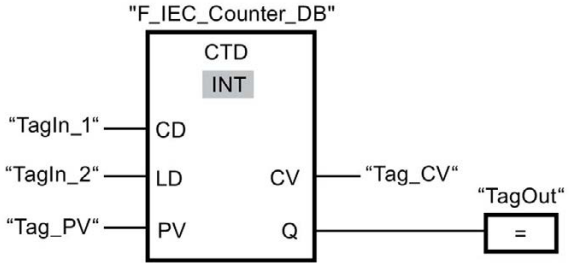
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




If the signal state of the "TagIn_1" operand changes from "0" to "1", the "Count down" instruction is executed and the current count value at output "Tag_CV" is decreased by one. The count value is decreased on each additional positive signal edge until the low limit of the specified data type (-32768) is reached.

Output "TagOut" delivers the signal state "1" as long as the current count value is less than or equal to zero. In all other cases, output TagOut has signal state "0".

### 13.3.5.3 CTUD: Count up and down (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Count up and down" instruction to increment and decrement the count value at output CV. If the signal state at input CU changes from "0" to "1" (positive signal edge), the current count value at output CV is increased by one. If the signal state at input CD changes from "0" to "1" (positive signal edge), the count value at output CV is decreased by one. If a positive signal edge is present at inputs CU and CD in one program cycle, the current count value at output CV remains unchanged.

The count value can be increased until it reaches the high limit of the data type specified at output CV. When the high limit is reached, the count value is no longer incremented on a positive signal edge. When the low limit of the specified data type is reached, the count value is no longer decreased.

When the signal state at input LD changes to "1", the count value at output CV is set to the value of parameter PV. As long as signal state "1" exists at input LD, the signal state at inputs CU and CD has no effect on the instruction.

The count value is set to zero, when the signal state at input R changes to "1". As long as signal state "1" exists at input R, the signal state at inputs CU, CD, and LD has no effect on the "Count up and down" instruction.

The status of the up counter can be queried at output QU. When the current count value is greater than or equal to the value of parameter PV, output QU delivers signal state "1". In all other cases, the signal state at output QU is "0".

The status of the down counter can be queried at output QD. When the current count value is lesser than or equal to zero, output QD delivers signal state "1". In all other cases, the signal state at output QD is "0".

Every call of the "Count up and down" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., F_IEC_Counter_DB_1) or a multi-instance (e.g., F_IEC_Counter_Instance_1) for the "Count up and down" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the *help on STEP 7*.

The operating system resets the instances of the "Count up and down" instruction on a startup of the F-system.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| CU | Input | BOOL | Count up input |
| CD | Input | BOOL | Count down input |
| R | Input | BOOL | Reset input |
| LD | Input | BOOL | Load input |
| PV | Input | INT | Value set at the output QU/ at which the output CV is set at LD = 1. |
| QU | Output | BOOL | Status of up counter |
| QD | Output | BOOL | Status of down counter |
| CV | Output | INT | Current count value |

## Instruction versions

A number of versions are available for this instruction:

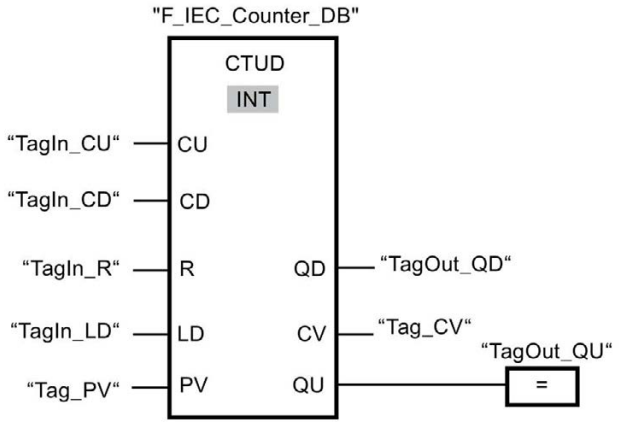
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions have identical functions to version V1.0. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




If the signal state at input "TagIn_CU" or at input "TagIn_CD" changes from "0" to "1" (positive signal edge), the "Count up and down" instruction is executed. When a positive signal edge is present at input "TagIn_CU", the current count value of the "Tag_CV" operand is increased by one. When a positive signal edge is present at input "TagIn_CD", the current count value at output "Tag_CV" is decreased by one. The count value is increased on each positive signal edge at input CU until it reaches the high limit of 32767. The count value is decreased on each positive signal edge at input CD until it reaches the low limit of -32768.

Output "TagOut_QU" delivers the signal state "1" as long as the current count value is greater than or equal to the value at input "Tag_PV". In all other cases, output TagOut_QU has signal state "0".

Output "TagOut_QD" delivers the signal state "1" as long as the current count value is less than or equal to zero. In all other cases, output TagOut_QD has signal state "0".

## 13.3.6 Comparator operations

### 13.3.6.1 CMP ==: Equal (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

Using the "Equal" instruction you can query whether the value at input IN1 is equal to the value at input IN2.

If the condition of the comparison is fulfilled, the instruction returns result of logic operation (RLO) "1". If the condition is not fulfilled, the instruction returns RLO "0".

#### Parameters

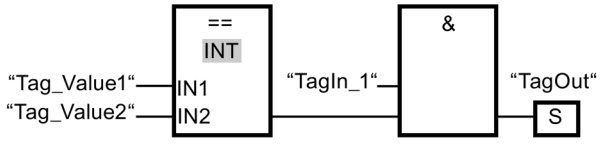
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT, TIME, WORD, DWORD | First value to compare |
| IN2 | Input | INT, DINT, TIME, WORD, DWORD | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

#### Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- The condition of the comparison instruction is fulfilled ("Tag_Value1" = "Tag_Value2").

### 13.3.6.2 CMP <>: Unequal (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

Using the "Not equal" instruction you can query whether the value at input IN1 is not equal to the value at input IN2.

If the condition of the comparison is fulfilled, the instruction returns result of logic operation (RLO) "1". If the condition is not fulfilled, the instruction returns RLO "0".

## Parameters

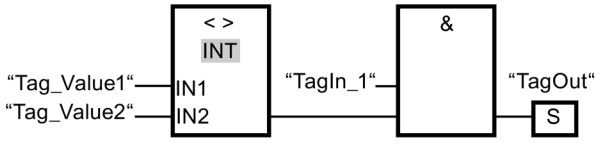
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT, TIME, WORD, DWORD | First value to compare |
| IN2 | Input | INT, DINT, TIME, WORD, DWORD | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- The condition of the comparison instruction is fulfilled ("Tag_Value1" <> "Tag_Value2").

### 13.3.6.3 CMP >=: Greater or equal (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

Using the "Greater or equal" instruction you can query whether the value at input IN1 is greater than or equal to the value at input IN2. Both comparison values must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not fulfilled, the instruction returns RLO "0".

## Parameters

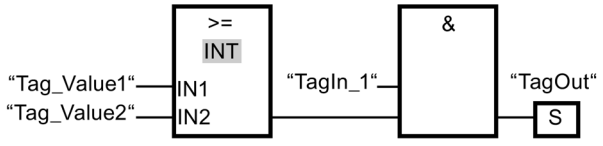
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT, TIME | First value to compare |
| IN2 | Input | INT, DINT, TIME | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- The condition of the comparison instruction is fulfilled ("Tag_Value1" >= "Tag_Value2").

### 13.3.6.4 CMP <=: Less or equal (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

Using the "Less or equal" instruction you can query whether the value at input IN1 is less than or equal to the value at input IN2. Both comparison values must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not fulfilled, the instruction returns RLO "0".

## Parameters

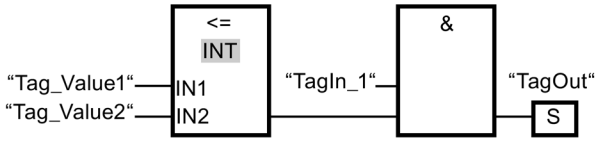
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT, TIME | First value to compare |
| IN2 | Input | INT, DINT, TIME | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- The condition of the comparison instruction is fulfilled ("Tag_Value1" <= "Tag_Value2").

### 13.3.6.5 CMP >: Greater than (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

Using the "Greater than" instruction you can query whether the value at input IN1 is greater than the value at input IN2. Both comparison values must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not fulfilled, the instruction returns RLO "0".

## Parameters

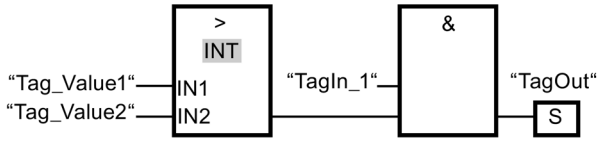
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT, TIME | First value to compare |
| IN2 | Input | INT, DINT, TIME | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- The condition of the comparison instruction is fulfilled ("Tag_Value1" > "Tag_Value2").

### 13.3.6.6 CMP <: Less than (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

Using the "Less than" instruction you can query whether the value at input IN1 is less than the value at input IN2. Both comparison values must be of the same data type.

If the condition of the comparison is fulfilled, the instruction returns result of logic operation (RLO) "1". If the condition of the comparison is not fulfilled, the instruction returns RLO "0".

## Parameters

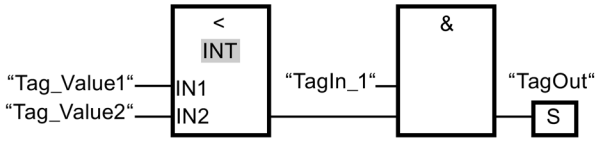
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT, TIME | First value to compare |
| IN2 | Input | INT, DINT, TIME | Second value to compare |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Example

The following example shows how the instruction works:




Output "TagOut" is set when the following conditions are fulfilled:

- Operand "TagIn_1" has signal state "1".
- The condition of the comparison instruction is fulfilled ("Tag_Value1" < "Tag_Value2").

## 13.3.7 Math functions

### 13.3.7.1 ADD: Add (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Add" instruction to add the value at input IN1 and the value at input IN2 and query the sum at the OUT output (OUT = IN1 + IN2).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic alarms.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

#### Parameters

The following table shows the parameters of the instruction:

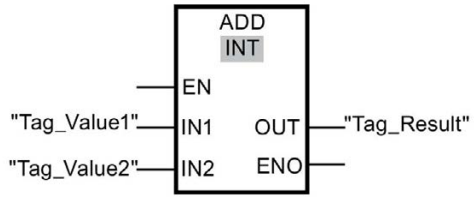
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT | First addend |
| IN2 | Input | INT, DINT | Second addend |
| OUT | Output | INT, DINT | Total |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

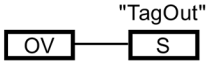
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Add" instruction is always executed (regardless of the signal state at enable input EN).

The value of the "Tag_Value1" operand is added to value of the Tag_Value2 operand. The result of the addition is stored in the "Tag_Result" operand.

(S7-300, S7-400) If an overflow occurs during execution of the "Add" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 645)

### 13.3.7.2 SUB: Subtract (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Subtract" instruction to subtract the value at input IN2 from the value at input IN1 and query the difference at the OUT output (OUT = IN1 – IN2).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic alarms.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

## Parameters

The following table shows the parameters of the instruction:

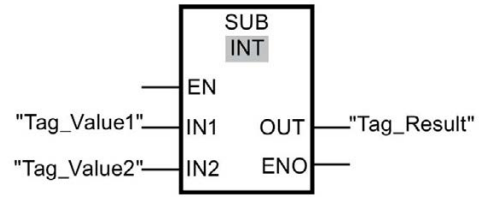
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT | Minuend |
| IN2 | Input | INT, DINT | Subtrahend |
| OUT | Output | INT, DINT | Difference |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

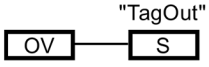
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Subtract" instruction is always executed (regardless of the signal state at enable input EN).

The value of operand "Tag_Value2" is subtracted from the value of operand "Tag_Value1". The result of the addition is stored in operand "Tag_Result".

(S7-300, S7-400) If an overflow occurs during execution of the "Subtract" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 645)

### 13.3.7.3 MUL: Multiply (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Multiply" instruction to multiply the value at input IN1 by the value at input IN2 and query the product at output OUT (OUT = IN1 × IN2).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic alarms.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

## Parameters

The following table shows the parameters of the instruction:

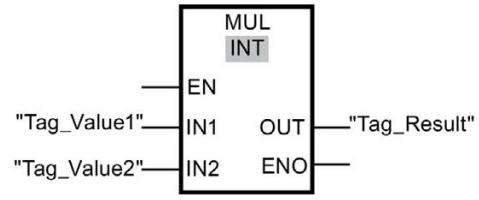
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT | Multiplier |
| IN2 | Input | INT, DINT | Multiplicand |
| OUT | Output | INT, DINT | Product |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

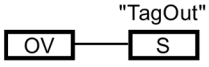
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Multiply" instruction is always executed (regardless of the signal state at enable input EN).

The value of the "Tag_Value1" operand is multiplied by the value of the "Tag_Value2" operand. The result of the multiplication is stored in the "Tag_Result" operand.

(S7-300, S7-400) If an overflow occurs during execution of the "Multiply" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 645)

### 13.3.7.4 DIV: Divide (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Divide" instruction to divide the value at input IN1 by the value at input IN2 and query the quotient at the OUT output (OUT = IN1 / IN2).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts in this case like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

**Note**

S7-300/400:

If the divisor (input IN2) of a DIV instruction = 0, the quotient of the division (result of division at output OUT) = 0. The result reacts like the corresponding instruction in a standard block. The F-CPU does *not* go to STOP mode.

S7-1200/1500:

If the divisor (input IN2) of a DIV instruction = 0, the F-CPU goes to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages. We recommend that you rule out a divisor (input IN2) = 0 when creating the program.

## Parameters

The following table shows the parameters of the instruction:

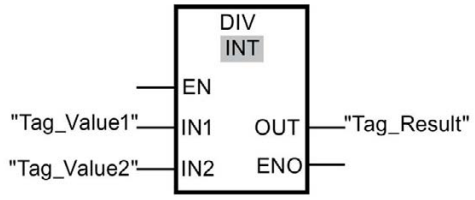
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | INT, DINT | Dividend |
| IN2 | Input | INT, DINT | Divisor |
| OUT | Output | INT, DINT | Quotient |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

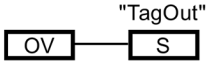
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Divide" instruction is always executed (regardless of the signal state at enable input EN).

The value of operand "Tag_Value1" is divided by the value of operand "Tag_Value2". The result of the division is stored in operand "Tag_Result".

(S7-300, S7-400) If an overflow occurs during execution of the "Divide" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 645)

### 13.3.7.5 NEG: Create twos complement (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Create twos complement" instruction to change the sign of the value at input IN input and query the result at output OUT. If there is a positive value at input IN, for example, the negative equivalent of this value is sent to output OUT.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

If the result of the instruction is located outside the permitted range for this data type, the F-CPU may switch to STOP. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic alarms.

You must therefore ensure that the permitted range for the data type is observed when creating the program!

(S7-300, S7-400) You can avoid a STOP of the F-CPU by inserting a "Get status bit OV" instruction in the next network, thereby programming overflow detection.

Note the following:

- The result of the instruction reacts like the analogous instruction in a standard block.
- The network with the "Get status bit OV" instruction must not contain any jump labels.
- The execution time of the instruction is extended (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).
- A warning is issued if you do not insert a "Get status bit OV" instruction.

## Parameters

The following table shows the parameters of the instruction:

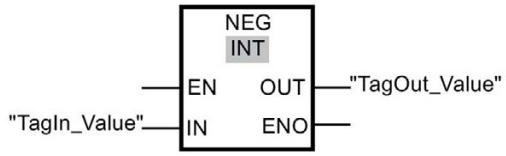
| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT, DINT | Input value |
| OUT | Output | INT, DINT | Twos complement of the input value |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

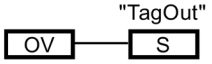
## Example

The following example shows how the instruction works:

Network 1:




Network 2: (S7-300, S7-400)




The "Create twos complement" instruction is always executed (regardless of the signal state at enable input EN).

