SIEMENS

SIMATIC

S7 Distributed Safety - Configuring and Programming

Programming and Operating Manual

10/2007
A5E00109537-04

Safety Guidelines

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
with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

CAUTION
without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

NOTICE
indicates that an unintended result or situation can occur if the corresponding information is not taken into account.

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 device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel**. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

Prescribed Usage

Note the following:

⚠WARNING
This device may only be used for the applications described in the catalog or the technical description and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens. Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

Trademarks

All names identified by ® are registered trademarks of the 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
Automation and Drives
Postfach 48 48
90327 NÜRNBERG
GERMANY

Ordernumber: A5E00109537-04
℗ 10/2007

Preface

Preface

Purpose of this Documentation

The information in this documentation enables you to configure and program S7 Distributed Safety fail-safe systems.

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/S7-400 automation systems
- Distributed I/O systems on PROFIBUS DP/PROFINET IO
- STEP 7 standard package, particularly:
  - Working with *SIMATIC Manager*
  - LAD and FBD programming languages
  - Hardware configuration with *HW Config*
  - Communication between CPUs

Scope of Documentation

This documentation is applicable to the following optional package:

Software	Order number	Release Number and Higher
S7 Distributed Safety optional package	6ES7833-1FC02-0YA5	V5.4 SP4

The *S7 Distributed Safety* optional package is used for configuring and programming S7 Distributed Safety fail-safe systems. Integration of the fail-safe I/O listed below in S7 Distributed Safety is also addressed:

- ET 200S fail-safe modules
- ET 200eco fail-safe I/O modules
- ET 200pro fail-safe modules
- S7-300 fail-safe signal modules
- Fail-safe DP standard slaves
- Fail-safe standard I/O devices

What's New

This documentation reflects the following significant changes/additions to the previous version:

- The contents of the Product Information for *S7 Distributed Safety* V5.4 SP1 and SP3 Edition 01/2007 have been integrated into this manual.
- Description of the following important innovations in *S7 Distributed Safety* V5.4 SP4:
  - Ability to install the S7 Distributed Safety optional package in Windows Vista
  - Support of SM 336, F-AI 6 x 0/4 ... 20 mA HART fail-safe signal module
  - Support of the "Compatibility mode" F-CPU parameter

Approvals

S7 Distributed Safety, ET 200S, ET 200eco, and ET 200 pro fail-safe modules, and S7-300 fail-safe signal modules are certified for use in safety mode up to and including the following:

- SIL3 (Safety Integrity Level) in accordance with IEC 61508
- Category 4 in accordance with EN 954-1

Position in the Information Landscape

Depending on your application, you will need the following supplementary documentation when working with *S7 Distributed Safety*.

This documentation includes references to the supplementary documentation where appropriate.

Documentation	Brief Description of Relevant Contents
Safety Engineering in SIMATIC S7 system manual	• Provides general information about the use, structure, and function of S7 Distributed Safety and S7 F/FH fail-safe automation systems • Contains detailed technical information about the S7 Distributed Safety and S7 F/FH systems • Contains monitoring time and response time calculations for S7 Distributed Safety and S7 F/FH fail-safe systems
For S7 Distributed Safety system	The following documentation is required according to the utilized F-CPU: • S7-300, CPU 31xC and CPU 31x: *Installation* operating instructions describe how to assemble and wire S7-300 systems. • The *CPU 31xC and CPU 31x, Technical Specifications* 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. • The *Automation System S7-400 Hardware and Installation* installation manual describes how to assemble and wire S7-400 systems. • The *Automation System S7-400 CPU Specifications* reference manual describes the CPU 416-2 and the CPU 416-3 PN/DP. • The *ET 200S IM 151-7 CPU Interface Module* manual describes the IM 151-7 CPU. • Every applicable F-CPU has its own product information. The product information describes only the deviations from the corresponding standard CPUs.
ET 200eco Distributed I/O Station Fail-Safe I/O Module manual	Describes the ET 200eco fail-safe I/O module hardware (including installation, wiring, and technical specifications)
ET 200S Distributed I/O System Fail-Safe Modules operating instructions	Describes the hardware of the ET 200S fail-safe modules (including installation, wiring, and technical specifications)
Automation System S7-300 Fail-Safe Signal Modules manual	Describes the hardware of the S7-300 fail-safe signal modules (including installation, wiring, and technical specifications)
ET 200pro Distributed I/O System - Fail-Safe Modules operating instructions	Describes the hardware of the ET 200pro fail-safe modules (including installation, wiring, and technical specifications)

Documentation	Brief Description of Relevant Contents
STEP 7 manuals	• The *Configuring Hardware and Communication Connections with STEP 7 V5.x* manual describes how to operate the applicable *STEP 7* standard tools. • The *Ladder Diagram (LAD) for S7-300/400* reference manual describes the Ladder Diagram standard programming language in *STEP 7*. • The *Function Block Diagram (FBD) for S7-300/400* reference manual describes the Function Block Diagram standard programming language in *STEP 7*. • The *System Software for S7-300/400 System and Standard Functions* reference manual describes functions for accessing and performing diagnostics on the distributed I/O and CPU. • The *Programming with STEP 7 V 5.x* manual provides an overview of programming with STEP 7 (e.g., installation, startup, program creation, and user program components).
STEP 7 online help	• Describes the operation of *STEP 7* standard tools • Contains information about configuration and parameter assignment for modules and I-slaves with *HW Config* • Contains a description of the FBD and LAD programming languages

The complete *SIMATIC S7* documentation is available on CD-ROM.

Guide

This documentation describes how to work with the *S7 Distributed Safety* optional package. It includes both instructional material and reference material (description of fail-safe library blocks).

The following topics are addressed:

- Configuring of S7 Distributed Safety
- Access protection for S7 Distributed Safety
- Programming of safety program (safety-related user program)
- Safety-related communication
- F-libraries
- Support for system acceptance test
- Operation and maintenance of S7 Distributed Safety

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 *"S7 Distributed Safety"* appears in italics, it refers to the optional package for the "S7 Distributed Safety" fail-safe system.

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 user program is referred to as the "standard user program".

All fail-safe blocks are represented with a yellow background on the STEP 7 user interface (in SIMATIC Manager, for example) to distinguish them from standard user program blocks.

Additional Support

For any unanswered questions about the use of products presented in this manual, contact your local Siemens representative.

A list of Siemens representatives is available at:

http://www.siemens.com/automation/partner

Access to technical documentation for individual SIMATIC products and systems is available at:

http://www.automation.siemens.com/simatic/portal/html_76/techdoku.htm

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.

Phone: +49 (911) 895–3200

http://www.sitrain.com

H/F Competence Center

The H/F Competence Center in Nuremberg offers special workshops on *SIMATIC S7* fail-safe and fault-tolerant automation systems. The H/F Competence Center can also provide assistance with on-site configuration, commissioning, and troubleshooting.

Phone: +49 (911) 895-4759
Fax: +49 (911) 895-5193

For questions about workshops, etc., contact: hf-cc.aud@siemens.com

Technical Support

Technical support is available for all A&D products

- Using the Web form for a support request
  http://www.siemens.com/automation/support-request
- Phone: + 49 180 5050 222
- Fax: + 49 180 5050 223

You can find additional information about our technical support at
http://www.siemens.com/automation/service&support

Service & Support on the Internet

In addition to our paper documentation, we also provide all of our technical information on the Internet at:

http://www.siemens.com/automation/service&support

Here, you will find the following information:

- Our newsletter, containing the latest information on your products.
- A search engine in Service & Support for locating the documents you need.
- A forum for global information exchange by users and experts.
- A list of local Siemens representatives.
- Information regarding on-site service, repairs, spare parts, and much more is available under "Services".