The sign of the "TagIn_Value" operand is changed and the result is stored in the "TagOut_Value" operand.

(S7-300, S7-400) If an overflow occurs during execution of the "Create twos complement" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## See also

OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 645)

## 13.3.8 Move operations

### 13.3.8.1 MOVE: Move value (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Move value" instruction to transfer the content of the operand at input IN to the operand at output OUT1.

Only identical operand widths can be specified for input IN and output OUT1.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

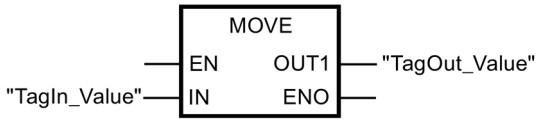
#### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT, DINT, WORD, DWORD, TIME | Source value |
| OUT1 | Output | INT, DINT, WORD, DWORD, TIME | Destination address |

#### Example

The following example shows how the instruction works:




The instruction is always executed regardless of the signal state at enable input "EN". The instruction copies the content of operand "TagIn_Value" to operand "TagOut_Value".

### 13.3.8.2 WR_FDB: Write value indirectly to an F-DB (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

This instruction writes the value specified in input IN to the tag addressed by INI_ADDR and OFFSET in an F-DB.

The address of the tags addressed using INI_ADDR and OFFSET must be within the address range defined by addresses INI_ADDR and END_ADDR.

If the F-CPU has gone to STOP mode with diagnostic event ID 75E2, check to determine if this condition is satisfied.

The start address of the area in an F-DB to which the value at input IN is to be written is transferred using input INI_ADDR. The associated offset in this area is transferred using input OFFSET.

The addresses transferred in input INI_ADDR or END_ADDR must point to a tag of the selected data type in an F-DB. Only tags of the selected data type are permitted between the INI_ADDR and END_ADDR addresses. The INI_ADDR address must be smaller than the END_ADDR address.

As shown in the following example, the INI_ADDR and END_ADDR addresses must be transferred fully-qualified as "DBx".DBWy or in the corresponding symbolic representation. Transfers in other forms are not permitted.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT, DINT | Value to be written to the F-DB |
| INI_ADDR | Input | POINTER | Start address of the area in an F-DB |
| END_ADDR | Input | POINTER | End address of the area in an F-DB |
| OFFSET | Input | INT | Offset |

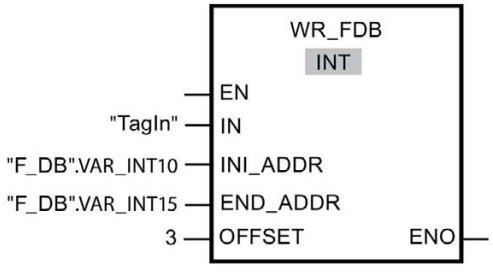
You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

## Examples of parameter assignment of INI_ADDR, END_ADDR, and OFFS

| Name | Data type | Initial value | Comment |
|---|---|---|---|
| Static | | | |
| VAR_BOOL10 | Bool | false | |
| VAR_BOOL11 | Bool | false | |
| VAR_BOOL12 | Bool | false | |
| VAR_BOOL13 | Bool | false | |
| VAR_TIME10 | Time | T#0MS | |
| VAR_TIME11 | Time | T#0MS | |
| VAR_INT10 | Int | 0 | <- INI_ADDR = "F-DB".VAR_INT10 Example 1 |
| VAR_INT11 | Int | 0 | |
| VAR_INT12 | Int | 0 | |
| VAR_INT13 | Int | 0 | <- OFFSET = 3 |
| VAR_INT14 | Int | 0 | |
| VAR_INT15 | Int | 0 | <- END_ADDR = "F-DB".VAR_INT15 |
| VAR_BOOL20 | Bool | false | |
| VAR_BOOL21 | Bool | false | |
| VAR_BOOL22 | Bool | false | |
| VAR_BOOL23 | Bool | false | |
| VAR_INT20 | Int | 0 | <- INI_ADDR = "F-DB".VAR_INT20 Example 2 |
| VAR_INT21 | Int | 0 | |
| VAR_INT22 | Int | 0 | |
| VAR_INT23 | Int | 0 | <- END_ADDR = "F-DB".VAR_INT23 |
| VAR_INT30 | Int | 0 | <- INI_ADDR = "F-DB".VAR_INT30 Example 3 |
| VAR_INT31 | Int | 0 | <- OFFSET = 1 |
| VAR_INT32 | Int | 0 | |
| VAR_INT33 | Int | 0 | |
| VAR_INT34 | Int | 0 | <- END_ADDR = "F-DB".VAR_INT34 |
| VAR_TIME20 | TIME | T#0MS | |
| VAR_DINT10 | DInt | 0 | <- INI_ADDR = "F-DB".VAR_DINT10 Example 4 |
| VAR_DINT11 | DInt | 0 | |
| VAR_DINT12 | DInt | 0 | <- OFFSET = 2 |
| VAR_DINT13 | DInt | 0 | <- END_ADDR = "F-DB".VAR_DINT13 |

## Example

The following example shows how the instruction works:



### 13.3.8.3 RD_FDB: Read value indirectly from an F-DB (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

This instruction reads the tag addressed via INI_ADDR and OFFSET in an F-DB and provides it at output OUT.

The address of the tags addressed using INI_ADDR and OFFSET must be within the address range defined by addresses INI_ADDR and END_ADDR.

If the F-CPU has gone to STOP mode with diagnostic event ID 75E2, check to determine if this condition is satisfied.

The start address of the area in an F-DB from which the tag is to be read is transferred using input INI_ADDR. The associated offset in this area is transferred using input OFFSET.

The addresses transferred in input INI_ADDR or END_ADDR must point to a tag of the selected data type in an F-DB. Only tags of the selected data type are permitted between the INI_ADDR and END_ADDR addresses. The INI_ADDR address must be smaller than the END_ADDR address.

The INI_ADDR and END_ADDR addresses must be transferred fully-qualified as "DBx".DBWy or in the corresponding symbolic representation. Transfers in other forms are not permitted. Examples of parameter assignment of INI_ADDR, END_ADDR, and OFFSET are contained in WR_FDB: Write value indirectly to an F-DB (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400) (Page 614).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameters

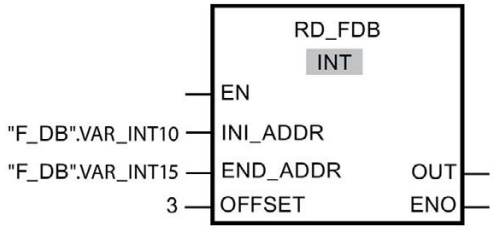
The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| INI_ADDR | Input | POINTER | Start address of the area in an F-DB |
| END_ADDR | Input | POINTER | End address of the area in an F-DB |
| OFFSET | Input | INT | Offset |
| OUT | Output | INT, DINT | Value to be read from the F-DB |

You can select the data type of the instruction in the "<???>" drop-down list in the instruction box.

### Example

The following example shows how the instruction works:




## 13.3.9 Conversion operations

### 13.3.9.1 CONVERT: Convert value (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

### Description

The "Convert value" instruction reads the content of parameter IN and converts it according to the data types selected in the instruction box. The converted value is output at output OUT.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

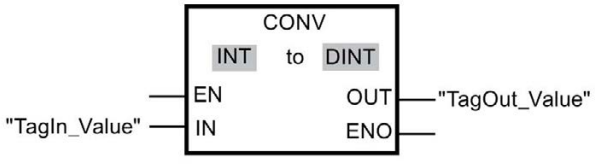
### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT | Value to be converted. |
| OUT | Output | DINT | Result of the conversion |

### Example

The following example shows how the instruction works:




The instruction is always executed regardless of the signal state at enable input "EN". The content of the operand "TagIn_Value" is read and converted to a double integer (32 bit). The result is stored in operand "TagOut_Value".

### 13.3.9.2 BO_W: Convert 16 data elements of data type BOOL to a data element of data type WORD (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction converts the 16 values of data type BOOL at inputs IN0 to IN15 to a value of data type WORD, which is made available at output OUT. The conversion takes place as follows: The i-th bit of the WORD value is set to 0 (or 1), if the value at input INi = 0 (or 1).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN0 | Input | BOOL | Bit 0 of WORD value |
| IN1 | Input | BOOL | Bit 1 of WORD value |
| ... | | | ... |
| IN15 | Input | BOOL | Bit 15 of WORD value |
| OUT | Output | WORD | WORD value consisting of IN0 to IN15 |

## Instruction versions

A number of versions are available for this instruction:

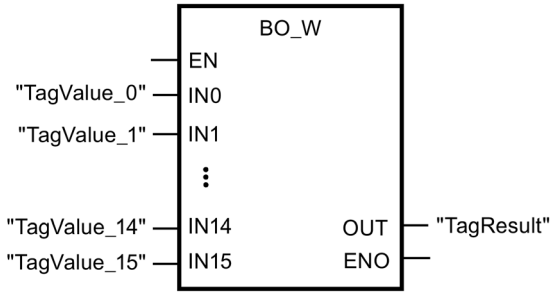
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | These versions have identical functions to version V1.0. |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




The following table shows how the instruction works using specific operand values:

| Parameter | Operand | Value |
|---|---|---|
| IN0 | TagValue_0 | FALSE |
| IN1 | TagValue_1 | FALSE |
| ... | | ... |
| IN13 | TagValue_13 | FALSE |
| IN14 | TagValue_14 | TRUE |
| IN15 | TagValue_15 | TRUE |
| OUT | TagResult | W#16#C000 |

The values of operands "TagValue_0" to " TagValue_15" are combined to form data type WORD and assigned to operand "TagResult".

### 13.3.9.3 W_BO: Convert 16 data elements of data type WORD to a data element of data type BOOL (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

This instruction converts the value of data type WORD at input IN to 16 values of data type BOOL, which are provided at outputs OUT0 to OUT15. The conversion takes place as follows: Output OUTi is set to 0 (or 1), if the i-th bit of the WORD value is 0 (or 1).

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | WORD | WORD value |
| OUT0 | Output | BOOL | Bit 0 of WORD value |
| OUT1 | Output | BOOL | Bit 1 of WORD value |
| ... | | | ... |
| OUT15 | Output | BOOL | Bit 15 of WORD value |

## Instruction versions

A number of versions are available for this instruction:

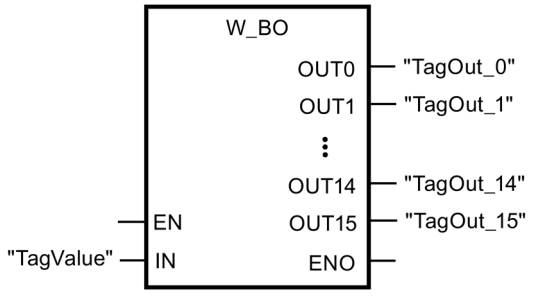
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | These versions have identical functions to version V1.0. |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




The following table shows how the instruction works using specific operand values:

| Parameter | Operand | Value |
|---|---|---|
| IN | TagValue | W#16#C000 |
| OUT0 | TagOUT_0 | FALSE |
| OUT1 | TagOUT_1 | FALSE |
| ... | | ... |
| OUT13 | TagOUT_13 | FALSE |
| OUT14 | TagOUT_14 | TRUE |
| OUT15 | TagOUT_15 | TRUE |

The value of operand "TagValue" of data type WORD is converted to the 16 values "TagOUT_0" to "TagOUT_15" of data type BOOL.

### 13.3.9.4 SCALE: Scale values (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500)

## Description

This instruction scales the value at input IN in physical units between the low limit value at input LO_LIM and the high limit value at input HI_LIM. It is assumed that the value at input IN is between 0 and 27648. The scaling result is provided at output OUT.

The instruction uses the following equation:

OUT = [ IN × (HI_LIM – LO_LIM) ] / 27648 + LO_LIM

As long as the value at input IN is greater than 27648, output OUT is linked to HI_LIM and OUT_HI is set to 1.

As long as the value at input IN is less than 0, output OUT is linked to LO_LIM and OUT_LO is set to 1.

For inverse scaling, you must assign LO_LIM > HI_LIM. With inverse scaling, the output value at output OUT decreases while the input value at input IN increases.

Every call of the "Scale values" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened for this reason when the instruction is inserted in the program; in it you can create a data block (single instance) (e.g., SCALE_DB_1) or a multi-instance (e.g., SCALE_Instance_1) for the "Scale values" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | INT | Input value to be scaled in physical units |
| HI_LIM | Input | INT | High limit value of value range of OUT |
| LO_LIM | Input | INT | Low limit value of value range of OUT |
| OUT | Output | INT | Result of scaling |
| OUT_HI | Output | BOOL | 1 = Input value > 27648: OUT = HI_LIM |
| OUT_LO | Output | BOOL | 1 = Input value < 0: OUT = LO_LIM |

## Instruction versions

A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | This version has identical functions to version V1.0. |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Behavior in the event of overflow or underflow of analog values and fail-safe value output

**Note**

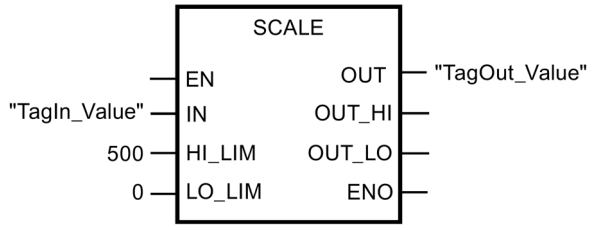
If inputs from the PII of an SM 336; AI 6 x 13Bit or SM 336; F-AI 6 x 0/4 ... 20 mA HART are used as input values, note that the F-system detects an overflow or underflow of a channel of this F-SM as an F-I/O fault or channel fault. The fail-safe value 0 is provided in place of $7FFF_H$ (for overflow) or $8000_H$ (for underflow) in the PII for the safety program.

If other fail-safe values are to be output in this case, you must evaluate the QBAD tag in the F-I/O DB (branch to output of an individual fail-safe value).

If the value in the PII of the F-SM is within the overrange or underrange, but is > 27648 or < 0, you can likewise branch to the output of an individual fail-safe value by evaluating outputs OUT_HI and OUT_LO, respectively.

## Example

The following example shows how the instruction works:




When operand "TagIn_Value" = 20000, the result is "TagOut_Value" 361.

## 13.3.10 Program control operations

### 13.3.10.1 JMP: Jump if RLO = 1 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "Jump if RLO = 1" instruction to interrupt the linear execution of the program and resume it in another network. The destination network must be identified by a jump label (Page 628) (LABEL). The designation of the jump label is specified in the placeholder above the instruction box.

The specified jump label must be in the same block in which the instruction is executed. The name you specify can only occur once in the block.

If the result of logic operation (RLO) at the input of the instruction is "1" or the input is not connected, the jump to the network identified by the jump label is executed. The jump direction can be towards higher or lower network numbers.

If the result of logic operation (RLO) at the input of the instruction is "0", the program continues executing in the next network.

**Note**

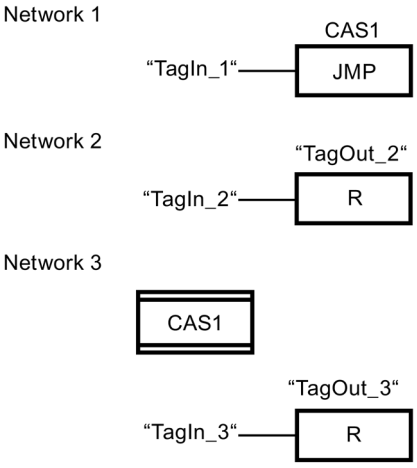
(S7-1200, S7-1500)
If the jump destination (jump label) for an instruction "JMP" or "JMPN" is above the associated instruction "JMP" or "JMPN" (backwards jump), you cannot insert any other instructions for program control (JMP, JMPN, RET, jump label) between them.
**Exception:** You can insert an instruction "JMP" or "JMPN" between them if you also insert the associated jump destination in between as well as below the associated instruction "JMP" or "JMPN" (forward jump).
Non-compliance can lead to compilation errors or to the CPU going to STOP.

**Note**

You are not permitted to insert any SENDDP or SENDS7 instructions between an instruction JMP or JMPN and the associated jump destination (jump label).

## Example

The following example shows how the instruction works:




When operand "TagIn_1" has signal state "1", the "Jump if RLO = 1" instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. When input "TagIn_3" has signal state "1", output "TagOut_3" is reset.

### 13.3.10.2 JMPN: Jump if RLO = 0 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Jump if RLO = 0" instruction to interrupt the linear execution of the program and resume it in another network, when the result of logic operation at the input of the instruction is "0". The destination network must be identified by a jump label (Page 628) (LABEL). The designation of the jump label is specified in the placeholder above the instruction box.

The specified jump label must be in the same block in which the instruction is executed. The name you specify can only occur once in the block.

If the result of logic operation (RLO) at the input of the instruction is "0", the jump to the network identified by the jump label is executed. The jump direction can be towards higher or lower network numbers.

If the result of logic operation (RLO) at the input of the instruction is "1", the program continues executing in the next network.

**Note**

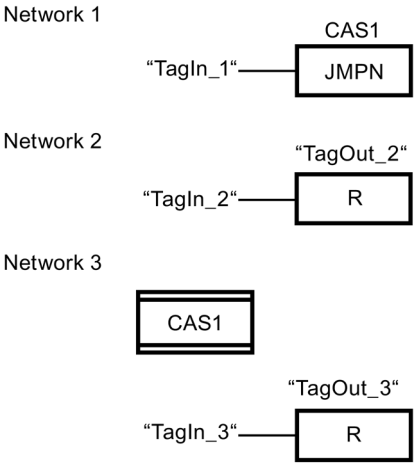
(S7-1200, S7-1500)
If the jump destination (jump label) for an instruction "JMP" or "JMPN" is above the associated instruction "JMP" or "JMPN" (backwards jump), you cannot insert any other instructions for program control (JMP, JMPN, RET, jump label) between them.
**Exception:** You can insert an instruction "JMP" or "JMPN" between them if you also insert the associated jump destination in between as well as below the associated instruction "JMP" or "JMPN" (forward jump).
Non-compliance can lead to compilation errors or to the CPU going to STOP.

**Note**

You are not permitted to insert any SENDDP or SENDS7 instructions between an instruction JMP or JMPN and the associated jump destination (jump label).

## Example

The following example shows how the instruction works:




When operand "TagIn_1" has signal state "0", instruction "Jump if RLO = 0" is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. When input "TagIn_3" has signal state "1", output "TagOut_3" is reset.

### 13.3.10.3 LABEL: Jump label (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

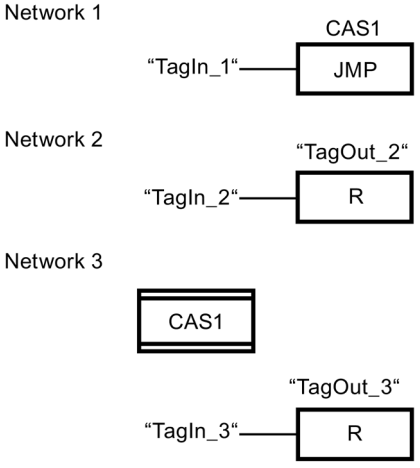
You can use a jump label to specify a destination network, in which the program execution should resume after a jump.

The jump label and the instruction in which the jump label is specified must be located in the same block. The name of a jump label can only be assigned once in a block.

Only one jump label can be placed in a network. To each jump label can be jumped from several locations.

## Example

The following example shows how the instruction works:




When operand "TagIn_1" has signal state "1", the "JMP: Jump if RLO = 1" instruction is executed. The linear execution of the program is interrupted and continues in Network 3, which is identified by the jump label CAS1. When input "TagIn_3" has signal state "1", output "TagOut_3" is reset.

## See also

JMP: Jump if RLO = 1 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 624)

JMPN: Jump if RLO = 0 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 626)

RET: Return (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 629)

### 13.3.10.4 RET: Return (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "Return" instruction to stop the processing of a block.

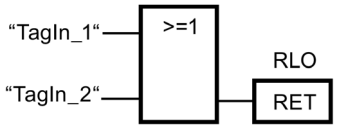
If the result of logic operation (RLO) at the input of the "Return" instruction is "1" or the box input of the S7-1200/1500 F-CPUs is not connected, program execution is terminated in the currently called block and continued in the calling block (for example, in the main safety block) after the call function. If the RLO at the input of the "Return" instruction is "0", the instruction is not executed. Program execution continues in the next network of the called block.

**Note**

You cannot program a "Return" instruction before a main safety block call.

## Example

The following example shows how the instruction works:




If either the "TagIn_1" or "TagIn_2" operand has the signal state "1", the "Return" instruction is executed. Program execution in the called block is terminated and continues in the calling block.

## See also

LABEL: Jump label (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 628)

JMPN: Jump if RLO = 0 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 626)

JMP: Jump if RLO = 1 (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500) (Page 624)

### 13.3.10.5 OPN: Open global data block (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

## Description

You can use the "Open global data block" instruction to open a data block. The number of the data block is transferred to the DB register. Subsequent DB commands access the relevant blocks depending on the register contents.

**Note**

Note when using the "Open global data block" instruction that the content of the DB register can be changed after calls of F-FB/F-FC and "fully qualified DB accesses", such that there is no guarantee that the last data block you opened with "Open global data block" is still open.

You should therefore use the following method for addressing data to avoid errors when accessing data of the DB register:

- Use symbolic addressing.
- Use only fully qualified DB accesses.

If you still want to use the "Open global data block" operation, you must ensure that the DB register is restored by repeating the "Open global data block" instruction after calls of F-FB/F-FC and "fully qualified DB accesses." Otherwise, a malfunction could result.

## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| <Data block> | Input | BLOCK_DB | Data block that is opened |

## "Fully qualified DB access"

The initial access to data of a data block in an F-FB/F-FC must always be a "fully qualified DB access," or it must be preceded by the "Open global data block" instruction. This also applies to the initial access to data of a data block after a jump label.

An example of "fully qualified DB access" and "non-fully qualified DB access" is provided in Restrictions in the programming languages FBD/LAD (Page 90).

## Access to instance DBs

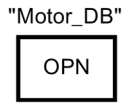
You can also access instance DBs of F-FBs with fully qualified access, e.g., for transfer of block parameters. It is not possible to access static local data in single/multi-instances of other F-FBs.

Note that accessing instance DBs of F-FBs that are not called in the safety program can cause the F-CPU to go to STOP mode.

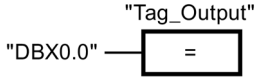
## Example

The following example shows how the instruction works:

Network 1




Network 2




The "Motor_DB" data block is called in network 1. The number of the data block is transferred to the DB register. The "DBX0.0" operand is queried in network 2. The signal state of the "DBX0.0" operand is assigned to the "Tag_Output" operand.

## 13.3.11 Word logic operations

### 13.3.11.1 AND: AND logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

You can use the "AND logic operation" instruction to combine the value at input IN1 to the value at input IN2 bit-by-bit by AND logic and query the result at output OUT.

When the instruction is executed, bit 0 of the value at input IN1 and bit 0 of the value at input IN2 are ANDed. The result is stored in bit 0 of output OUT. The same logic operation is executed for all other bits of the specified values.

The result bit has signal state "1" only when both of the bits in the logic operation also have signal state "1". If one of the two bits of the logic operation has signal state "0", the corresponding result bit is reset.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

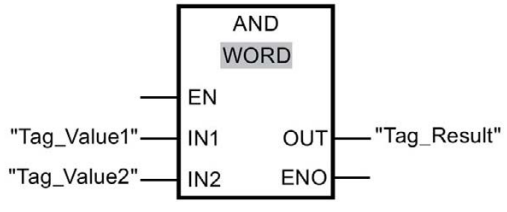
#### Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | WORD | First value of logic operation |
| IN2 | Input | WORD | Second value of logic operation |
| OUT | Output | WORD | Result of the instruction |

#### Example

The following example shows how the instruction works:




| | |
|---|---|
| IN1 | "Tag_Value1" = 01010101 01010101 |
| IN2 | "Tag_Value2" = 00000000 00001111 |
| OUT | "Tag_Result" = 00000000 00000101 |

The instruction is always executed regardless of the signal state at enable input "EN". The value of the "Tag_Value1" operand and the value of the "Tag_Value2" operand are ANDed. The result is mapped bit-by-bit and output in the "Tag_Result" operand.

### 13.3.11.2 OR: OR logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "OR logic operation" instruction to connect the value at input IN1 input to the value at input IN2 bit-by-bit by OR logic and query the result at output OR.

When the instruction is executed, bit 0 of the value at input IN1 and bit 0 of the value at input IN2 are ORed. The result is stored in bit 0 of output OUT. The same logic operation is executed for all bits of the specified tags.

The result bit has signal state "1" when at least one of the two bits in the logic operation has signal state "1". If both of the bits of the logic operation have signal state "0", the corresponding result bit is reset.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

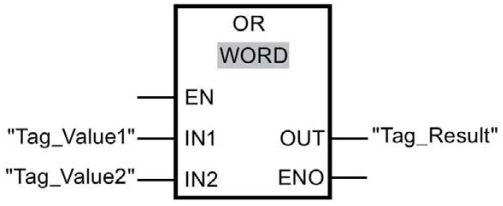
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | WORD | First value of logic operation |
| IN2 | Input | WORD | Second value of logic operation |
| OUT | Output | WORD | Result of the instruction |

## Example

The following example shows how the instruction works:




| | |
|---|---|
| IN1 | "Tag_Value1" = 01010101 01010101 |
| IN2 | "Tag_Value2" = 00000000 00001111 |
| OUT | "Tag_Result" = 01010101 01011111 |

The instruction is always executed regardless of the signal state at enable input "EN". The value of the "Tag_Value1" operand and the value of the "Tag_Value2" operand are ORed. The result is mapped bit-by-bit and output in the "Tag_Result" operand.

### 13.3.11.3 XOR: EXCLUSIVE OR logic operation (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

You can use the "EXCLUSIVE OR logic operation" instruction to combine the value at input IN1 and the value at input IN2 bit-by-bit by EXCLUSIVE OR logic and query the result at output OUT.

When the instruction is executed, bit 0 of the value at input IN1 input and bit 0 of the value at input IN2 are logically combined by EXCLUSIVE OR. The result is stored in bit 0 of output OUT. The same logic operation is executed for all other bits of the specified value.

The result bit has signal state "1" when one of the two bits in the logic operation has signal state "1". If both of the bits of the logic operation have signal state "1" or "0", the corresponding result bit is reset.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

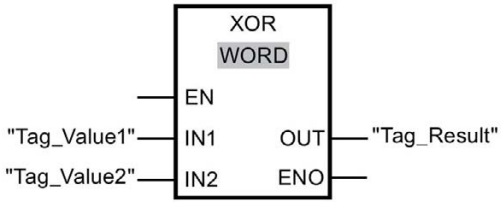
## Parameters

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN1 | Input | WORD | First value of logic operation |
| IN2 | Input | WORD | Second value of logic operation |
| OUT | Output | WORD | Result of the instruction |

## Example

The following example shows how the instruction works:




| | |
|---|---|
| IN1 | "Tag_Value1" = 01010101 01010101 |
| IN2 | "Tag_Value2" = 00000000 00001111 |
| OUT | "Tag_Result" = 01010101 01011010 |

The instruction is always executed regardless of the signal state at enable input "EN". The value of the "Tag_Value1" operand and the value of the "Tag_Value2" operand are logically combined by EXCLUSIVE OR. The result is mapped bit-by-bit and output in the "Tag_Result" operand.

## 13.3.12 Shift and rotate

### 13.3.12.1 SHR: Shift right (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description

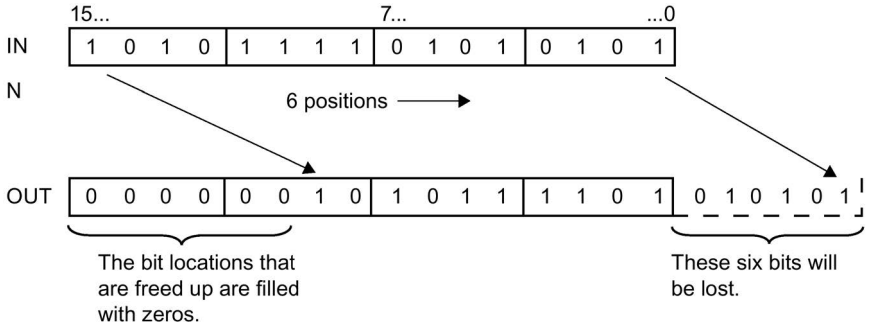
Use the "Shift right" instruction to shift the content of the operand at input IN bit-by-bit to the right and query the result at output OUT. Use input N to specify the number of bit positions by which the specified value is to be moved.

If the value at parameter N is "0", the value at input IN is copied unchanged to the operand at output OUT.

If the value at input N is greater than the number of available bit positions, the operand value at input IN is shifted to the right by the available number of bit positions.

The bit locations that are freed up in the left area of the operand during the shift operation are filled with zeros.

The following figure shows how the content of an operand of data type WORD is moved by 6 bit positions to the right:




Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

S7-300/400:

Only the low-byte is evaluated from input N.

S7-1200/1500:

If the value at input N < 0, the output OUT is set to 0.

## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | WORD | Value that is shifted |
| N | Input | INT | Number of bit positions by which the value is shifted |
| OUT | Output | WORD | Result of the instruction |

## Instruction versions

A number of versions are available for this instruction:

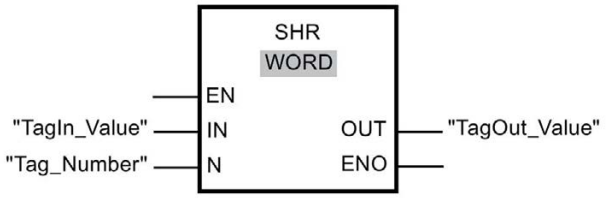
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | These versions have identical functions to version V1.0. |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




The following table shows how the instruction works using specific operand values:

| Parameter | Operand | Value |
|---|---|---|
| IN | TagIn_Value | 0011 1111 1010 1111 |
| N | Tag_Number | 3 |
| OUT | TagOut_Value | 0000 0111 1111 0101 |

The instruction is always executed regardless of the signal state at enable input "EN". The content of the operand "TagIn_Value" is moved three bit positions to the right. The result is output at output "TagOut_Value".

### 13.3.12.2 SHL: Shift left (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

## Description

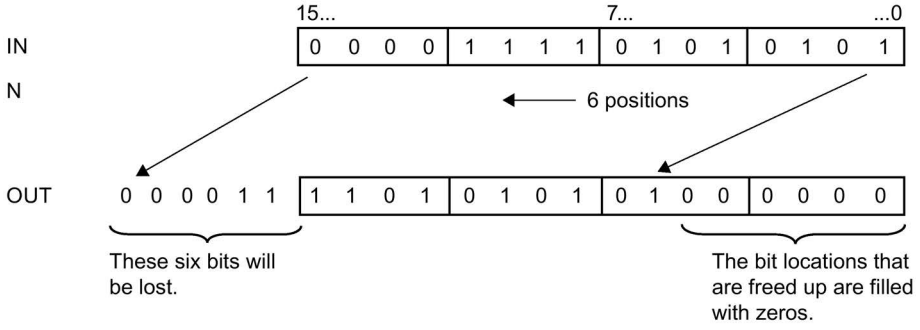
Use the "Shift left" instruction to shift the content of the operand at input IN bit-by-bit to the left and query the result at output OUT. Use input N to specify the number of bit positions by which the specified value is to be moved.