Important Information for Preserving the Operational Safety of your System

Note

Systems with safety-related characteristics are subject to special operational safety requirements on the part of the operator. The supplier is also obliged to comply with certain actions when monitoring the product. For this reason, we publish a special newsletter containing information on product developments and features that are (or could be) relevant to operation of safety-related systems. By subscribing to the relevant newsletter, you will always have the latest information and be able to make changes to your system, when necessary. To subscribe online, go to:

http://my.ad.siemens.de/myAnD/guiThemes2select.asp?subjectID=2&lang=en

and register for the following newsletters:

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

Select the "Updates" check box for each newsletter.

Table of contents

Product Overview

1

1.1 Overview

S7 Distributed Safety Fail-Safe System

The S7 Distributed 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 instrumentation and controls and burners).

Achievable Safety Requirements

S7 Distributed Safety fail-safe systems can satisfy the following safety requirements:

- Safety class (Safety Integrity Level) SIL1 to SIL3 in accordance with IEC 61508
- Category 2 to Category 4 in accordance with EN 954-1

Principles of Safety Functions in S7 Distributed Safety

Functional safety is implemented principally through safety functions in the software. Safety functions are executed by the S7 Distributed Safety system to place or maintain the system in a safe state in case of a dangerous occurrence. 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 fail-safe I/O ensure safe processing of field information (emergency STOP buttons, light barriers, motor control). They contain all of the required hardware and software components for safe processing in accordance with the required safety class. 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 deactivated, and the F-CPU switches to STOP mode, if necessary.

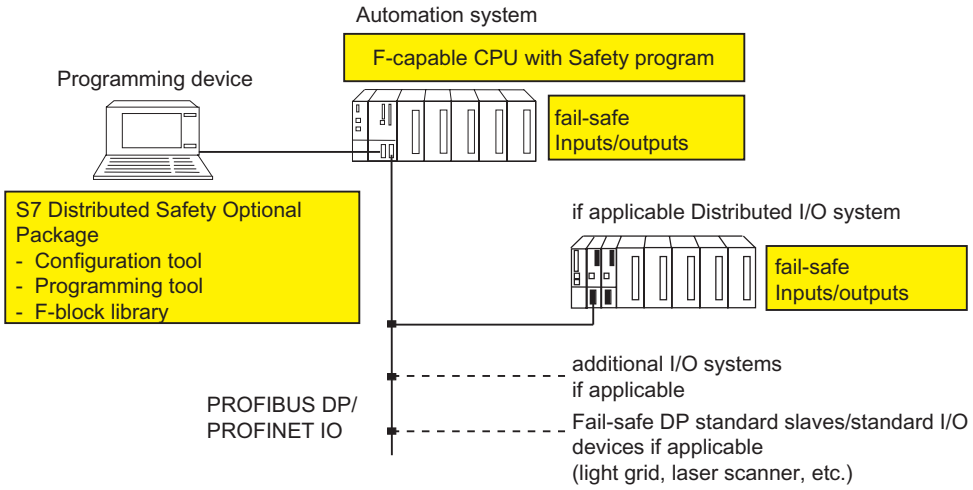
Example of User Safety Function and Fault Reaction Function

In the event of overpressure, the F-system opens a valve (user safety function). In the event of a hazardous fault in the F-CPU, all outputs are deactivated (fault reaction function), whereby the valve is opened, and the other actuators also attain a safe state. If the F-system is intact, only the valve is opened.

1.2 Hardware and Software Components

Hardware and Software Components of S7 Distributed Safety

The following figure provides an overview of the hardware and software components required to configure and operate an S7 Distributed Safety fail-safe system.



Hardware Components for PROFIBUS DP

The hardware components of S7 Distributed Safety include the following:

- F-CPU, such as 315F-2 DP CPU
- Fail-safe inputs and outputs (F-I/O), such as:
  - S7-300 fail-safe signal modules in S7 Distributed Safety (centralized configuration)
  - S7-300 fail-safe signal modules in ET 200M (distributed configuration)
  - Fail-safe power and electronic modules in ET 200S
  - ET 200eco fail-safe I/O module
  - Fail-safe modules in ET200pro
  - Fail-safe DP standard slaves

You can expand the configuration using standard I/O.

Hardware Components for PROFINET IO

You can use the following fail-safe components in S7 Distributed Safety F-systems on PROFINET IO:

- F-CPUs with PN interface, e.g., CPU 416F-3 PN/DP
- Fail-safe electronic modules in ET 200S
- Fail-safe electronic modules in ET 200pro
- Fail-safe standard I/O devices (light grid, laser scanner, etc.)

You can expand the configuration using standard I/O.

Additional Information

Detailed information on hardware components can be found in the *Safety Engineering in SIMATIC S7* system manual.

Software Components

Software components of S7 Distributed Safety include the following:

- *S7 Distributed Safety* optional package on the programming device/PC for configuring and programming the F-system
- Safety program in the F-CPU

In addition, you need the *STEP 7* basic software on the programming device or PC for configuring and programming the standard PLC.

S7 Distributed Safety Optional Package

This documentation describes the *S7 Distributed Safety* V5.4 SP4 optional package. *S7 Distributed Safety* is the configuration and programming software for the S7 Distributed Safety fail-safe system. With *S7 Distributed Safety,* you receive the following:

- Support for configuring the F-I/O in *STEP 7* using *HW Config*
- Support for creating the safety program and integrating error detection functions into the safety program
- F-library containing fail-safe application blocks that you can use in your safety program

Moreover, *S7 Distributed Safety* offers functions for comparing safety programs and for assisting you with the system acceptance test.

Safety Program

You create a safety program with the *FBD/LAD Editor* in *STEP 7*. You program fail-safe FBs and FCs in the F-FBD or F-LAD programming languages and create fail-safe DBs in the F-DB programming language. The supplied *Distributed Safety* F-library (V1) provides fail-safe application blocks that you can use in your safety program.

Safety checks are automatically performed and additional fail-safe blocks for error detection and fault 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 are exchanged between the safety program and the standard user program in the F-CPU by means of bit memory or by accessing the process input and output images.

1.3 Installing/Removing the S7 Distributed Safety V5.4 SP4 Optional Package

Software Requirements for S7 Distributed Safety V5.4 SP4

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

- STEP 7 V5.3 Service Pack 3 or higher

> **⚠ WARNING**
>
> Use of *S7 Distributed Safety Programming* V5.4 Service Pack 4 with earlier versions of STEP 7 is not permitted.

- *S7 F Configuration Pack* V5.2 Service Pack 3 or higher

Use of the following functions requires the software indicated below:

Function	Software Requirement
Safety-related I-slave-slave communication for S7-300 fail-safe signal modules (ET 200M)	*STEP 7* V5.4 and *S7 F Configuration Pack* V5.5 or higher
Disabling the deactivation of safety mode	*S7 F Configuration Pack* V5.5 SP1
F-iPar_CRC parameter for support of fail-safe DP standard slaves/standard I/O devices with individual device parameters (i-parameters)	*S7 F Configuration Pack* V5.5 SP1
Write-protected saving of F-blocks	*STEP 7* V5.4 SP2
"Rewiring" function of STEP 7 for F-blocks	*STEP 7* V5.4 SP2 and *S7 F Configuration Pack* V5.5 SP1
Fail-safe standard I/O devices	*STEP 7* V5.4 SP2 *S7 F Configuration Pack* V5.4
Support of SM 336, F-AI 6 x 0/4 ... 20 mA HART **without** use of HART function	*S7 F Configuration Pack* V5.5 SP4
Use of SM 336, F-AI 6 x 0/4 ... 20 mA HART **with** use of HART function	*STEP 7* V5.4 SP3 and *S7 F Configuration Pack* V5.5 SP4

Reading Readme Files

The readme files contain 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 at a later time by selecting the **Start > Simatic > Information > English** menu command.