If the value at input N is "0", the value at input IN is copied to the operand at output OUT.

If the value at input N is greater than the number of available bit positions, the operand value at input IN is shifted to the left by the available number of bit positions.

The bit positions that are freed up in the right area of the operand during the shift operation are filled with zeros.

The following figure shows how the content of an operand of data type WORD is moved by 6 bit positions to the left:




Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**Note**

S7-300/400:

Only the low-byte is evaluated from input N.

S7-1200/1500:

If the value at input N < 0, the output OUT is set to 0.

## Parameter

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | Input | WORD | Value that is shifted |
| N | Input | INT | Number of bit positions by which the value is shifted |
| OUT | Output | WORD | Result of the instruction |

## Instruction versions

A number of versions are available for this instruction:

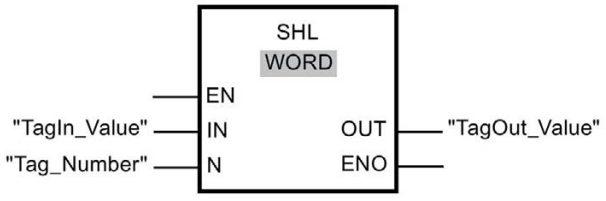
| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | These versions have identical functions to version V1.0. |
| 1.3 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Example

The following example shows how the instruction works:




The following table shows how the instruction works using specific operand values:

| Parameter | Operand | Value |
|---|---|---|
| IN | TagIn_Value | 0011 1111 1010 1111 |
| N | Tag_Number | 4 |
| OUT | TagOut_Value | 1111 1010 1111 0000 |

The instruction is always executed regardless of the signal state at enable input "EN". The content of the operand "TagIn_Value" is moved four bit positions to the left. The result is output at output "TagOut_Value".

## 13.3.13 Operating

### 13.3.13.1 ACK_OP: Fail-safe acknowledgment (STEP 7 Safety V13 SP1) (S7-300, S7-400, S7-1200, S7-1500)

#### Description (S7-300, S7-400)

This instruction enables fail-safe acknowledgment from an HMI system. It allows, for example, reintegration of F-I/O to be controlled from the HMI system. Acknowledgment takes place in two steps:

- Input/output parameter IN changes to a value of 6 for exactly one cycle.
- Input/output parameter IN changes to a value of 9 within a minute for exactly one cycle

Once the in/out parameter IN has changed to a value of 6, the instruction evaluates whether this parameter has changed to a value of 9 after 1 second, at the earliest, or one minute, at the latest. Output OUT (output for acknowledgment) is then set to 1 for one cycle.

If an invalid value is input or if in/out parameter IN has not changed to the value 9 within one minute or the change occurred before one second has elapsed, then in/out parameter IN is reset to 0, and both steps listed above must be repeated.

During the time in which in/out parameter IN must change from 6 to the value 9, output Q is set to 1. Otherwise, Q has a value of 0.

Every call of the "Fail-safe acknowledgment" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., ACK_OP_DB_1) or a multi-instance (e.g., ACK_OP_Instance_1) for the "Fail-safe acknowledgment" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

**Note**

A separate data area must be used for each call of ACK_OP. Each call can be processed only once in an F-runtime group cycle.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**WARNING**

The two acknowledgment steps must **not** be triggered by one single operation, for example by automatically storing them along with the time conditions in a program and using a function key to trigger them.
Having two separate acknowledgment steps also prevents erroneous triggering of an acknowledgment by your non-fail-safe HMI system. *(S013)*

**WARNING**

If you have HMI systems and F-CPUs that are interconnected and use the ACK_OP instruction for fail-safe acknowledgment, you need to ensure that the intended F-CPU will be addressed **before** you perform the two acknowledgment steps.

- To do this, store a network-wide* unique name for the F-CPU in a DB of your standard user program in each F-CPU.
- In your HMI system, set up a field from which you can read out the F-CPU name from the DB online before executing the two acknowledgment steps.
- Optional:

  in your HMI system, set up a field to permanently store the F-CPU name. Then, you can determine whether the intended F-CPU is being addressed by simply comparing the F-CPU name read out online with the permanently stored name.*(S014)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

**WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

**Note**

You can read out output Q by means of HMI systems or, if applicable, you can evaluate it in your standard user program.

You can provide the in/out IN with a separate memory word or DBW of a DB of the standard user program supply for each instance of the ACK_OP instruction.

## Description (S7-1200, S7-1500)

This instruction enables fail-safe acknowledgment from an HMI system. It allows, for example, reintegration of F-I/O to be controlled from the HMI system. Acknowledgment takes place in two steps:

- Input/output parameter IN changes to a value of 6 for exactly one cycle.
- Input/output parameter IN changes to the value at the ACK_ID input within a minute for exactly one cycle

Once the in/out parameter IN has changed to a value of 6, the instruction evaluates whether this parameter has changed to a value at the ACK_ID input after 1 second, at the earliest, or one minute, at the latest. Output OUT (output for acknowledgment) is then set to 1 for one cycle.

If an invalid value is input or if in/out parameter IN has not changed to the value at the ACK_ID input within one minute or the change occurred before one second has elapsed, then in/out parameter IN is reset to 0, and both steps listed above must be repeated.

During the time in which in/out parameter IN must change from 6 to the value at the ACK_ID input, output Q is set to 1. Otherwise, Q has a value of 0.

Every call of the "Fail-safe acknowledgment" instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., ACK_OP_DB_1) or a multi-instance (e.g., ACK_OP_Instance_1) for the "Fail-safe acknowledgment" instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

**Note**

A separate data area must be used for each call of ACK_OP. Each call can be processed only once in an F-runtime group cycle.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

**⚠ WARNING**

The two acknowledgment steps must **not** be triggered by one single operation, for example by automatically storing them along with the time conditions in a program and using a function key to trigger them.
Having two separate acknowledgment steps also prevents erroneous triggering of an acknowledgment by your non-fail-safe HMI system. *(S013)*

**WARNING**

If you have HMI systems and F-CPUs that are interconnected and use the ACK_OP instruction for fail-safe acknowledgment, you need to ensure that the intended F-CPU will be addressed **before** you perform the two acknowledgment steps.

Alternative 1:

- The value for each identifier of the acknowledgment (ACK_ID input; data type: INT) can be freely selected in the range from 9 to 30000, but must be unique network-wide* for all instances of the ACK_OP instruction.
  You must supply the ACK_ID input with constant values when calling the instruction. Direct read or write access in the associated instance DB is not permitted in the safety program!

Alternative 2:

- To do this, store a network-wide* unique name for the F-CPU in a DB of your standard user program in each F-CPU.
- In your HMI system, set up a field from which you can read out the F-CPU name from the DB online before executing the two acknowledgment steps.
- Optional:

  in your HMI system, set up a field to permanently store the F-CPU name. Then, you can determine whether the intended F-CPU is being addressed by simply comparing the F-CPU name read out online with the permanently stored name. *(S047)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

**WARNING**

When using an instruction with time processing, take the following timing imprecision sources into account when determining your response times:

- Known timing imprecision (based on standard systems) resulting from cyclic processing
- Timing imprecision resulting from the update time of the time base used in the instruction (see figure in section "Timing imprecision resulting from the update time of the time base used in the instruction")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (assigned) time value
  - For time values greater than or equal to 100 ms, a maximum of 2% of the (assigned) time value

You must choose the interval between two call times of an instruction with time processing in such a way that the required response times are achieved, taking into account the possible timing imprecision. *(S034)*

**Note**

You can read out output Q by means of HMI systems or, if applicable, you can evaluate it in your standard user program.

You need to provide the in/out IN with a separate memory word or DBW of a DB of the standard user program supply for each instance of the ACK_OP instruction.

## Parameters (S7-300, S7-400)

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| IN | InOut | INT | Input variable from HMI system |
| OUT | Output | BOOL | Output for acknowledgment |
| Q | Output | BOOL | Time status |

## Parameters (S7-1200, S7-1500)

The following table shows the parameters of the instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ACK_ID | Input | INT | Identifier of the acknowledgment (9 to 30000) |
| IN | InOut | INT | Input variable from HMI system |
| OUT | Output | BOOL | Output for acknowledgment |
| Q | Output | BOOL | Time status |

## Instruction versions

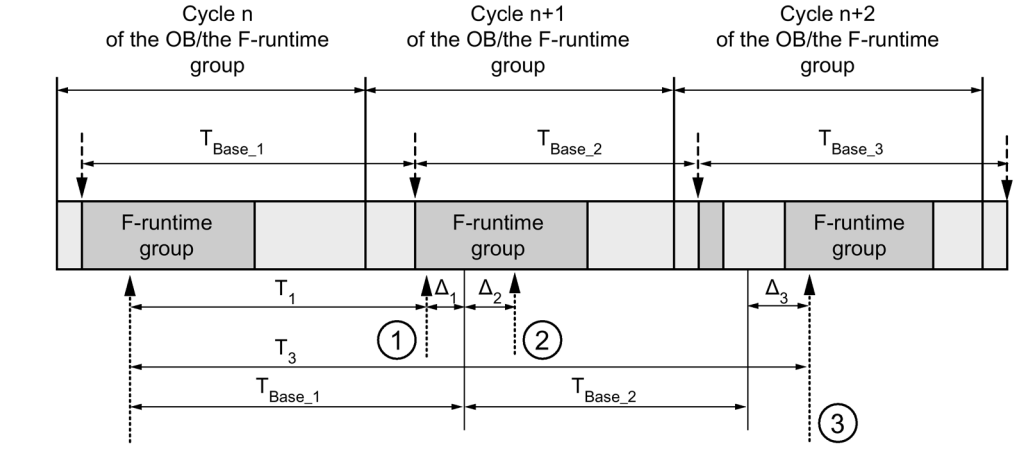
A number of versions are available for this instruction:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | x | — | x | These versions are identical in function to version V1.0 for S7-300/400 F-CPUs.<br>The input ACK_ID must also be taken into consideration for S7-1200/1500 F-CPUs. |
| 1.2 | x | x | x | |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

### Timing imprecision resulting from the update time of the time base used in the instruction:




① For the first call in cycle n+1, the call time of the instruction relative to the start of the F-runtime group is earlier than that in cycle n by the amount of $\Delta_1$, e.g. because parts of the safety program of the F-runtime group before the call time of the instruction in cycle n+1 are skipped. For the time update, the instruction takes account of time $T_{Base\_1}$ instead of the time $T_1$ that has actually elapsed in cycle n since the call.

② The instruction is called a second time in cycle n+1. This does not involve another time update (by $\Delta_2$).

③ For the call in cycle n+2, the call time of the instruction relative to the start of the F-runtime group is later than that in cycle n by the amount of $\Delta_3$, e.g. because the F-runtime group was interrupted by a higher priority interrupt before the call time of the instruction in cycle n+2. The instruction took into account time $T_{Base\_1} + T_{Base\_2}$ instead of the time $T_3$ that has actually elapsed in cycle n since the call. This would also be the case if no call occurred in cycle n+1.

## Example

An example how the instruction is used is available under Implementing User Acknowledgment in the Safety Program of the F-CPU of a DP Master or IO controller (Page 151).

## See also

Implementing user acknowledgment in the safety program of the F-CPU of a I-slave or I-device (S7-300, S7-400, S7-1500) (Page 156)

## 13.3.14 Additional instructions

### 13.3.14.1 OV: Get status bit OV (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

#### Description

You can use the "Get status bit OV" instruction to detect whether a number range overflow occurred in the last arithmetic instruction processed.

The "Get status bit OV" evaluation must be inserted in the network that follows the instruction that influences the OV. This network must not contain any jump labels.

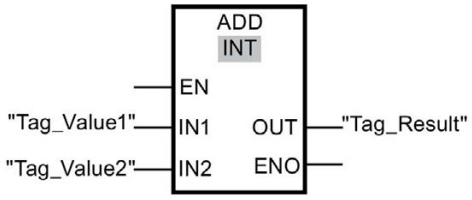
**Note**

The execution time of the OV-affecting instruction is extended when the "Get status bit OV " instruction is used (see also Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100)).

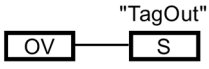
#### Example

The following example shows how the instruction works:

Network 1:




Network 2:




The value of the "Tag_Value1" operand is added to value of the Tag_Value2 operand. The result of the addition is stored in the "Tag_Result" operand.

If an overflow occurs during execution of the "Add" instruction, the status bit OV is set to "1". In network 2, following the query of the status bit OV, the "Set output" (S) instruction is executed and the "TagOut" operand is set.

## 13.3.15 Communication

### 13.3.15.1 PROFIBUS/PROFINET

#### SENDDP and RCVDP: Send and receive data via PROFIBUS DP/PROFINET IO (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400, S7-1500)

#### Introduction

You use the SENDDP and RCVDP instructions for fail-safe sending and receiving of data using:

- Safety-related master-master communication
- Safety-related master-master communication for S7 Distributed Safety
- Safety-related master-I-slave communication
- Safety-related I-slave-I-slave communication
- Safety-related IO controller-IO controller communication
- Safety-related IO controller-IO controller communication for S7 Distributed Safety
- Safety-related IO controller-I-device communication
- Safety-related IO controller-I-slave communication

#### Description

The SENDDP instruction sends 16 data elements of data type BOOL and 2 data elements of data type INT or one data element of the data type DINT (S7-1500) in a fail-safe manner to another F-CPU via PROFIBUS DP/PROFINET IO. The data can be received there by the related RCVDP instruction.

Every call of this instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., RCVDP_DB_1) for these instructions. Once it is created, the new data block can be found in the "STEP 7 Safety" folder in the project tree under "Program blocks > System blocks". For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

With the SENDDP instruction, the data to be sent (for example, outputs of other F-blocks/instructions) are available at input SD_BO_xx or SD_I_xx or SD_DI_00 as alternative.

With the RCVDP instruction, the data received are available at outputs RD_BO_xx and RD_I_xx or RD_DI_00 as alternative for additional processing by other F-blocks/instructions.

(S7-1500) At the DINTMODE input of the SENDDP instruction you specify if the data at the inputs SD_I_00 and SD_I_01 or the data at the input SD_DI_00 is sent.

The operating mode of the F-CPU with the SENDDP instruction is provided at output SENDMODE. If the F-CPU with the SENDDP instruction is in disabled safety mode, output SENDMODE = 1.

Communication between F-CPUs takes place in the background by means of a special safety protocol. You must define the communication relationship between a SENDDP instruction in one F-CPU and a RCVDP instruction in the other F-CPU by specifying an address relationship at the DP_DP_ID inputs of the SENDDP and RCVDP instructions. Associated SENDDP and RCVDP instructions are assigned the same value for DP_DP_ID.

**⚠ WARNING**

The value for each address relationship (DP_DP_ID input; data type: INT) is user-defined; however, it must be unique network-wide* for all safety-related communication connections. The uniqueness must be checked in the safety summary during acceptance of the safety program. You can find additional information in Correctness of the communication configuration (Page 316).

You must supply inputs DP_DP_ID and LADDR with constant values when calling the instruction. Direct read or write access to the associated instance DB to DP_DP_ID and LADDR is not permitted in the safety program. *(S016)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/3 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

**Note**

Within a safety program, you must assign a different start address (S7-300, S7-400) or HW identifier (S7-1200, S7-1500) for every call of the SENDDP and RCVDP instructions at input LADDR.

A separate instance DB must be used for each call of the SENDDP and RCVDP instructions. You must not declare and call these instructions as multi-instances.

The input and outputs of the RCVDP instruction cannot be initialized with temporary or static local data of the main safety block.

The inputs of the RCVDP instruction cannot be initialized with outputs (using fully qualified DB accesses) of a RCVDP or RCVS7 instruction called in an upstream network.

The RD_D_00 output must not be evaluated for DINTMODE = 0; the RD_I_xx outputs of the RCVDP instruction must not be evaluated for DINTMODE = 1.

(S7-1500) The outputs of the SENDDP and RCVDP instructions must not be supplied with tags from the standard user program. Exception: RET_DPRD, RET_DPWR and DIAG outputs.

You cannot use an actual parameter for an output of an RCVDP instruction, if it is already being used for an input of the same or another RCVDP or RCVS7 instruction.

The F-CPU can go to STOP if this is not observed. The cause of the diagnostic event is entered in the diagnostic buffer of the F-CPU. You can find additional information on the cause in the online help for diagnostic messages.

**Note**

You are not permitted to insert any SENDDP instructions between an instruction JMP or JMPN and the associated jump destination (jump label).

You cannot insert a RET instruction prior to a SENDDP instruction.

## SENDDP parameter

The following table shows the parameters of the SENDDP instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| SD_BO_00 | Input | BOOL | Send data BOOL 00 |
| ... | | | ... |
| SD_BO_15 | Input | BOOL | Send data BOOL 15 |
| SD_I_00 | Input | INT | Send data INT 00 |
| SD_I_01 | Input | INT | Send data INT 01 |
| SD_DI_00 | Input | DINT | (S7-1500)<br>(hidden)<br>Send data DINT 00 |
| DINTMODE | Input | DINT | (S7-1500)<br>(hidden)<br>0=SD_I_00 u. SD_I_01 are sent<br>1=SD_DI_00 is sent |
| DP_DP_ID | Input | INT | Network-wide unique value for the address relationship between a SENDDP and RCVDP instruction |
| TIMEOUT | Input | TIME | Monitoring time in ms for safety-related communication (see also Monitoring and response times (Page 661)) |
| LADDR | Input | INT | The start address (S7-300, S7-400) or HW identifier (S7-1200, S7-1500) of the address area/transfer area:<br>• Of DP/DP coupler for safety-related master-master communication<br>• For safety-related master-I-slave communication<br>• For safety-related I-slave-I-slave communication<br>• Of PN/PN coupler for safety-related IO controller-IO controller communication<br>• For safety-related IO controller-I-device communication<br>• For safety-related IO controller-I-slave communication |
| ERROR | Output | BOOL | 1=Communication error |
| SUBS_ON | Output | BOOL | 1=RCVDP outputs fail-safe values |
| RET_DPRD | Output | WORD | Error code RET_VAL of the DPRD_DAT instruction<br>(for a description of error codes, refer to the online help for DPRD_DAT instruction ("Extended instructions > Distributed I/O > Other".)) |

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| RET_DPWR | Output | WORD | Error code RET_VAL of the DPWR_DAT instruction (for a description of error codes, refer to the online help for DPWR_DAT instruction ("Extended instructions > Distributed I/O > Other".)) |
| DIAG | Output | BYTE | Service information |

## RCVDP parameter:

The following table shows the parameters of the RCVDP instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ACK_REI | Input | BOOL | 1=Acknowledgment for reintegration of send data following communication error |
| SUBBO_00 | Input | BOOL | Fail-safe value for receive data BOOL 00 |
| ... | | | ... |
| SUBBO_15 | Input | BOOL | Fail-safe value for receive data BOOL 15 |
| SUBI_00 | Input | INT | Fail-safe value for receive data INT 00 |
| SUBI_01 | Input | INT | Fail-safe value for receive data INT 01 |
| SUBDI_00 | Input | DINT | (S7-1500)<br>(hidden)<br>Fail-safe value for receive data DINT 00 |
| DP_DP_ID | Input | INT | Network-wide unique value for the address relationship between a SENDDP and RCVDP instruction |
| TIMEOUT | Input | TIME | Monitoring time in ms for safety-related communication<br>(see also Monitoring and response times (Page 661)) |
| LADDR | Input | INT | The start address (S7-300, S7-400) or HW identifier (S7-1200, S7-1500) of the address area/transfer area:<br>• Of DP/DP coupler for safety-related master-master communication<br>• For safety-related master-I-slave communication<br>• For safety-related I-slave-I-slave communication<br>• Of PN/PN coupler for safety-related IO controller-IO controller communication<br>• For safety-related IO controller-I-device communication<br>• For safety-related IO controller-I-slave communication |
| ERROR | Output | BOOL | 1=Communication error |
| SUBS_ON | Output | BOOL | 1=Fail-safe values are output |
| ACK_REQ | Output | BOOL | 1=Acknowledgment for reintegration of send data required |
| SENDMODE | Output | BOOL | 1=F-CPU with SENDDP instruction in disabled safety mode |
| RD_BO_00 | Output | BOOL | Receive data BOOL 00 |
| ... | | | ... |
| RD_BO_15 | Output | BOOL | Receive data BOOL 15 |
| RD_I_00 | Output | INT | Receive data INT 00 |
| RD_I_01 | Output | INT | Receive data INT 01 |

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| RD_DI_00 | Output | DINT | (S7-1500)<br>(hidden)<br>Receive data DINT 00 |
| RET_DPRD | Output | WORD | Error code RET_VAL of the DPRD_DAT instruction (for a description of error codes, refer to the online help for DPRD_DAT instruction ("Extended instructions > Distributed I/O > Other".)) |
| RET_DPWR | Output | WORD | Error code RET_VAL of the DPWR_DAT instruction (for a description of error codes, refer to the online help for DPWR_DAT instruction ("Extended instructions > Distributed I/O > Other".)) |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for these instructions:

| Version | S7-300/400 | S7-1200 | S7-1500 | Function |
|---|---|---|---|---|
| 1.0 | x | — | — | When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.0 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.1 | — | — | — | This version is invalid. |
| 1.2 | x | — | x | This version has identical functions to version V1.0. |
| 1.3 | x | — | x | S7-300/400: This version has identical functions to version V1.0.<br>S7-1500: Instead of 2 data of data type INT, one data of data type DINT can be sent/received as alternative. Otherwise identical function as version V1.0. |

When a new F-CPU is created with *STEP 7 Safety*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Placement

The RCVDP instruction must be inserted at the start of the main safety block and the SENDDP instruction at the end.

## Startup characteristics

After the sending and receiving F-systems are started up, communication must be established between the connection partners for the fist time (SENDDP and RCVDP instructions). During this time, the receiver (RCVDP instruction) outputs the fail-safe values present at its inputs SUBBO_xx and SUBI_xx or alternatively SUBDI_00.

The SENDDP and RCVDP instructions signal this at output SUBS_ON with 1. Output SENDMODE has a default of 0 and is not updated, as long as output SUBS_ON = 1.

## Behavior in the event of communication errors

If a communication error occurs, for example, due to a signature error (CRC) or when monitoring time TIMEOUT expires, outputs ERROR and SUBS_ON = 1 are set. The receiver (RCVDP instruction) then outputs the fail-safe values assigned at its SUBBO_xx and SUBI_xx or alternatively SUBDI_00 inputs. Output SENDMODE is not updated while output SUBS_ON = 1.

The send data of the SENDDP instruction present at inputs SD_BO_xx and SD_I_xx, SD_DI_00 as alternative, are only output again when communication errors are no longer detected (ACK_REQ = 1) and you acknowledge (Page 151) the RCVDP instruction with a positive edge at input ACK_REI.

**⚠ WARNING**

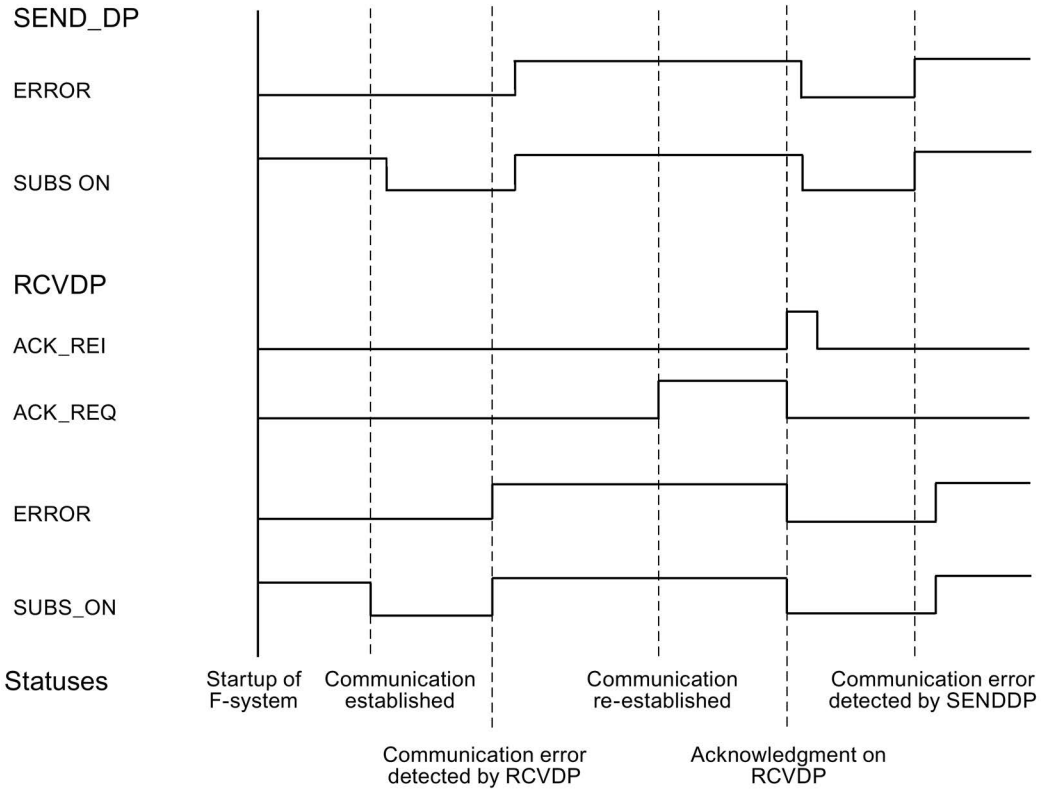
For the user acknowledgment, you must interconnect input ACK_REI with a signal generated by the operator input.

An interconnection with an automatically generated signal is not permitted. *(S040)*

Note that output ERROR (1=communication error) for a communication error will not be set unless communication between the connection partners (SENDDP and RCVDP instructions) has been previously established. If communication cannot be established after startup of the sending and receiving F-Systems, check the configuration of the safety-related CPU-CPU communication, the parameter assignment of the SENDDP and the RCVDP instructions, and the bus connection. You also obtain information on possible error causes by evaluating outputs RET_DPRD and RETDP_WR.

In general, always evaluate RET_DPRD and RETDP_WR, since it is possible that only one of the two outputs will contain error information.

## Timing diagrams SENDDP/RCVDP




## Output DIAG

In addition, non-fail-safe information about the type of communication errors that occurred is provided at output DIAG of the SENDDP and RCVDP instructions for service purposes.

You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until you acknowledge the errors at input ACK_REI of the RCVDP instruction.

## Structure of DIAG of the SENDDP/RCVDP instruction

| Bit no. | Assignment | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Reserved | — | — |
| Bit 1 | Reserved | — | — |
| Bit 2 | Reserved | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Timeout of SENDDP/RCVDP detected | Interference in bus connection to partner F-CPU. | Check bus connection and ensure that no external fault sources are present. |
| | | Monitoring time setting for F-CPU and partner F-CPU is too low. | Check assigned monitoring time TIMEOUT for SENDDP and RCVDP of both F-CPUs. If possible, set a higher value. Recompile safety program |
| | | DP/DP coupler or PN/PN coupler configuration is invalid. | Check DP/DP coupler or PN/PN coupler configuration. |
| | | Internal fault of DP/DP coupler or PN/PN coupler | Replace DP/DP coupler or PN/PN coupler |
| | | CP in STOP mode, or internal fault in CP | Switch CP to RUN mode, check diagnostic buffer of CP, and replace CP, if necessary |
| | | F-CPU/partner F-CPU in STOP mode, or internal fault in F-CPU/partner F-CPU | Switch F-CPUs to RUN mode, check diagnostic buffer of F-CPUs, and replace F-CPUs, if necessary |
| Bit 5 | Sequence number error of SENDDP/RCVDP detected | See description for bit 4 | See description for bit 4 |
| Bit 6 | CRC-error of SENDDP/RCVDP detected | See description for bit 4 | See description for bit 4 |
| Bit 7 | Reserved | — | — |

## See also

Configuring and programming communication (S7-300, S7-400) (Page 163)

Safety-related IO controller-IO controller communication (Page 166)

Safety-related master-master communication (Page 175)

Safety-related communication between IO controller and I-device (Page 185)

Safety-related master-I-slave communication (Page 192)

Safety-related IO controller-I-slave communication (Page 211)

Communication with S7 Distributed Safety via PN/PN coupler (IO controller-IO controller communication) (Page 220)

Communication with S7 Distributed Safety via DP/DP coupler (master-master communication) (Page 221)

### 13.3.15.2 S7 communication

## SENDS7 and RCVS7: Communication via S7 connections (STEP 7 Safety Advanced V13 SP1) (S7-300, S7-400)

### Introduction

You use the SENDS7 and RCVS7 instructions for fail-safe sending and receiving of data using S7 connections.

**Note**

In *STEP 7 Safety Advanced*, S7 connections are generally permitted over Industrial Ethernet only.

Safety-related communication via S7 connections is possible from and to F-CPUs with PROFINET interface or S7-400 F-CPUs with PROFINET-capable CPs. See also Safety-related communication via S7 connections (Page 212).

### Description

The SENDS7 instruction sends the send data contained in an F-communication DB to the F-communication DB of the associated RCVS7 instruction of another F-CPU in a fail-safe manner using an S7 connection.

Every call of this instruction must be assigned a data area in which the instruction data are stored. The "Call options" dialog is automatically opened when the instruction is inserted in the program for this reason. There you can create a data block (single instance) (e.g., SENDS7_DB_1) or a multi-instance (e.g., SENDS7_Instance_1) for this instruction. Once it is created, you can find the new data block in the project tree in the "STEP 7 Safety" folder under "Program blocks > System blocks" or the multi-instance as a local tag in the "Static" section of the block interface. For more information, refer to the help on *STEP 7*.

Enable input "EN" and enable output "ENO" cannot be connected. The instruction is therefore always executed (regardless of the signal state at enable input "EN").

Information on the F-communication DB is contained in "Safety-related communication via S7 connections (Page 212)".

An F-communication DB is an F-DB for safety-related CPU-CPU communication with special properties. You must specify the numbers of the F-communication DBs at inputs SEND_DB and RCV_DB of instructions SENDS7 and RCVS7.

The operating mode of the F-CPU with the SENDS7 instruction is provided at output SENDMODE of the RCVS7 instruction. If the F-CPU with the SENDS7 instruction is in disabled safety mode, then output SENDMODE = 1.

To reduce the bus load, you can temporarily shut down communication between the F-CPUs at input EN_SEND of the SENDS7 instruction. To do so, supply input EN_SEND with "0" (default = "1"). In this case, send data are no longer sent to the F-communication DB of the associated RCVS7 instruction, and the receiver provides fail-safe values for this period of time (initial values in its F-communication DB). If communication was already established between the partners, a communication error is detected.

You must specify the local ID - from the perspective of the F-CPU - of the S7 connection (from the connection table in the network view) at input ID of the SENDS7 instruction (see also Configuring (Page 37)).