Installing S7 Distributed Safety

1. Start the programming device or PC on which the *STEP 7* standard package has been installed, and make sure that all *STEP 7* applications are closed.
2. Insert the product CD for the optional package.
3. Initiate the *SETUP.EXE* program on the CD.
4. Follow the instructions of the Setup program, bearing in mind the information in the readme files.

Starting S7 Distributed Safety

S7 Distributed Safety is completely integrated in *STEP 7*. This means you do not specifically start *S7 Distributed Safety*, rather each *STEP 7* application (*SIMATIC Manager*, *HW Config*, and *FBD/LAD Editor*) assists you in configuring and programming *S7 Distributed Safety*.

Displaying Integrated Help

Context-sensitive help is available for the *S7 Distributed Safety* dialogs. You can access this help during each configuration and programming step by pressing the F1 key or clicking the "Help" button. For advanced help, select **Help > Contents > Access Help for Optional Packages > S7 Distributed Safety Work with F-systems** menu command.

Removing S7 Distributed Safety

The *S7 Distributed Safety* optional package has two components as follows:

- *"S7 F Configuration Pack* V5.5 SP4*"*
- *"S7 Distributed Safety Programming* V5.4 SP4*"*

You can remove these components individually. Use the normal procedure in Windows for removing software:

1. In Windows, double-click the "Add or Remove Programs" icon in "Control Panel" to open the dialog box for installing software.
2. Select the appropriate entry in the list of installed software. Click "Add/Remove..." to remove the software.
3. If the "Remove shared file" dialog appears, click "No" in case you are in doubt.

Changeover to S7 Distributed Safety V5.4 SP4

Reading a safety program with S7 Distributed Safety V5.4 SP4

If you would like to use *S7 Distributed Safety* V5.4 SP4 to read, but not change, a safety program created with an earlier version of *S7 Distributed Safety*, open the "Safety Program" dialog with V5.4 SP4. Do **not** compile the safety program and do **not** save and compile with replacement of F-library blocks of the *Distributed Safety* F-library (V1) in *HW Config*.

Note

When you open the "Safety Program" dialog for a consistent safety program created with *S7 Distributed Safety* V5.1, the status "The safety program is consistent." is output, although different signatures are displayed.

Reason: the length of the signatures has changed from 16 to 32 bits.

Changing a safety program with S7 Distributed Safety V5.4 SP4

You can use the new functions of *S7 Distributed Safety* V5.4 SP4 in a safety program that was created with an earlier version of *S7 Distributed Safety* (see also "What's New" in the preface).

Note

Note that channel-level passivation of F-I/O and connection of F-I/O to PROFINET IO extend the runtime of the F-runtime group(s) and increase the work memory requirement of the safety program (see also *Excel file s7cotia.xls for response time calculation*). In addition, you must make at least 330 bytes of local data available for the safety program (see Chapter "Configuring the F-CPU").

If you want to use *S7 Distributed Safety* V5.4 SP4 to change a safety program created with an earlier version of S7 Distributed Safety, proceed as follows:

1. Compile the safety program with *S7 Distributed Safety* V 5.4 SP4 prior to making changes.

   Result: All F-blocks of the Distributed Safety F-library (V1) that were used in the safety program and for which there is a new version in the *Distributed Safety* F-library (V1) in V5.4 SP4 are automatically replaced following confirmation.

   The collective signature of all F-blocks and the signature of individual F-blocks change for the following reasons:

   – The length of the collective signature has been changed from 16 to 32 bits (for conversion from V5.1 to V5.4 SP4 only)

   – F-blocks of the *Distributed Safety* F-library (V1) were replaced

   – Automatically compiled F-blocks have changed

   When changing from V5.4 SP3 to V5.4 SP4, the collective signature of all F-blocks remains the same although the F-_CTRL_1 F-system block is replaced by a newer version (non-safety-related change).

2. Change the safety program according to your requirements.

3. Recompile the safety program.
4. Perform a comparison of the old and new version of the safety program in the "Compare safety program" dialog (see Chapter "Comparing safety programs").
   - You can identify changes to the version of an F-block of the *Distributed Safety* F-library (V1) by the changes to F-block signatures. The modified signatures and initial value signatures of all F-application blocks and F-system blocks must correspond to those in Annex 1 of the Certification Report.
   - Furthermore, you can identify whether changes have been made in the safety program. If necessary, the safety program must undergo another acceptance test.

Changing from *S7 Distributed Safety* V5.4 SP4 to an Earlier Version

If you want to change to an *S7 Distributed Safety* version < V5.4 SP4, you must completely remove *S7 Distributed Safety* V5.4 SP4 beforehand.

Changing from *S7 Distributed Safety* V5.4 SP4 to V5.3

When you open the "Safety Program" dialog for a consistent safety program created with *S7 Distributed Safety* V5.4 SP4, the status "The safety program is consistent." is output.

You can use V5.3 to modify a safety program created with V5.4 SP4 if you use only those functions that were made available in V5.3.

If you want to use V5.3 to change a safety program created with *S7 Distributed Safety* V5.4 SP4, proceed as follows:

1. Delete all automatically generated and added F-blocks in the offline block container of the safety program.
2. Save and compile the hardware configuration in *HW Config*.
3. Change the safety program according to your requirements.
4. Recompile the safety program.

Changing from *S7 Distributed Safety* V5.4 SP4 to V5.2

When you open the "Safety Program" dialog for a consistent safety program created with *S7 Distributed Safety* V5.4 SP4, the status "The safety program is not consistent." is output, even though the safety program is consistent.

You can use V5.2 to modify a safety program created with V5.4 SP4 if you use only those functions that were made available in V5.2.

The procedure for changing from V5.4 SP4 to V5.3 applies.

Calculation of the Maximum Response Time of your F-System

Use the Microsoft Excel file available for *S7 Distributed Safety* V5.4 SP4 to calculate the maximum response time of your F-system. This file is available for download at:

http://support.automation.siemens.com/WW/view/en/11669702/133100

See also

Safety Program Acceptance Test (Page 297)

Configuration

2

2.1 Overview of Configuration

Introduction

You configure an S7 Distributed Safety fail-safe system in basically the same way as a standard S7-300, S7-400, or ET 200S automation system.

For this reason, this section presents only the essential differences you encounter when configuring an S7 Distributed Safety F-system compared to standard PLC configuration.

F-Components That Must Be Configured

The following hardware components are configured for an S7 Distributed Safety F-system:

1. F-CPU, such as CPU 315F-2 DP
2. F-I/O, such as:
   - ET 200S fail-safe modules
   - S7-300 fail-safe signal modules (for centralized configuration next to the F-CPU or decentralized configuration in ET 200M)
   - ET 200pro fail-safe modules
   - ET 200eco fail-safe I/O modules
   - Fail-safe DP standard slaves
   - Fail-safe standard I/O devices

Information on F-I/O that Can be Used

For detailed information on which F-I/O can be used, refer to the manuals in the following table:

Topic	Reference
Configuration rules, such as: • Centralized configuration, distributed configuration with F-I/O • Coexistence of F-I/O and standard I/O	• *Safety Engineering in SIMATIC S7* system manual • *Manual for specific F-I/O*
PROFIsafe address assignment for F-I/O	*Manual and context-sensitive online Help for specific F-I/O*
Allocation of address areas by F-I/O in the F-CPU	*Manual for specific F-I/O*
Fail-safe DP standard slaves	*Documentation for specific fail-safe DP standard slave*
Fail-safe standard I/O devices	*Documentation for specific fail-safe standard I/O devices*

Safety-Related Communication Options that Can Be Configured

You must use *HW Config* to configure the following safety-related communication options:

- Safety-related master-master communication
- 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 communication via S7 connections

2.2 Particularities for Configuring the F-System

F-Systems Configured Same as Standard Systems

You configure an S7 Distributed Safety fail-safe system the same as a standard S7 system. That is, you configure and assign parameters for the hardware in *HW Config* as a centralized configuration (F-CPU and, if necessary, S7-300 F-SMs) and/or as a decentralized (distributed) configuration (F-CPU, F-SMs in ET 200M, F-modules in ET 200S, ET 200 pro, and ET 200eco, fail-safe DP standard slaves, fail-safe standard I/O devices).