Communication between F-CPUs takes place in the background by means of a special safety protocol. You must define a communication relationship between an SENDS7 instruction in one F-CPU and a communication relationship between an RCVS7 instruction and the other F-CPU by assigning an odd number at input R_ID (of the SENDS7 and RCVS7 instructions). Associated SENDS7 and RCVS7 instructions receive the same value for R_ID.

**⚠ WARNING**

The value for each address relationship (R_ID input; data type: DWORD) is user-defined; however, it must be an odd number and unique network-wide* for all safety-related communication connections. The value R_ID + 1 is internally assigned and must not be used.

You must supply inputs ID and R_ID with constant values when calling the instruction. Direct read or write access to the associated instance DB is not permitted in the safety program! *(S020)*

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

**Note**

A separate instance DB must be used for each call of the SENDS7 and RCVS7 instructions within a safety program. You must not declare and call these instructions as multi-instances.

The input and outputs of the RCVS7 instruction cannot be initialized with temporary or static local data of the main safety block.

The inputs of the RCVS7 instruction cannot be initialized with outputs (using fully qualified DB accesses) of a RCVS7 or RCVDP instruction called in an upstream network.

You cannot use an actual parameter for an output of an RCVS7 instruction, if it is already being used for an input of the same or another RCVS7 or RCVDP instruction. The F-CPU can go to STOP if this is not observed. A diagnostic event is entered in the diagnostic buffer of the F-CPU.

**Note**

You must not program any SENDS7 instruction between a JMP or JMPN instruction and the associated destination network of the JMP or JMPN instruction.

You cannot program a RET instruction prior to a SENDS7 instruction.

## SENDS7 parameter

The following table shows the parameters of the SENDS7 instruction:

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| SEND_DB | Input | BLOCK_DB | Number of F-communication DB |
| TIMEOUT | Input | TIME | Monitoring time in ms for safety-related communication (see also Monitoring and response times (Page 661)) |
| EN_SEND | Input | BOOL | 1= Send enable |
| ID | Input | WORD | Local ID of the S7 connection |
| R_ID | Input | DWORD | Network-wide unique value for an address relationship between a SENDS7 instruction and a RCVS7 instruction |
| ERROR | Output | BOOL | 1=Communication error |
| SUBS_ON | Output | BOOL | 1=Receiving block outputs fail-safe values |
| STAT_RCV | Output | WORD | Status parameter STATUS of the URCV instruction (You can find a description of error codes in the online help for the URCV instruction ("Communication > S7 Communication")) |
| STAT_SND | Output | WORD | Status parameter STATUS of the USEND instruction (You can find a description of error codes in the online help for the USEND instruction ("Communication > S7 Communication")) |
| DIAG | Output | BYTE | Service information |

## RCVS7 parameter

The following table shows the parameters of the RCVS7 instruction.

| Parameter | Declaration | Data type | Description |
|---|---|---|---|
| ACK_REI | Input | BOOL | Acknowledgment for reintegration of send data after communication error |
| RCV_DB | Input | BLOCK_DB | Number of F-communication DB |
| TIMEOUT | Input | TIME | Monitoring time in ms for safety-related communication (see also Monitoring and response times (Page 661)) |
| ID | Input | WORD | Local ID of the S7 connection |
| R_ID | Input | DWORD | Network-wide unique value for an address relationship between a SENDS7 instruction and a RCVS7 instruction |
| ERROR | Output | BOOL | 1=Communication error |
| SUBS_ON | Output | BOOL | 1=Fail-safe values are output |
| ACK_REQ | Output | BOOL | 1=Acknowledgment for reintegration of send data required |
| SENDMODE | Output | BOOL | 1=F-CPU with the SENDS7 instruction in disabled safety mode |
| STAT_RCV | Output | WORD | Status parameter STATUS of the URCV instruction (You can find a description of error codes in the online help for the URCV instruction ("Communication > S7 Communication")) |
| STAT_SND | Output | WORD | Status parameter STATUS of the USEND instruction (You can find a description of error codes in the online help for the USEND instruction ("Communication > S7 Communication")) |
| DIAG | Output | BYTE | Service information |

## Instruction versions

A number of versions are available for these instructions:

| Version | S7-300/400 | S7-1500 | Function |
|---|---|---|---|
| 1.0 | x | — | |
| 1.1 | x | — | This version has identical functions to version V1.0.<br>It also supports later versions of internally called instructions.<br>When projects that were created with *S7 Distributed Safety V5.4 SP5* are migrated, version 1.1 of the instruction is used automatically.<br>If you want to compile a migrated safety program with *STEP 7 Safety Advanced* for the first time, we recommend that you first update to the latest available version of the instruction. |
| 1.2 | x | — | This version has identical functions to version V1.0/1.1.<br>It also supports later versions of internally called instructions. |

When a new F-CPU is created with *STEP 7 Safety Advanced*, the latest available version for the F-CPU created is automatically preset.

For more information on the use of instruction versions, refer to the help on *STEP 7* under "Using instruction versions".

## Startup characteristics

After the sending and receiving F-systems are started up, communication must be established between the connection partners for the fist time (SENDS7 and RCVS7 instructions). The receiver (RCVS7 instruction) provides fail-safe values for this time period (initial values in its F-communication DB).

The SENDS7 and RCVS7 instructions signal this at output SUBS_ON with 1. Output SENDMODE (RCVS7 instruction) has a default of 0 and is not updated, as long as output SUBS_ON = 1.

## Behavior in the event of communication errors

If a communication error occurs, for example, due to a signature error (CRC) or when monitoring time TIMEOUT expires, outputs ERROR and SUBS_ON = 1 are set. The receiver (RCVS7 instruction) then provides fail-safe values (initial values in its F-communication DB). Output SENDMODE is not updated while output SUBS_ON = 1.

The send data present in the F-communication DB (SENDS7 instruction) are not output before the communication error is no longer detected ((ACK_REQ = 1)) and you acknowledge (Page 151) with a positive edge at input ACK_REI of the RCVS7 instruction.

**⚠ WARNING**

For the user acknowledgment, you must interconnect input ACK_REI with a signal generated by the operator input.

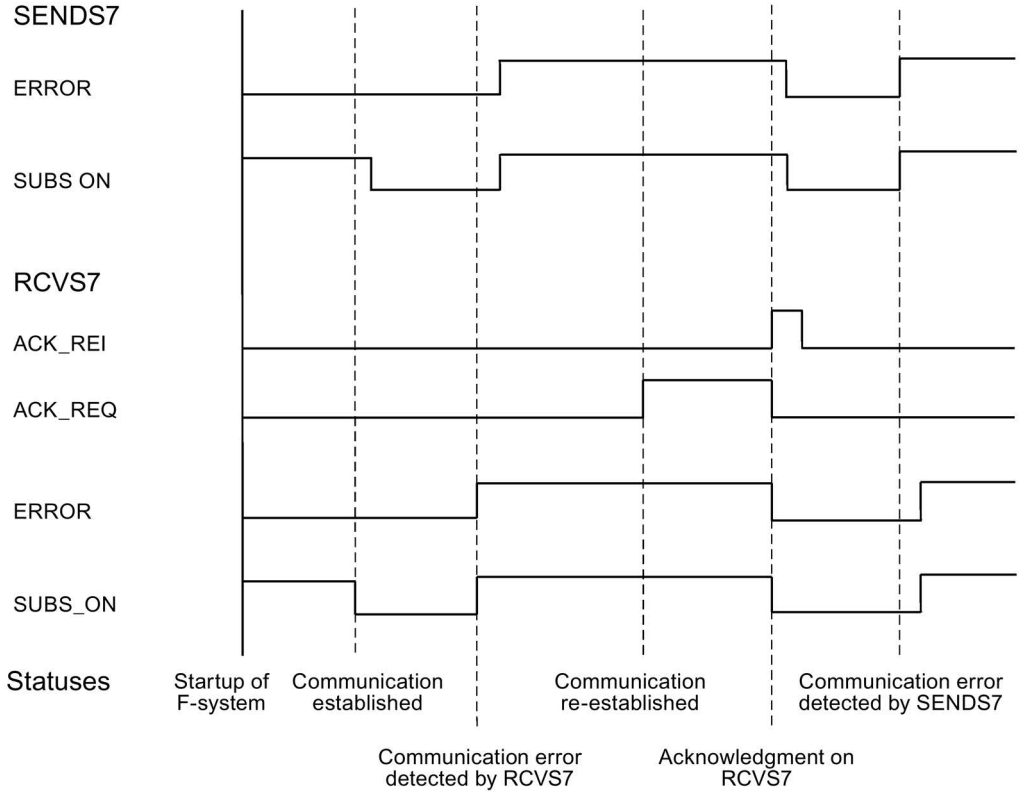
An interconnection with an automatically generated signal is not permitted. *(S040)*

Note that output ERROR (1=communication error) will be set for the first time on a communication error if communication has already been established between the connection partners (SENDS7 and RCVS7 instructions). If communication cannot be established after startup of the sending and receiving F-Systems, check the configuration of the safety-related CPU-CPU communication, the parameter assignment of the SENDS7 and the RCVS7 instructions, and the bus connection. You can also receive information on possible error causes by evaluating the STAT_RCV and STAT_SND outputs.

In general, always evaluate STAT_RCV and STAT_SND, since it is possible that only one of the two outputs will contain error information.

If one of the DIAG bits is set at output DIAG, also check whether the length and structure of the associated F-communication DB on both the sending and receiving ends match.

## Timing diagrams SENDS7 and RCVS7




## Output DIAG

Non-fail-safe information on the type of communication errors that have occurred is made available at output DIAG for service purposes. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until you acknowledge them at input ACK_REI of the associated RCVS7 instruction.

## Structure of DIAG

| Bit no. | Assignment of SENDS7 and RCVS7 | Possible error causes | Remedies |
|---|---|---|---|
| Bit 0 | Reserved | — | — |
| Bit 1 | Reserved | — | — |
| Bit 2 | Reserved | — | — |
| Bit 3 | Reserved | — | — |
| Bit 4 | Timeout detected by SENDS7 and RCVS7 | Fault in bus connection to partner F-CPU | Check bus connection and ensure that no external fault sources are present. |
| | | Monitoring time setting for F-CPU and partner F-CPU is too low | Check assigned monitoring time TIMEOUT for SENDS7 and RCVS7 of both F-CPUs. If possible, set a higher value. Recompile safety program |
| | | CPs in STOP mode, or internal fault in CPs | • Switch CPs to RUN mode<br>• Check diagnostic buffer of CPs<br>• Replace CPs, if necessary |
| | | F-CPU/partner F-CPU in STOP mode, or internal fault in F-CPU/partner F-CPU | • Switch F-CPUs to RUN mode<br>• Check diagnostic buffer of F-CPUs<br>• Replace F-CPUs, if necessary |
| | | Communication was shut down with EN_SEND = 0. | Enable communication again at the associated SENDS7 with EN_SEND = 1 |
| | | S7 connection has changed, the IP address of the CP has changed, for example | Recompile the safety programs and download them to the F-CPUs |
| Bit 5 | Sequence number error detected by SENDS7 and RCVS7 | See description for bit 4 | See description for bit 4 |
| Bit 6 | CRC-error detected by SENDS7 and RCVS7 | See description for bit 4 | See description for bit 4 |
| Bit 7 | RCVS7:<br>Communication cannot be established | Configuration of the safety-related CPU-CPU communication is incorrect, parameter assignment of the SENDS7 and RCVS7 instructions is incorrect<br>See also description for Bit 4 | Check configuration of the safety-related CPU-CPU communication, the parameter assignment of the SENDS7 and the RCVS7 instructions is incorrect<br>See also description for Bit 4 |
| | SENDS7:<br>Reserved | — | — |

# Monitoring and response times

# A

## Introduction

In the following, you will learn:

- Which F-specific monitoring times you must configure
- Which rules must be followed when specifying monitoring times
- Where you enter the F-specific monitoring times
- Which rules must be followed with regard to the maximum response time of a safety function

## Support for calculations

An Excel file is available on the Internet (http://support.automation.siemens.com/WW/view/en/49368678/133100) to assist you in calculating approximately the runtimes of the F-runtime groups, the minimum F-specific monitoring times, and the maximum response times of your F-System.

## Additional information

The monitoring and response times for the standard part are calculated in SIMATIC Safety in exactly the same way as for standard S7-300, S7-400, S7-1200 and S7-1500 automation systems and are not addressed here. For a description of this calculation, refer to the *hardware manuals for the CPUs*.

# A.1 Configuring monitoring times

## Monitoring times to be configured

You must configure the following monitoring times:

| Monitoring... | Setting... | Parameters | See |
|---|---|---|---|
| F-cycle time or cycle time warning limit of the F-runtime groups that contain the safety program | in *Safety Administration Editor*:<br>• Dialog for definition of an F-runtime group | Maximum cycle time of the F-runtime group | • Procedure for defining an F-runtime group (S7-300, S7-400) (Page 104)<br>• Procedure for defining an F-runtime group (S7-1200, S7-1500) (Page 107) |
| of the safety-related communication between F-CPU and F-I/O via PROFIsafe (PROFIsafe monitoring time) | in the *hardware and network editor*:<br>• Centrally when configuring the F-CPU; properties of the F-CPU; or<br>• when configuring the F-I/O; properties of the F-I/O | F-monitoring time<br>F_WD_TIME | • Configuring an F-CPU (Page 42)<br>• Configuring F-I/O (Page 47)<br>• Peculiarities when configuring fail-safe GSD based DP slaves and fail-safe GSD based I/O devices (Page 61) |
| of the safety-related CPU-CPU communication | Parameter "TIMEOUT" of the instructions:<br>• SENDDP; RCVDP; SENDS7; RCVS7 | TIMEOUT | • Communication (Page 646) |

You do not have to configure the monitoring time for safety-related communication between F-runtime groups.

### Rules for configuring monitoring times

When configuring monitoring times, you must take into account the availability as well as the safety of the F-system as follows:

- Availability: To ensure that the time monitoring is not triggered when there is no error, the monitoring times selected must be sufficiently long.
- Safety: To ensure that the process safety time is not exceeded for the process, the monitoring times selected must be sufficiently short.

**⚠ WARNING**

It can only be ensured (from a fail-safe standpoint) that a signal state to be transferred will be acquired at the sender end and transferred to the receiver if the signal level is pending for at least as long as the assigned F-monitoring time. *(S018)*

### General procedure for configuring monitoring times

Use the following procedure for configuring monitoring times:

1. Configure the standard system.

   Refer to the applicable *hardware manuals* and *Help on STEP 7* for the necessary information.

2. Configure the specific monitoring times of the F-System with respect to availability. You use the Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100) to calculate the approximate minimum monitoring time.

3. Use the Excel file for response time calculation to calculate the maximum response time and then verify that the process safety time of the process is not exceeded. If necessary, you must reduce the specific monitoring times of the F-System.

## A.1.1 Minimum monitoring time for the F-runtime group cycle time

### Parameter "Maximum cycle time of the F-runtime group"

You configure the monitoring time for the F-runtime group cycle time in the *Safety Administration Editor* in the work area for definition of the F-runtime group (Page 102).

The specified maximum cycle time of the F-runtime group must be high enough to prevent F-runtime group cycle time monitoring from being triggered when there are no pending faults and causing the F-CPU to go to STOP. For S7-1200/1500 F-CPUs, you can use the "F-runtime group cycle time warning limit (Page 107)" and the TCYC_CURR (Page 115) and TCYC_LONG (Page 115) tags for dimensioning.

Use the Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100) to determine the minimum monitoring time for the F-runtime group cycle time. Note also the comments in the Excel file.