For a detailed description of the configuration options, refer to the " *Safety Engineering in SIMATIC S7* " system manual.

Special F-Relevant Tabs

There are a few special tabs for the F-functionality included in the object properties of the fail-safe components (F-CPU and F-I/O). These tabs are described in the following sections.

Assigning Symbols for Fail-Safe Inputs/Outputs of F-I/O

For convenience when programming S7 Distributed Safety, it is particularly important that you assign symbols for the fail-safe inputs and outputs of the F-I/O in *HW Config*.

Saving and Compiling the Hardware Configuration

You must save and compile the hardware configuration of the S7 Distributed Safety F-system in *HW Config*. This is required for subsequent programming of the safety program.

Changing Safety-Relevant Parameters

Note

If you change a safety-relevant parameter for an F-I/O, a fail-safe DP standard slave, a fail-safe standard I/O device, or an F-CPU, you must recompile the safety program.

The same applies to changes in the configuration of safety-related communication and, in particular, for changes in the S7 connections for safety-related communication via S7 connections.

2.3 Configuring the F-CPU

Introduction

You configure the F-CPU in basically the same way as a standard automation system. For an *S7 Distributed Safety* F-system, you must also do the following:

- Configure Level of Protection 1.
- Configure the F parameters.

Configuring the Level of Protection of the F-CPU

⚠WARNING

In safety mode, access by means of the F-CPU password must not be authorized when making changes to the standard user program, since changes to the safety program can also be made. To rule out this possibility, you must configure **Level of Protection 1**. If only **one person** is authorized to change the standard user program **and** the safety program, level of protection "2" or "3" 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).

Use the following procedure to configure Level of Protection 1:

1. In *HW Config*, select the F-CPU, such as CPU 315F-2 DP, and select the **Edit > Object Properties** menu command.
2. Open the "Protection" tab.
3. Set Level of Protection "1: Access protection for F-CPU" and "Removable with Password".
   Enter a password for the F-CPU in the field provided, and select the "CPU contains safety program" option. Note that the "Mode" field is not relevant for safety mode.

   For information on the password for the F-CPU, refer to Chapter "Overview of Access Protection. Pay particular attention to the warnings in Chapter "Setting Up Access Permission for the F-CPU".

Configuring the F-Parameters of the F-CPU

Use the following procedure to configure the F-parameters:

1. In *HW Config,* select the F-CPU and select the **Edit > Object Properties** menu command.
2. Open the "F Parameters" tab. After opening the tab, you will be prompted to enter the password for the safety program, or you have to assign the password for the safety program in a separate dialog box. For information on the password for the safety program, refer to Chapter "Overview of Access Protection".

   In the "F parameters" tab, you can change or accept the default settings for the following parameters:

   – Enabling or disabling the function for deactivating safety mode
   – Base for PROFIsafe addresses
   – Compatibility mode for F-CPUs

     (only for F-CPUs that support PROFIsafe V2 MODE and that have only PROFIBUS DP interfaces (not PROFINET IO)
   – Band of numbers for F-data blocks
   – Band of numbers for F-function blocks
   – Local data volume provided for the safety program

   Note

   A change in the F-parameters of the F-CPU can cause changes in the safety program when it is recompiled, and consequently, a new acceptance test may be required.

"Safety Mode Can Be Deactivated" Parameter

You can enable or disable the function for deactivating safety mode in the "F-Parameters" tab. "Safety mode can be deactivated" is enabled in the default settings.

If you disable the function for deactivating safety mode, safety mode can generally no longer be deactivated. That is, you cannot deactivate safety mode even if you enter the password for the safety program:

- In the "Safety Program" dialog
- In the dialog box for deactivating safety mode during testing/commissioning functions and while loading F-blocks

"Basis for PROFIsafe Addresses" Parameter

This information is required for internal administration of the PROFIsafe addresses of the F-system.

The PROFIsafe addresses are used to uniquely identify the source and destination.

You can set the "Base for PROFIsafe addresses", i.e., the range for automatically assigning the PROFIsafe destination addresses, for:

- Newly placed ET 200S, ET 200pro, and ET 200eco F-I/O in *HW Config*
- S7-300 fail-safe signal modules:
  - That are newly placed and operable only in safety mode (see *S7-300 Fail-Safe Signal Modules* manual)
  - For which you have set safety mode for the first time in *HW Config* and whose PROFIsafe addresses are **not** assigned using the module starting addresses (see *S7-300 Fail-Safe Signal Modules* manual)

For all other F-I/O, this parameter has no affect.

Setting this parameter defines a range for the PROFIsafe target addresses. This is useful if several DP master systems and PROFINET IO systems are operated on one network. Subsequent address changes are possible, but not necessary, because the address range was reserved according to your parameter assignment.

You can specify the "Base for PROFIsafe addresses" in increments of 1000. PROFIsafe target addresses are always assigned automatically based on the following formula: Base for PROFIsafe address divided by 10. The maximum PROFIsafe target address possible is 1022.

Example: You set the base as "2000". PROFIsafe target addresses are automatically assigned starting with address 200.

"Compatibility Mode" Parameter

This parameter is available only for F-CPUs that support PROFIsafe V2 MODE and that have only PROFIBUS DP interfaces (not PROFINET IO).

A change in the default setting (= compatibility mode off) is only relevant if you want to replace an F-CPU in your hardware configuration that supports only PROFIsafe V1 MODE with an F-CPU that also supports PROFIsafe V2 MODE.

To prevent this CPU replacement and subsequent compilation from changing the safety program, thus requiring a new acceptance test, you must enable compatibility mode.

If you do not, the PROFIsafe MODE of all F-I/O that support V2 MODE will be changed to V2 MODE when the hardware configuration is saved and compiled in *HW Config*.

If your project uses F-I/O on PROFINET IO or in a hybrid configuration on PROFIBUS DP and PROFINET IO based on IE/PB Links, compatibility mode must be disabled.

"F-Data Blocks" Parameter

F-blocks are automatically added when the safety program is compiled to create an executable safety program from your safety program. You must reserve a band of numbers for the automatically added F-data blocks. You define the first and last number of the band.

Rule for selecting the magnitude of the band of numbers:

At a minimum, the default setting should be accepted. In addition, the following is applicable:

Number of automatically added F-data blocks =

Number of configured F-I/O

+ Number of F-DBs (except "DBs for F-runtime group communication")

+ 5 x number of "DBs for F-runtime group communication"

+ Number of F-block calls of type FB (F-FBs/F-PBs/F-application blocks)

+ Number of F-blocks of type FC (F-FCs/F-PBs/F-application blocks)

+ Number of F-blocks of type FC (F-FCs/F-PBs/ F-application blocks) used in two F-runtime groups

+6 x number of F-runtime groups

If the configured band of numbers turns out to be insufficient, *S7 Distributed Safety* signals this with an error message. You must then increase the size of the number band accordingly.

Tip: Allocate the band of numbers for the automatically added F-data blocks starting from the largest possible number in the F-CPU and working down. Assign numbers for DBs of the standard user program and for F-DBs and instance DBs of F-FBs or F-application blocks of the safety program starting with "1".

You are not permitted to use the reserved automatically added F-data blocks in the safety program or the standard user program.

If you have changed the band of numbers, e.g., you replaced an F-CPU with an F-CPU having a narrower band of numbers, some of the automatically added F-DBs in the modified band of numbers (the band of numbers associated with the new F-CPU) will not be created during the next compile operation. Instead, these F-DBs retain their old number. As a result, it may not be possible to download them to the F-CPU.

Solution: Delete all automatically generated F-blocks in the offline block container of the safety program, and recompile the safety program.

"F-Function Blocks" Parameter

F-blocks are automatically added when the safety program is compiled to create an executable safety program from your safety program. You must reserve a band of numbers for the automatically added F-function blocks. You define the first and last number of the band.

Rule for selecting magnitude of the band of numbers:
At a minimum, the default setting should be accepted. In addition, the following is applicable:

Number of automatically added F-function blocks =

Number of F-blocks (F-FBs/F-FCs/F-PBs/F-application blocks)

+ Number of F-blocks (F-FBs/F-FCs) that are called in two F-runtime groups

+ Number of F-application blocks contained in the reserved band of numbers

+ 5

If the configured band of numbers turns out to be insufficient, *S7 Distributed Safety* signals this with an error message. You must then increase the size of the number band accordingly.

Tip: Allocate the band of numbers for the automatically added F-data blocks starting from the largest possible number in the F-CPU and working down. Assign numbers for FBs of the standard user program and F-FBs of the safety program starting with "1".

You are not permitted to use the reserved automatically added F-function blocks in the safety program or the standard user program.

F-application blocks from the *Distributed Safety* F-library may be within this band of numbers.

"F-Local Data" Parameter

F-blocks are automatically added when the safety program is compiled to create an executable safety program from your safety program. This parameter is used to specify the amount of local data in bytes for the entire safety program, i.e. local data that are available for the F-CALL blocks of the F-runtime groups of the safety program and, thus, also for the automatically added F-blocks called in the F-CALL.

Note

The local data setting is applicable to all F-runtime groups of a safety program.

You must provide **at least 330 bytes** of local data for the safety program. However, the local data requirement for the automatically added F-blocks may be higher depending on the requirements of your safety program.

Thus, you should provide as much local data as possible for the automatically added F-blocks. If there is not enough local data available for the automatically added F-blocks (330 bytes or more), the safety program will be compiled nevertheless. Data in automatically added F-DBs are then used instead of local data. This increases the runtime of the F-runtime group(s), however. You will receive a notice via *S7 Distributed Safety* if the automatically added F-blocks would require more local data than configured.

⚠WARNING

The calculated maximum runtime of the F-runtime group using the MS Excel file s7fcotia.xls is no longer correct in this case because the calculation assumes sufficient F-local data are available.

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.

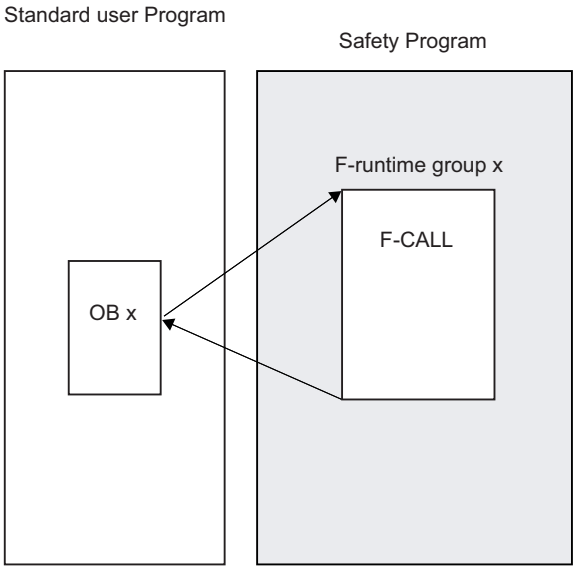
Note

Note that the maximum possible amount of F local data depends on the following:

- Local data requirement of your higher-level standard user program. For this reason, you should call the F-CALL blocks directly in OBs (cyclic interrupt OBs whenever possible), and additional local data should not be declared in these cyclic interrupt OBs.
- Maximum amount of local data of the utilized F-CPU (see *technical specifications in the Product Information for the utilized F-CPU*). For CPU 416F-2, you can configure the local data for each priority class. Therefore, allocate the largest possible local data area for the priority classes in which the safety program (F-CALL blocks) will be called (e.g., OB35).

Maximum Possible Amount of F Local Data According to Local Data Requirement of Higher-Level Standard User Program

Case 1: F-CALL blocks called directly in OBs

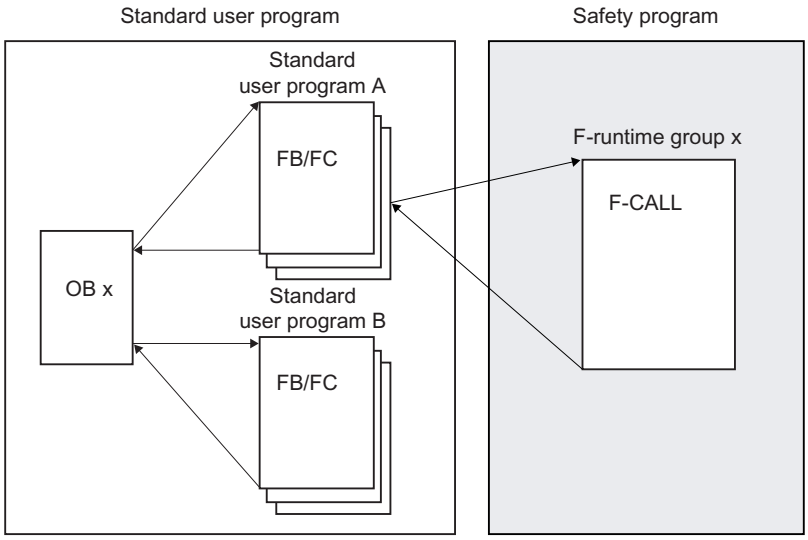


Set the "F local data" parameter to one the following:

- Maximum amount of local data of the F-CPU you are using minus 32 bytes
- Maximum amount of local data of the F-CPU you are using minus the local data requirement of OB x (for two F-runtime groups of OB x with the greatest local data requirement), if this amount is greater than 32 bytes.

Note: You can derive the local data requirement of the OBs from the program structure. For this purpose, select the **Options > Reference Data > Display** menu command in *SIMATIC Manager* (setting: "Program Structure" selected). This shows you the local data requirement in the path or for the individual blocks (see also *STEP 7 online Help*).

Case 2: F-CALL blocks not called directly in OBs



Set the "F local data" parameter to one of the following:

- Maximum amount of local data of the F-CPU you are using minus 32 bytes
- Maximum amount of local data of the F-CPU you are using minus the local data requirement of OB x (for two F-runtime groups of OB x with the greatest local data requirement) and minus the local data requirement of standard user program A, if these amounts combined are greater than 32 bytes.

Note: You can derive the local data requirement of the OBs and standard user program A from the program structure. For this purpose, select the **Options > Reference Data > Display** menu command in *SIMATIC Manager* (setting: "Program Structure" selected). This shows you the local data requirement in the path or for the individual blocks (see also *STEP 7 online Help*).

Local Data Requirement for the Automatically Added F-Blocks According to the Local Data Requirement of Your Safety Program

The information below must be taken into account only if the amount of local data available for your safety program is insufficient and you received a message from S7 Distributed Safety to that effect.

You can estimate the probable local data requirement for the automatically added F-blocks as follows:

For each F-runtime group, determine the local data requirement for each call hierarchy (path in the F-runtime group starting from and including the F-PB through all nesting levels down to the lowest) of your safety program:

Local data requirement for a call hierarchy (path local data requirement in bytes) =

2 x amount of all local data of F-FBs/F-FCs of data type BOOL in the path

+ 4 x amount of all local data of F-FBs/F-FCs of data type INT or WORD in the path

+ 6 x amount of all local data of F-FBs/F-FCs of data type TIME in the path

+ 42 x number of nesting levels in which an F-application block is called

+ 18 x number of nesting levels

+ 14 x number of nesting levels in which a fixed-point function or word logic instruction is programmed.