## A.1.2 Minimum monitoring time for safety-related communication between F-CPU and F-I/O

### Parameter "F-monitoring time"

You have two options for configuring the monitoring time of safety-related communication between the F-CPU and F-I/O:

- Centrally in the *hardware and network editor* during parameter assignment of the F-CPU (Page 42); in the properties of the F-CPU, or
- During parameter assignment of the F-I/O (Page 47) in the *hardware and network editor*; in the properties of the F-I/O

### "F-monitoring time" = PROFIsafe monitoring time $T_{PSTO}$

The specified PROFIsafe monitoring time $T_{PSTO}$ must be high enough to prevent tripping the F-cycle time monitoring when no faults are present.

Use the Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100) available for SIMATIC Safety to calculate the minimum monitoring time for safety-related communication between the F-CPU and F-I/O.

Note also the comments in the Excel file.

### Check to determine whether configured PROFIsafe monitoring time is too short

**Note**

During commissioning of the F-system, you can perform a check while safety mode is active to determine whether the configured PROFIsafe monitoring time is too short.

This check of the PROFIsafe monitoring time is useful if you want to ensure that the configured monitoring time exceeds the minimum monitoring time by a sufficient amount. In this way, you can avoid the possible occurrence of sporadic monitoring time errors.

**Procedure:**

1. Insert an F-I/O (one that will not be needed later for system operation).
2. Assign a shorter monitoring time for this F-I/O than for the F-I/O of the system.
3. If the additional F-I/O fails and the "Monitoring time for safety message frame exceeded" diagnostic is signaled, you have fallen below the minimum possible PROFIsafe monitoring time.
4. Increase the monitoring time for the additional F-I/O just to the point where it no longer fails. This monitoring time corresponds approximately to the minimum possible monitoring time.

**Conditions:**

The F-I/O to be inserted additionally and the F-I/O whose PROFIsafe monitoring time is to be checked must have the following properties in common:

- They must be inserted in the same rack
- They must be nodes in the same subnet

**Tip:**

It may be useful to leave the additional F-I/O in place for systems that will be modified or expanded during operation after commissioning. This F-I/O will then provide an early warning in the event of changes in the time behavior, enabling you to avoid a process shutdown triggered by the F-I/O in the process.

## A.1.3 Minimum monitoring time of safety-related CPU-CPU communication

### Parameter TIMEOUT at SENDDP and RCVDP or SENDS7 and RCVS7

The time monitoring is performed in the SENDDP and RCVDP (Page 646) or SENDS7 and RCVS7 (Page 654) instructions of the communication partner. You must assign the time monitoring with identical monitoring time for both instructions in the TIMEOUT parameter.

The specified monitoring time TIMEOUT must be high enough to prevent tripping the monitoring when no faults are present.

Use the Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100) available for SIMATIC Safety to determine the minimum value for TIMEOUT.

Note also the comments in the Excel file.

### A.1.4 Monitoring time for safety-related communication between F-runtime groups (S7-300, S7-400)

The monitoring time for safety-related communication between F-runtime groups is determined automatically from the values for the "Maximum cycle time of F-runtime group" (work area for Definition of the F-runtime group (Page 102) in the *Safety Administration Editor*).

Monitoring time = (maximum cycle time of the 1st F-runtime group) + (maximum cycle time of the 2nd F-runtime group)

## A.2 Response Times of Safety Functions

### Definition of response time

The response time is the time from detection of an input signal until the linked output signal changes.

### Fluctuation range

The actual response time lies between a minimum response time and maximum response time. You must always take the maximum response time into account in your system configuration.

### Rules for maximum response time of a safety function

The maximum response time of a safety function must be less than the process safety time of the process.

### Definition of process safety time of a process

The process safety time of a process is the time interval within which the process can be left on its own without causing injury to operating personnel or damage to the environment.

Within the process safety time, any type of F-system process control is tolerated. That is, during this time, the F-system can control its process incorrectly or it can even execute no control at all. The process safety time of a process depends on the process type and must be determined on a case-by-case basis.

### Procedure for response time calculation

The Excel file for response time calculation (http://support.automation.siemens.com/WW/view/en/49368678/133100) is available for calculating the maximum response time of a safety function.

Use the Excel file to calculate the approximate maximum response time of the safety function and then check that the process safety time of the process is not exceeded.

If necessary, you must reduce the specific monitoring times of the F-system (see Minimum monitoring time for safety-related communication between F-CPU and F-I/O (Page 664)).

# Checklist

# B

## Life cycle of fail-safe automation systems

The following table contains a checklist summarizing all activities in the life cycle of a fail-safe SIMATIC Safety system, including requirements and rules that must be observed for each activity.

## Checklist

Legend:

- Stand-alone section references refer to this documentation.
- *"F-SMs Manual"* refers to the Automation System S7-300, ET 200M Distributed I/O System, Fail-Safe Signal Modules (http://support.automation.siemens.com/WW/view/en/19026151) manual.
- *"F-Modules Manual"*" refers to the ET 200S Distributed I/O System, Fail-Safe Modules (http://support.automation.siemens.com/WW/view/en/27235629) manual.
- *"ET 200eco Manual"* refers to the ET 200eco Distributed I/O Station, Fail-Safe I/O Module (http://support.automation.siemens.com/WW/view/en/19033850) manual.
- *"ET 200pro Manual"* refers to the ET 200pro Distributed I/O System, Fail-Safe I/O Module (http://support.automation.siemens.com/WW/view/en/22098524) manual.
- *"ET 200iSP Manual"* refers to the ET 200iSP Distributed I/O Device, Fail-Safe Modules (http://support.automation.siemens.com/WW/view/en/47357221) manual.
- *"ET 200SP Manual*" refers to the ET 200SP Distributed I/O System (http://support.automation.siemens.com/WW/view/en/58649293) manual
- *"ET 200MP Manual*" refers to the ET 200MP Distributed I/O System (http://support.automation.siemens.com/WW/view/en/57251728/133300) manual
- *"ET 200SP Modules Manual"* refers to the device manuals for F-Modules of the ET 200SP Distributed I/O System (http://support.automation.siemens.com/WW/view/en/55679227/133300)

| Phase | Note the following | Reference | Check |
|---|---|---|---|
| **Planning** | | | |
| Requirement: "Safety requirements specification" available for the intended application | Process-dependent | — | |
| Specification of system architecture | Process-dependent | — | |
| Assignment of functions and subfunctions to system components | Process-dependent | under Product Overview (Page 21) | |
| Selection of sensors and actuators | Requirements for actuators | *F-SMs Manual*, section 6.5;<br>*F-Modules Manual*, section 4.5;<br>*ET 200eco Manual*, section 5.5<br>*ET 200pro Manual*, section 4.4<br>*ET 200S Manual*, section 4.5<br>*ET 200SP manual*, section 5.2.2<br>*ET 200MP manual*, section 5.2.2 | |
| Specification of required safety properties for individual components | IEC 61508:2010 | — | |
| **Configuration** | | | |
| Installing the optional package | Requirements for installation | under Installing/uninstalling the STEP 7 Safety Basic V13 SP1 optional package (Page 27) orInstalling/uninstalling the STEP 7 Safety Advanced V13 SP1 optional package (Page 28) | |
| Selection of S7 components | Descriptions of configuration | under Product Overview (Page 21);<br>*F-SMs Manual*, section 3;<br>*F-Modules Manual*, section 3;<br>*ET 200eco Manual*, section 3<br>*ET 200pro Manual*, section 2<br>*ET 200iSP Manual*, section 3<br>*ET 200SP Modules Manual*, section 3<br>*ET 200MP Modules Manual*, section 3 | |
| Configuration of hardware | • Description of F-systems<br>• Verification of utilized hardware components based on Annex 1 of Report in the Certificate | under Configuring (Page 37);<br>Annex 1 of Report on the Certificate | |

| Phase | Note the following | Reference | Check |
|---|---|---|---|
| Configuration of F-CPU | • Protection level, "Write protection for F-blocks" (S7-300, S7-400)<br>• Protection level, at least "Full access" (S7-1200, S7-1500)<br>• Password<br>• F-capability enabled<br>• Definition/setting of F-specific parameters<br>• Cycle time for the F-runtime group in which the safety program is to be executed, defined in accordance with the requirements and safety regulations - same as with standard system | under Configuring an F-CPU (Page 42);<br>*Standard S7-300;*<br>*Standard S7-400;*<br>*S7-1200 standard;*<br>*S7-1500 standard;*<br>under Monitoring and response times (Page 661) | |
| Configuration of F-I/O | • Settings for safety mode<br>• Setting of passivation type<br>• Configuring monitoring times<br>• Defining type of sensor interconnection/evaluation<br>• Defining diagnostic behavior<br>• Special F-parameters<br>• Assigning symbolic names<br>• Unique PROFIsafe address | under Configuring F-I/O (Page 47) orPeculiarities when configuring fail-safe GSD based DP slaves and fail-safe GSD based I/O devices (Page 61)<br>under Monitoring and response times (Page 661);<br>*F-SMs Manual*, sections 3, 9, 10;<br>*F-Modules Manual*, sections 2.4, 7;<br>*ET 200eco Manual*, sections 3, 8;<br>*ET 200pro Manual,* Sections 2.4, 8;<br>*ET 200iSP Manual,* Sections 2.4, 7, 8<br>*ET 200SP Modules Manual*, section 4<br>*ET 200MP Modules Manual*, section 4 | |

| Phase | Note the following | Reference | Check |
|---|---|---|---|
| **Programming** | | | |
| Defining program design and structure | • Follow warnings and notes on programming | under Overview of Programming (Page 84), Program structure of the safety program (S7-300, S7-400) (Page 85), Program structure of the safety program (S7-1200, S7-1500) (Page 87); Programming startup protection (Page 121); | |
| Creating the F-runtime groups | • Assignment of F-FB or F-FC as main safety block to the calling block (S7-300, S7-400) or F-OB (S7-1200, S7-1500)<br>• Setting maximum cycle time for the F-runtime group in accordance with requirements (dependent on process and safety regulations)<br>• Creating DB for F-runtime group communication<br>• (S7-300, S7-400) Call of main safety blocks directly in OBs (e.g. OB 35), FBs, or FCs<br>• (S7-1200, S7-1500) Call of the main safety block from the F-OB | under Defining F-Runtime Groups (Page 102)<br>under Monitoring and response times (Page 661) | |
| Creating/inserting the F-blocks | • Generating, editing, and saving F-FBs, F-FCs, and F-DBs in accordance with the requirements of the program structure<br>• Description:<br>– F-I/O access<br>– Passivation and reintegration of F-I/O<br>– Inserting F-blocks from global libraries<br>– Safety-related CPU-CPU communication<br>– Communication with the standard user program | under Creating F-blocks in FBD / LAD (Page 118)<br>under F-I/O access (Page 122)<br>under Implementation of user acknowledgment (Page 151)<br>under Using libraries (Page 120)<br>under Configuring and programming communication (S7-300, S7-400) (Page 163) and Configuring and programming communication (S7-1500) (Page 226)<br>under Data exchange between standard user program and safety program (Page 159) | |
| Compiling the safety program | — | under Compiling the safety program (Page 265) | |
| Implementing safety program call (S7-300, S7-400) | Check whether the main safety block is called directly in OBs (e.g., OB 35), FBs, or FCs. | under Defining F-Runtime Groups (Page 102) | |

| Phase | Note the following | Reference | Check |
|---|---|---|---|
| **Installation** | | | |
| Hardware configuration | Description of<br>• Installation<br>• Wiring | under Overview of Configuration (Page 37); Particularities for configuring the F-System (Page 41);<br>*F-SMs Manual*, sections 5, 6;<br>*F-Modules Manual*, sections 3, 4;<br>*ET 200eco Manual*, sections 3, 4;<br>*ET 200pro Manual,* Sections 2, 3;<br>*ET 200iSP Manual,* sections 3 and 4;<br>*ET 200SP Manual*, sections 4 and 5<br>*ET 200MP Manual*, sections 4 and 5 | |
| **Commissioning, Testing** | | | |
| Switching on | Description of commissioning – same as in standard | Standard S7-300;<br>S7-400 standard;<br>S7-1200 standard;<br>S7-1500 standard;<br>WinAC RTX F | |
| Downloading safety program and standard user program | Description<br>• Downloading<br>• Program identification<br>• Comparing safety programs | under Downloading the Safety Program (Page 268)<br>under Comparing Safety Programs (Page 286) | |
| Testing the safety program | • Description of disabling of safety mode<br>• Procedures for changing safety program data | under Function test of safety program and protection through program identification (Page 274); Testing the safety program (Page 295); Disabling safety mode (Page 293) | |
| Changing the safety program | Description<br>• Disabling safety mode<br>• Changing the safety program | under Changing the safety program in RUN mode (S7-300, S7-400) (Page 302); Disabling safety mode (Page 293); Deleting the safety program (Page 305); | |
| Testing the safety-related parameters | Description of configuration | under Printing project data (Page 290);<br>*F-SMs Manual*, sections 4, 9, 10;<br>*F-Modules Manual*, sections 2.4, 7;<br>*ET 200eco Manual*, sections 3, 8;<br>*ET 200pro Manual,* Sections 2.4, 8;<br>*ET 200iSP Manual,* Sections 2.4, 7, 8<br>*ET 200SP Modules Manual*, section 4<br>*ET 200MP Modules Manual*, section 4 | |
| **System acceptance** | | | |
| Acceptance | • Description and notes on acceptance<br>• Printouts | under System Acceptance (Page 308) | |

| Phase | Note the following | Reference | Check |
|---|---|---|---|
| **Operation, maintenance** | | | |
| General operation | Notes on operation | under Notes on Safety Mode of the Safety Program (Page 322) | |
| Access protection | — | under Access protection (Page 76) | |
| Diagnostics | Responses to faults and events | under Guide to diagnostics (S7-300, S7-400) (Page 326); Guide to diagnostics (S7-1200) (Page 327); Guide to diagnostics (S7-1500) (Page 327); | |
| Replacement of software and hardware components | Description<br>• Module replacement<br>• Update of operating systems of F-CPU – same as in standard<br>• Update of SW components<br>Notes<br>• Operating system update of IMs | under Replacing Software and Hardware Components (Page 324); F-I/O access (Page 122);<br>Help on *STEP 7* | |
| Uninstallation, disassembly | • Notes for uninstalling software components<br>• Notes for disassembling modules | under Installing/uninstalling the STEP 7 Safety Advanced V13 SP1 optional package (Page 28); Installing/uninstalling the STEP 7 Safety Basic V13 SP1 optional package (Page 27); Replacing Software and Hardware Components (Page 324); | |

# Glossary

### Access protection

→ Fail-safe systems must be protected against dangerous, unauthorized access. Access protection for F-systems is implemented by assigning two passwords (one for the → F-CPU, and one for the → safety program).

### Automatically generated F-blocks

→ F-blocks that are automatically generated and, if necessary, called when the → safety program is compiled, in order to generate an executable safety program from the safety program programmed by the user.

### Category

Category in accordance with ISO 13849-1:2006 or EN ISO 13849-1:2008

With SIMATIC Safety, use in → safety mode up to category 4 is possible.

### Channel fault

Channel-related fault, such as a wire break or short-circuit.

### Collective F-signatures

The collective F-signatures uniquely identify a particular state of the → safety program and the safety-related parameters of the F-CPU and F-I/O. They are important for the on-site acceptance of the safety program, e.g., by → experts.

### CPU-wide

"CPU-wide" means all F-I/Os assigned to an F-CPU: central F-I/O of this F-CPU as well as F-I/Os for which the F-CPU is DP master/IO controller. An F-I/O that is addressed using I-slave-slave communication is assigned to the F-CPU of the I-slave and not to the F-CPU of the DP master / IO controller.

### CRC

Cyclic Redundancy Check → CRC signature

### CRC signature

The validity of the process data in the → safety message frame, the correctness of the assigned address relationships, and the safety-related parameters are validated by means of a CRC signature contained in the safety message frame.

### DB for F-runtime group communication

-> F-DB for safety-related communication between F-runtime groups of a safety program.