The estimated local data requirement for the automatically added F-blocks is then equivalent to the maximum path local data requirement for all paths of all F-runtime groups.

Note

If you are unable to provide a sufficient amount of local data for the automatically added F-blocks, we recommend that you reduce the local data requirement of your safety program, by reducing nesting depth, for example.

Use of Local Data in an F-FB or F-FC

Note

F-blocks are automatically added when the safety program is compiled to create an executable safety program from your safety program. If you use the local data memory area in an F-FB/F-FC, remember the following limit (irrelevant for S7-400 F-CPUs):

Local data requirement < maximum local data amount per block
(see *technical specifications in the Product Information for the F-CPU you are using*)

Mean local data requirement in bytes =

2 x amount of all local data of the F-FB/F-FC of data type BOOL

+ 4 x amount of all local data of the F-FB/F-FC of data type INT or WORD

+ 6 x amount of all local data of the F-FB/F-FC of data type TIME

+ 12

+ 14 (if a fixed-point function or word logic instruction is programmed)

+ 6 (if an F-FB, F-FC, or F-application block is called)

If the amount of local data required is greater, you cannot download your safety program to the F-CPU. Reduce the local data requirement of your programmed F-FB or F-FC.

See also

Installing/Removing the S7 Distributed Safety V5.4 SP4 Optional Package (Page 17)

Overview of Access Protection (Page 47)

Access Permission for the F-CPU (Page 53)

Structure of the Safety Program in S7 Distributed Safety (Page 57)

Overview of System Acceptance Test (Page 293)

2.4 Configuring the F-I/O

F-I/O Configured Same as Standard I/O

The ET 200S, ET 200eco, and ET 200pro F-modules and the S7-300 F-SMs are always configured in the same way:

Once the F-I/O have been inserted in the station window of *HW Config*, you can access the configuration dialog by selecting **Edit > Object Properties** or by double-clicking the F-I/O. After opening the dialog box, you will be prompted to enter the password for the safety program, or you have to assign the password for the safety program in a separate dialog box. For information on the password for the safety program, refer to *Overview of Access Protection*.

The values in the shaded fields are automatically assigned by *S7 Distributed Safety* in the F-relevant tab. You can change the values in the non-shaded fields.

Channel-Level 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, ET 200pro F-modules). You configure this behavior in the object properties for the relevant F-I/O ("Behavior after channel faults" 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

Note that channel-level passivation increases the runtime of the F-runtime group(s) compared to passivation of the entire F-I/O (see also *Excel file s7cotia.xls for response time calculation*).

Additional Information

For information on which ET 200S, ET 200eco, and ET 200pro **F-modules** and which S7-300 **F-SMs** you can use (centrally or decentrally), refer to the *Safety Engineering in SIMATIC S7* system manual.

For a description of the **parameters**, refer to the *context-sensitive online Help for the tab* and the relevant *F-I/O manual*.

For information on what you must consider when configuring the **monitoring time** for F-I/O, refer to the *Safety Engineering in SIMATIC S7* system manual.

PROFIsafe Addresses

The PROFIsafe addresses ("F_source_address", "F_destination_address" parameters) uniquely identify the source and destination.

F_destination_address

The F_destination_address uniquely identifies the PROFIsafe destination (of the F-I/O). Therefore, the F_destination_address must be unique network-wide and station-wide (see the following rules for address assignment).

To prevent incorrect parameter assignment, a **station-wide unique** F_destination_address is automatically assigned when the F-I/O is placed in *HW Config*.

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 object properties for the F-CPU) in S7 Distributed Safety F-systems differently **before** placing the F-I/Oin the various stations of a network.

If you change the F_destination_address, the uniqueness of the F_destination_address within the station is checked automatically. You yourself must make sure that the F_destination_address is unique network-wide.

You must set the F_destination_address on the F-I/O via the DIP switch before installing the F-I/O.

Note

For the following S7-300 F-SMs, the F_destination_address is the same as the start address of the F-SM/8:

- SM 326; DI 24 x DC 24 V (order no. 6ES7326-1BK00-0AB0)
- SM 326; DI 8 x Namur (order no. 6ES7326-1RF00-0AB0)
- SM 326 DO 10 x DC 24 V/2A (order no. 6ES7326-2BF01-0AB0)
- SM 336; AI 6 x 13 Bit (order no. 6ES7336-1HE00-0AB0)

The "Basis for PROFIsafe addresses" does not affect the assignment of the F_destination_address for these F-SMs.

Assign low start addresses for these F-SMs if you are also using other F-I/O.

F_source_address

The F_source_address is automatically assigned in *S7 Distributed Safety*.

Rules for Address Assignment

⚠WARNING

Rule for PROFIBUS subnets:

The PROFIsafe destination address and, thus, the switch setting on the address switch of the F-I/O must be unique network-wide* and station-wide** (system-wide). For S7-300 F-SMs and ET 200S, ET 200eco and ET 200pro F-modules, you can assign a maximum of 1022 different PROFIsafe destination addresses.

Exception: The F-I/O in different I-slaves may be assigned the same PROFIsafe destination address, as they are only addressed within the station, that is, by the F-CPU in the I-slave.

Rules for Ethernet subnets and hybrid configurations of PROFIBUS and Ethernet subnets:

The PROFIsafe destination address and, thus, the address switch setting on the F-I/O have to be unique only*** within the Ethernet subnet, including all lower-level PROFIBUS subnets, and station-wide** (system-wide). For S7-300 F-SMs and ET 200S, ET 200eco and ET 200pro F-modules, you can assign a maximum of 1022 different PROFIsafe destination addresses.

Exception: The F-I/O in different I-slaves may be assigned the same PROFIsafe destination address, as they are only addressed within the station, that is, by the F-CPU in the I-slave.

The networked nodes of an Ethernet subnet are characterized by having IP addresses with the subnet address, i.e. the IP addresses match in the digits that have the value "1" in the subnet mask.

Example:

IP address: 140.80.0.2.

Subnet mask: 255.255.0.0 = 11111111.11111111.00000000.00000000

Meaning: Bytes 1 and 2 of the IP address define the subnet; subnet address = 140.80.

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

** The address is unique for a station configured in *HW Config* (for example, an S7-300 station or I-slave)

*** Across Ethernet subnets, excluding cyclic PROFINET IO communication (RT communication)

Group Diagnostics for S7-300 F-SMs

The "Group diagnostics" parameter activates and deactivates the transmission of channel-specific diagnostic messages of F-SMs (such as wire break and short circuit) to the F-CPU. For availability reasons, you should shut down the group diagnostics on **unused** input or output channels of the following F-SMs:

- SM 326; DI 8 x NAMUR
- SM 326; DO 10 x 24 VDC/2 A
- SM 336; AI 6 x 13 Bit

> **⚠ WARNING**
>
> For fail-safe F-SMs in safety mode, "group diagnostics" must be activated on all **connected** channels.
>
> It is recommended that you check to verify that you shut down group diagnostics only for unused input and output channels.

Diagnostic interrupts can be enabled optionally.

For SM 326; DI 24 x 24 VDC (Order-No. 6ES7326-1BK01-0AB0 or higher) and SM 326; DO 8 x 24 VDC/2 A PM, the following applies:

If you deactivate a channel in *STEP 7 HW Config*, the group diagnostics for this channel is deactivated simultaneously.

2.5 Configuring Fail-Safe DP Standard Slaves and Fail-Safe Standard I/O Devices

Requirements

In order to use fail-safe DP standard slaves with S7 Distributed Safety, the standard slaves must be on the PROFIBUS DP and support the PROFIsafe bus profile. Fail-safe DP standard slaves used in hybrid configurations on PROFIBUS DP and PROFINET IO based on IE/PB links must support the PROFIsafe bus profile in V2 mode.

In order to use fail-safe standard I/O devices with S7 Distributed Safety, the standard devices must be on the 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 DP standard slaves is the device specification in the GSD file.

A GSD file contains all of the properties of a DP standard slave or standard I/O device. For fail-safe DP standard slaves/standard I/O devices, portions of the specification are ensured by a CRC.

The GSD files are supplied by the device manufacturers.

Protection of the Data Structure of the Device in GSD Files

Starting with *PROFIsafe Specification* V2.0, the device data structure described in the GSD file must be protected with a CRC stored in this file ("setpoint" for F_IO_StructureDescCRC).

F_IO_StructureDescCRC

You receive one of the following items of information for each configured fail-safe DP standard slave or standard I/O device when it is placed in *HW Config* or in the printout of the hardware configuration project data:

- The value calculated by *S7 Distributed Safety* for F_IO_StructureDescCRC matches/does not match the "setpoint" in the installed GSD file
- The "setpoint" for F_IO_StructureDescCRC is not available in the installed GSD file

Note

The information of the F_IO_StructureDescCRC is irrelevant for the system acceptance test (see Chapter "System Acceptance Test") if the project was compiled with *S7 Distributed Safety* V5.4 SP4.

For versions of S7 Distributed Safety > V5.4 SP4, the F_IO_StructureDescCRC check must be without errors (calculated value matches the setpoint). For this reason, you should obtain the appropriate GSD file containing the setpoint for F_IO_StructureDescCRC from the device manufacture.

Procedure for Configuring with GSD Files

You import the GSD files in your project (see *STEP 7 Online Help*).

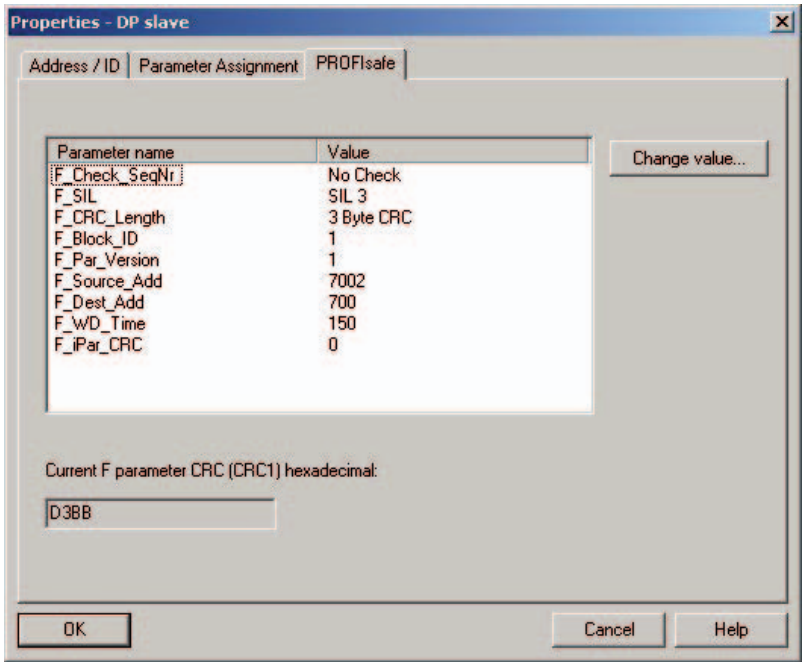
1. Select the fail-safe DP standard slave/standard I/O device in the hardware catalog of *HW Config* and insert it in your DP master system or IO system.
2. Select the fail-safe DP standard slave/standard I/O device.
3. Open the object properties dialog using the **Edit > Object Properties** menu command or by double-clicking the slot of the F-component. After opening the dialog box, you will be prompted to enter the password for the safety program, or you have to assign the password for the safety program in a separate dialog box. For information on the password for the safety program, refer to Chapter "Overview of Access Protection".

Channel-level passivation is not supported for fail-safe DP standard slaves/standard I/O devices.

"PROFIsafe" tab

The parameter texts specified in the GSD file are contained in the "PROFIsafe" tab under "Parameter name", and the current value for each parameter is found under "Value". You can modify this value using the "Change Value..." button.

The parameters are explained below.



"F_Check_SeqNr" Parameter

This parameter defines whether the sequence number is to be incorporated in the consistency check (CRC calculation) of the F-user data frame.

In PROFIsafe V1-MODE, you need to set the "F_Check_SeqNr" parameter to "No check". Only fail-safe DP standard slaves that behave accordingly are supported.
"F_CHECK_SeqNr" is irrelevant in PROFIsafe V2 mode.

"F_SIL" Parameter

This parameter defines the safety class of the fail-safe DP standard slave or standard I/O device. The parameter is device-dependent. Possible settings for the "F_SIL" parameter are "SIL 1" to "SIL 3", depending on the GSD file.

"F_CRC_Length" Parameter

Depending on the length of the F-user data (process data), the safety class, and the PROFIsafe MODE , the length of the CRC signature must be 2, 3 or 4 bytes. This parameter provides information to the F-CPU on the size of the CRC2 key in the safety message frame.

In PROFIsafe V1 mode:

For a user data length less than or equal to 12 bytes, select "2-byte CRC" as the setting for the "F_CRC_Length" parameter; for a user data length ranging from 13 bytes to 122 bytes, select "4-byte CRC".

S7 Distributed Safety supports only "2-byte CRC"; the fail-safe DP standard slave must behave accordingly.

In PROFIsafe V2 mode:

For a user data length less than or equal to 12 bytes, select "3-byte CRC" as the setting for the "F_CRC_Length" parameter; for a user data length ranging from 13 bytes to 123 bytes, select "4-byte CRC".

S7 Distributed Safety supports only "3-byte CRC"; the fail-safe DP standard slave/standard I/O device must behave accordingly.

"F_Block_ID" Parameter

The F_Block_ID parameter has the value 1 if the F_iPar_CRC parameter exists, otherwise it has the value 0.

The F_Block_ID parameter indicates that the data record for the value of F_iPar_CRC has been extended by 4 bytes. You must not change the parameter.

"F_Par_Version" Parameter

This parameter identifies the PROFIsafe operating mode. You can find out the operating modes supported by the device from the range of values offered. The parameter is set to "1" (PROFIsafe V2 MODE) for fail-safe standard IO devices and cannot be changed.

For fail-safe DP standard slaves, you can set this parameter to the following:

- Set "F_Par_Version" to "1" (PROFIsafe V2 MODE) for a PROFIBUS DP-homogeneous network, if the device and the F-CPU support this. Otherwise, set it to "0" (PROFIsafe V1 MODE).
- "F_Par_Version" must be set to "1" (PROFIsafe V2-MODE) for a network composed of PROFIBUS DP and PROFINET IO subnets.

Note

The following F-CPUs support V2 MODE:

- CPU 416F-2, firmware version V4.1 and higher
- CPU 416F-3 PN/DP
- IM 151-7 F-CPU, firmware version V2.6 and higher
- CPU 315F-2 PN/DP
- CPU 315F-2 DP, firmware version V2.6 and higher
- CPU 317F-2 PN/DP
- CPU 317F-2 DP, firmware version V2.5 and higher
- CPU 319F-3 PN/DP

If you set "F_Par_Version" to "1" for F-CPUs that do not support V2 MODE, this will result in a communication error for the safety-related communication with the device. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "F-I/O passivated": Check value error (CRC)/Sequence number error ...
- "F-I/O passivated": Monitoring time for safety message frame exceeded ...