### Depassivation

→ Reintegration

### Disabled safety mode

Temporary deactivation of → safety mode for test purposes, commissioning, etc.

The following actions are possible only in disabled safety mode:

- Downloading changes of the → safety program to the -> F-CPU during ongoing operation (in RUN mode)
- Test functions such as "Modify" or other write access to data of the → safety program (with limitations)

Whenever safety mode is deactivated, the safety of the system must be ensured by other organizational measures, such as operation monitoring and manual safety shutdown.

### Discrepancy analysis

Discrepancy analysis for equivalence or non-equivalence is used for fail-safe inputs to detect errors caused by the time characteristic of two signals with the same functionality. The discrepancy analysis is initiated when different levels are detected in two associated input signals (when testing for non-equivalence: the same level). On expiration of an assignable period (→ discrepancy time), a check is made to determine whether the difference in levels (for non-equivalence testing, the same level) has disappeared after an assignable time period, the so-called discrepancy time. If not, there is a discrepancy error. The discrepancy analysis is performed between the two input signals of the 1oo2 sensor evaluation (→ sensor evaluation) in the fail-safe input.

### Discrepancy time

Assignable time for the → discrepancy analysis. If the discrepancy time is set too high, the fault detection time and → fault reaction time are prolonged unnecessarily. If the discrepancy time is set too low, availability is decreased unnecessarily because a discrepancy error is detected when, in reality, no error exists.

### DP/DP coupler

Device for coupling two PROFIBUS DP subnets required for master-master communication between → safety programs in different → F-CPUs in SIMATIC Safety and S7 Distributed Safety.

### Expert

The acceptance of a system, i.e., the safety-related acceptance test of the system, is usually carried out by an independent expert (for example, from TÜV).

### Fail-safe GSD based DP slaves

Fail-safe GSD based DP slaves are standard slaves that are operated on PROFIBUS with the DP protocol. They must operate in accordance with IEC 61784-1: 2010 (fieldbus profile) and the PROFIsafe bus profile. A GSD file is used for their configuration.

### Fail-safe GSD based I/O devices

Fail-safe GSD based I/O devices are standard devices that are operated on PROFINET with the I/O protocol. They must operate in accordance with IEC 61784-1: 2010 (fieldbus profile) and the PROFIsafe bus profile in V2 MODE. A GSD file is used for their configuration.

### Fail-safe I/O modules

ET 200eco modules that can be used for safety-related operation (-> safety mode). These modules are equipped with integrated → safety functions. They operate in accordance with IEC 61784-1: 2010 (fieldbus profile) and the PROFIsafe bus profile.

### Fail-safe modules

Fail-safe modules ET 200SP, ET 200S, ET 200pro, ET 200iSP that can be used in the ET 200SP, ET 200S, ET 200pro or ET 200iSP distributed I/O systems.

Fail-safe module ET 200MP, which can be used centrally in an S7-1500 or in a distributed I/O ET 200MP system.

Fail-safe module S7-1200 which can be used centrally in an S7-1200 system.

These modules are equipped with integrated safety functions (→ Safety functions) for fail-safe operation (→ Fail-safe operation). They operate in accordance with the → PROFIsafe bus profile.

### Fail-safe systems

Fail-safe systems (F-systems) are systems that remain in a safe state or immediately switch to another safe state as soon as particular failures occur.

### Fault reaction function

→ User safety function

### Fault reaction time

The maximum fault reaction time for an F-system specifies the time between the occurrence of any error and a safe reaction at all affected fail-safe outputs.

### F-blocks

The following fail-safe blocks are designated as F-blocks:

- those created by the user in LAD or FBD
- those created by the user as → F-DBs
- those selected by the user from a global library
- those added automatically in the → safety program (→ F-SBs, → automatically generated F-blocks, → F-shared DB, → F-I/O DBs; instance DBs of F-FBs)

All F-blocks are shown in yellow in the project tree.

### F-CALL

"F-call blocks" for the → safety program in *S7 Distributed Safety*.

### F-communication DBs

Fail-safe data blocks for the safety-related CPU-CPU communication via S7 connections.

### F-compliant PLC data type

An F-compliant PLC data type is a PLC data type in which you can use all data types that can be used in safety programs.

### F-CPU

An F-CPU is a central processing unit with fail-safe capability that is approved for use in SIMATIC Safety and in which a → safety program can run in addition to the → standard user program.

### F-DBs

Optional fail-safe data blocks that can be read-/write-accessed from anywhere within the safety program (exception: DBs for F-runtime group communication).

### F-FBs

Fail-safe function blocks (with instance DBs), in which the user programs the → safety program in FBD or LAD.

### F-FCs

Fail-safe FCs, in which the user programs the → safety program in → FBD or → LAD.

## F-I/O

Collective name for fail-safe inputs and outputs available in *SIMATIC S7* for integration in SIMATIC Safety, among others. The following are available:

- → ET 200eco fail-safe I/O module
- → S7-300 fail-safe signal modules
- → Fail-safe modules for S7-1200
- → Fail-safe modules for ET 200MP
- → Fail-safe modules for ET 200SP
- → Fail-safe modules for ET 200S
- → Fail-safe modules for ET 200pro
- → Fail-safe modules for ET 200iSP
- → Fail-safe GSD based DP slaves
- → Fail-safe GSD based I/O devices

## F-I/O DB

Fail-safe data block for F-CPUs to an → F-I/O in *STEP 7 Safety*. An F-I/O DB is automatically created for each F-I/O when the F-I/O is configured in the *hardware and network editor*. The F-I/O DB contains tags that the user can or must evaluate or write in the safety program as follows:

- For reintegration of the F-I/O after communication errors
- For reintegration of F-I/O after F-I/O or channel faults
- If F-I/O are to be passivated as a result of particular safety program statuses (for example, group passivation)
- For reassignment of parameters for fail-safe GSD based DP slaves/GSD based I/O devices or enabling HART communication for the F-I/O with the corresponding functionality
- In order to evaluate whether fail-safe values or process data are output

## F-I/O faults

Module-related F-I/O fault, such as a communication error or parameter assignment error

## F-I/Os of PROFIsafe address type 1

F-I/Os which ensure the uniqueness of the PROFIsafe address solely with the F-destination address, for example, ET 200S F-modules. The PROFIsafe address is usually assigned by DIP switches.

### F-I/Os of PROFIsafe address type 2

F-I/Os which can ensure the uniqueness of the PROFIsafe address with a combination of F-source address and F-destination address, for example, ET 200MP F-modules. The PROFIsafe address is usually assigned with *STEP 7 Safety*.

### F-modules

→ Fail-safe modules

### F-OB

The F-OB calls the main safety block of an F-runtime group in S7-1200/1500 F-CPUs.

### F-runtime group

The → safety program consists of one or two F-runtime groups. An F-runtime group is a logical construct of several associated → F-blocks. It is generated internally by the F-system. An F-runtime group consists of the following F-blocks:

→ Main safety block, F-OB (S7-1200, S7-1500), if applicable → F-FBs/ → F-FCs, if applicable → F-DBs, → F-I/O DBs, F-blocks of global libraries, instance DBs, → F-SBs, and → automatically generated F-blocks.

### F-runtime group information DB

The F-runtime group information DB provides key information on the corresponding → F-runtime group and on the → safety program as a whole.

### F-SBs

Fail-safe system blocks that are automatically inserted and called when the → safety program is compiled in order to generate an executable safety program from the user's safety program.

### F-shared DB

(S7-300, S7-400) Fail-safe data block that contains all of the shared data of the → safety program and additional information needed by the F-system. The F-shared DB is automatically inserted and expanded when the hardware configuration is compiled. Using its name F_GLOBDB, the user can evaluate certain data of the → safety program.

### F-SMs

→ S7-300 fail-safe signal modules

### F-systems

→ Fail-safe systems

### I-device

The functionality of the "I-device" (intelligent I/O-device) of a CPU allows data exchange with an I/O-controller and thus, its use as an intelligent preprocessor of sub-processes, for example. In this case, the I-device is connected as an I/O-device to a "parent" I/O-controller.

### IE/PB link

Device for coupling PROFINET IO and PROFIBUS DP systems required, among other things, for IO controller-I-slave communication between -> safety programs in different → F-CPUs in SIMATIC Safety.

### i-parameter

Individual parameters of → fail-safe GSD based DP slaves and → fail-safe GSD based I/O devices

### I-slave

The functionality of the "I-slave" (intelligent DP slave) of a CPU allows data exchange with a DP master and, thus, its use as an intelligent preprocessor of sub-processes, for example. In this case, the I-slave is connected as a DP slave to a "parent" DP master.

### Main safety block

"Introductory F-block" for fail-safe programming of the → safety program in *STEP 7 Safety*. The main safety block is an → F-FB or → F-FC that the user assigns to the calling F-OB (S7-1200, S7-1500) or block (OB, FC, FB) (S7-300, S7-400) of an → F-runtime group.

The main safety block contains the safety program and any calls of other → F-FBs/F-FCs for program structuring.

### Network-wide

A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet. In PROFIBUS, a network includes all nodes accessible via PROFIBUS DP. In PROFINET IO, a network includes all nodes accessible via RT_Class_1/2/32 (Ethernet/WLAN/Bluetooth, Layer 2) and if applicable RT_Class_UDP (IP, Layer 3).

### Passivation

When passivation occurs in an → F-I/O with inputs, the → F-system provides the safety program with fail-safe values (0) instead of the process data pending at the fail-safe inputs in the PII.

When passivation occurs in an F-I/O with outputs, the F-system transfers fail-safe values (0) to the fail-safe outputs instead of the output values in the PIQ provided by the safety program.

## PL

Performance Level (PL) according to ISO 13849-1: 2006 or in accordance with EN ISO 13849-1: 2008

With SIMATIC Safety, use up to Performance Level (PL) e is possible in → safety mode.

## PN/PN coupler

Device for coupling two PROFINET IO systems required for IO controller-IO controller communication between → safety programs in different → F-CPUs in SIMATIC Safety and S7 Distributed Safety.

## PROFIsafe

Safety-related bus profile of PROFIBUS DP/PA and PROFINET IO for communication between the → safety program and the → F-I/O in an → F-system. See IEC 61784-3-3:2010 or PROFIsafe – Profile for Safety Technology on PROFIBUS DP and PROFINET IO; Order No: 3.192 (V2.6.1).

## PROFIsafe address

The PROFIsafe address (code name in accordance with IEC 61784-3-3: 2010) is used to uniquely identify the source and destination. The PROFIsafe address consists of an F-source address and an F-destination address. Each → F-I/O therefore has two address parts, an F-source address and an F-destination address.

The F-source address is automatically assigned and is displayed for fail-safe GSD based DP slaves/fail-safe GSD based I/O devices and ET 200SP fail-safe modules. The F-source address for fail-safe modules ET 200S, ET 200eco, ET 200pro, ET 200iSP and F-SMs S7-300 is always 1. The F-source address is the basis for the PROFIsafe addresses of the assigned F-CPU for ET 200SP/ET 200MP fail-safe modules.

You need to configure the F-destination address in the *hardware and network editor*. You assign the F-destination address for the ET 200S, ET 200eco, ET 200pro, ET 200iSP and F-SMs S7-300 F-modules with a switch. For ET 200SP F-modules and ET 200MP F-modules, assign the F-destination address in the *hardware and network editor*. For S7-1200 F-modules, the F-destination address is automatically assigned by the F-system.

## Program signature

→ collective F-signature

## Proof test interval

Time period after which a component must be put into error-free state. That is, it is replaced by an unused component or it is proven to be completely error-free.

### Reintegration

The switchover from fail-safe values (0) to process data (reintegration of an → F-I/O) takes place automatically or following user acknowledgment in the F-I/O DB. The reintegration method depends on the following:

- The reason for → passivation of the F-I/O/channels of the F-I/O
- The parameter assignment in the → F-I/O DB or in the configuration itself (for example, ET 200MP fail-safe modules on an S7-1500 F-CPU and S7-1200 fail-safe modules on an S7-1200 F-CPU)

Following reintegration for an → F-I/O module with inputs, the process data pending at the inputs in the PII are provided again for the safety program. For an F-I/O with outputs, the F-system again transfers the output values provided in the PIQ in the safety program to the fail-safe outputs.

### RIOforFA Safety

Remote IO for Factory Automation with PROFIsafe; profile for F-I/O

### S7-300 fail-safe signal modules

Fail-safe signal modules of the S7-300 module series that can be used for safety-related operation (→ Safety mode) as centralized modules in an S7 300 or as distributed modules in the ET 200M distributed I/O system. The fail-safe signal modules are equipped with integrated → safety functions.

### S7-PLCSIM

The *S7-PLCSIM* application enables you to execute and test your program on a simulated automation system on your programming device or PC. Because the simulation takes place completely in your programming device or PC, you do not need any hardware (CPU, I/O).

### SAE

→ Safety Administration Editor

### Safe state

The basic principle of the safety concept in → fail-safe systems is the existence of a safe state for all process variables. For digital → F-I/O that conform to IEC 61508:2010, this is always the value "0".

### Safety Administration Editor

The Safety Administration Editor provides support for the main tasks of your safety program.

### Safety function

Mechanism integrated in the → F-CPU and → F-I/O that allows them to be used in -> fail-safe systems.

According to IEC 61508:2010, a function that is implemented by a safety device in order to maintain the system in the safe state or bring the system to a safe state in the event of a specific fault. (fault reaction function -> user safety function)

### Safety message frame

In → safety mode, data are transferred in a safety frame between the → F-CPU and → F-I/O, or between the F-CPUs in safety-related CPU-CPU communication.

### Safety mode

1. Operating mode of → F-I/O in which → safety-related communication can take place using → safety message frames.
2. Operating mode of the safety program. In safety mode of the safety program, all safety mechanisms for error detection and error reaction are enabled. In safety mode, the safety program cannot be modified during operation. Safety mode can be disabled by the user (→ disabled safety mode).

### Safety program

Safety-related user program

### Safety protocol

→ Safety message frame

### Safety-related communication

Safety-related communication is used to exchange fail-safe data.

### Sensor evaluation

There are two types of sensor evaluation:

- 1oo1 evaluation – sensor signal is read once
- 1oo2 evaluation - sensor signal is read twice by the same → F-I/O and compared internally

### Signature

→ collective F-signatures

### SIL

Safety Integrity Level (SIL)according to IEC 61508:2010. The higher the Safety Integrity Level is, the more stringent the measures for avoiding and controlling system faults and random hardware failures.

With SIMATIC Safety, up to Safety Integrity Level SIL3 is possible in safety mode.

### Standard communication

Communication used to exchange non-safety-related data

### Standard mode

Operating mode of → F-I/O in which → safety-related communication between the F-CPU and the F-I/O by means of → safety message frames is not possible; only → standard communication is possible in this operating mode.

### Standard user program

Non-safety-related user program

### Startup of F-system

In the STOP/RUN transition of an → F-CPU, the → standard user program starts up in the normal way. When the → safety program is started up, all F-DBs are initialized with the values from the load memory - as is the case with a cold restart. This means that saved error information is lost.

The → F-system performs an automation → reintegration of the → F-I/O.

### User safety function

The → safety function for the process can be provided through a user safety function or a fault reaction function. The user only has to program the user safety function. In the event of an error, if the → F-system can no longer execute its actual user safety function, it executes the fault reaction function: for example, the associated outputs are disabled, and the → F-CPU switches to STOP mode, if necessary.

### Value status

The value status is additional binary information for a channel value. The value status is entered in the process image of the inputs and provides information on the validity of the channel value.

1: A valid process data is output for the channel value.

0: A fail-safe value is output for the channel value.

# Index

## D



## E

## F



## G



## H

## I



## J



## L



## M



## N



## O

## P



## Q



## R

## S

## T



## U



## V

## W



## X