⚠ WARNING

"F_Par_Version" must be set to "1" (PROFIsafe V2-MODE) for a network composed of PROFIBUS DP and PROFINET IO subnets. Devices that do not support PROFIsafe V2 MODE must not be used on PROFINET IO or in hybrid configurations of PROFIBUS DP and PROFINET IO.

"F_Source_Add" and "F_Dest_Add" Parameters

The PROFIsafe addresses ("F_Source_Add" and "F_Dest_Add" parameters) uniquely identify the source and destination.

The "F_Source_Add" and "F_Dest_Add" parameters for fail-safe DP standard slaves and standard I/O devices correspond to the "F_source_address" and "F_destination_address" parameters of other F-I/O. Exception: The range of values is specified by the GSD file and is not limited to a range of 1 to 1022 for the PROFIsafe target address. Otherwise, the information for the PROFIsafe address assignment in Chapter "Configuring the F-I/O" applies.

"F_WD_Time" Parameter

This parameter defines the monitoring time in the fail-safe DP standard slave/standard I/O device.

A valid current safety message frame must arrive from the F-CPU within the monitoring time. This ensures that failures and faults are detected and appropriate reactions are triggered to maintain the F-system in the safe state or bring it to a safe state.

The selected monitoring time should be long enough to tolerate frame delays in communication, while ensuring that the fault reaction function has a sufficiently fast reaction when a connection is interrupted or some other fault occurs (see *Safety Engineering in SIMATIC S7* system manual).

The "F_WD_Time" parameter can be set in 1 ms increments. The value range of the "F_WD_Time" parameter is specified by the GSD file.

"F_iPar_CRC" Parameter

CRC via individual device parameters (i-parameter).

The individual device parameters (i-parameters) of a fail-safe DP standard slave/standard I/O device are configured with their own parameter assignment tool provided by the device manufacturer.

Enter here the CRC calculated by the parameter tool from the device manufacturer for the protection of the i-parameters. *S7 Distributed Safety* takes the value into account when calculating the CRC F-parameter (CRC1).

See also

Configuring the F-I/O (Page 36)

2.6 Assigning Symbolic Names

Symbolic Name for F-I/O DBs

During compilation in *HW Config*, an F-I/O DB is automatically created for each F-I/O, and a symbolic name is entered for the F-I/O DB in the symbol table.

The symbolic name is generated in each case by combining the prefix "F" with the start address of the F-I/O and the name (maximum of 17 characters) entered for the F-I/O module in the object properties in *HW Config* (for example, F00005_4_8_F_DI_24VDC). In so doing, any special characters included in the name are replaced with "_".

For F-I/O accessed via I-slave-slave communication, an X (stands for "Mode: F-DX module" = fail-safe I-slave-slave communication) is added after the start address of the F-I/O (e.g. F00005_X_4_8_F_DI_DC24V).

If a name other than the default name entered in the object properties for the F-I/O is to be adopted as the symbolic name, you must change the name in the object properties for the F-I/O **before** compiling for the first time in *HW Config* . Be aware that only the first 17 characters are entered in the symbolic name.

After compiling for the first time, you can only change the symbolic name as follows:

- By editing the symbolic name directly in the symbol table

  (Note that the maximum symbol length comprises 24 characters and that the symbolic name will no longer match the name in the object properties for the F-I/O.)

  Or

- By deleting the applicable symbol table entry, changing the name in the object properties, and then recompiling in *HW Config*.

Note

In the case of fail-safe DP standard slaves/standard I/O devices, take care not to use the "description" that can be entered in *HW Config* (instead of the name) for generating the symbolic name for the associated F-I/O DB. The symbolic name is always generated in this case using the prefix "F," the start address of the fail-safe DP standard slave/standard I/O device, and a fixed character string. You can change the symbolic name only be editing it directly in the symbol table.

⚠ WARNING

An F-I/O DB is always assigned to a particular F-I/O module using the F-I/O DB number, and not the start address entered by default in the symbolic name.

For this reason, you must not modify the automatically assigned numbers of the F-I/O DBs; otherwise, your safety program can no longer access the F-I/O DB assigned to the F-I/O.

Symbolic Names for Input Channels of the SM 336; AI 6 x 13Bit and SM 336; F-AI 6 x 0/4 ... 20 mA HART

If you want to assign symbols for the input channels of the SM 336; AI 6 x 13Bit or SM 336; F-AI 6 x ... 0/4 20 mA HART, make sure that the symbols are of data type INT.

See also

F-I/O DB (Page 98)

Access Protection

3

3.1 Overview of Access Protection

Introduction

Access to the S7 Distributed Safety F-system is protected by two password prompts: one for the F-CPU and another for the safety program.

For the password for the safety program, a distinction is made between an offline password and an online password for the safety program:

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

The following table presents 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 for the safety program.

	Password for F-CPU	Password for Safety Program
Assignment	In *HW Config*, during configuration of the F-CPU, Properties - CPU, "Protection" tab, appropriate protection level, e.g., "1: Access protection for F-CPU", and "Removable with Password" and "CPU Contains Safety Program" check boxes selected	• In *SIMATIC Manager*, **Options > Edit Safety Program > Permission** menu command • When the F-PB is opened for the first time • When F-FBs/F-FCs are opened for the first time • When F-DBs are opened for the first time • When the "Edit F-Runtime Groups" dialog is opened for the first time • When compiling for the first time In *HW Config* after a first-time save operation: • When F-I/O that are set to "safety mode" are arranged in the configuration table • When the "F parameters" tab in the object properties for the F-CPU is opened for the first time • When the object properties for an F-I/O is opened for the first time • When the "F-Configuration" tab in the object properties dialog for an I-slave is opened for the first time • When the "PROFIsafe" tab in the object properties dialog for a fail-safe DP standard slave/standard I/O device is opened for the first time • When parameters in the tabs and dialog boxes indicated above are changed • When an F-I/O or F-CPU is deleted from the configuration table

	Password for F-CPU	Password for Safety Program
Prompt	• When the safety program is downloaded in its entirety • When F-blocks with an F-attribute are downloaded and deleted	**Offline password:** • When F-blocks are downloaded in *SIMATIC Manager* • When compiling in the "Safety Program" dialog • During compilation with "Check block consistency" function • When the F-PB is opened • When F-FBs/F-FCs are opened • When F-DBs are opened • When know-how protection is set for user-created F-FBs, F-FCs, and F-DBs • When the "Edit F-Runtime Groups" dialog is opened • When the password is changed • When an F-I/O that is set to "Safety mode" is arranged in the configuration table • When the "F parameters" tab in the object properties is opened • When the object properties dialog for an F-I/O is opened • When the "PROFIsafe" tab in the object properties dialog for a fail-safe DP standard slave/standard I/O device is opened • When parameters in the tabs and dialog boxes indicated above are changed • When an F-I/O or F-CPU is deleted from the configuration table • When a HW configuration is saved and compiled (only the safety program is protected from changes) • When (new) F-blocks are created/inserted/moved in the offline block container of the safety program* • When F-blocks are saved • When F-blocks are renamed in the offline block container of the safety program* • When F-blocks are cut and deleted from the offline block container of the safety program* • When F-blocks are "rewired" • When write-protected F-blocks are stored • When the offline block container is deleted* • When the "S7 Programs" folder is deleted* • When object properties of F-blocks are opened* • When object properties of an F-block are edited* **Offline password** • When safety mode is deactivated (the password must always be entered, even if access permission to the safety program is still valid) • When data in the safety program are modified • When new F-blocks are created in the online block container of the safety program
Validity	Once the correct password has been entered, access is authorized until *SIMATIC Manager* is closed or permission is revoked using the **PLC > Access Rights > Cancel** menu command.	Once the correct password has been entered, access is authorized for 1 hour. For additional information, see Chapter "Access Permission for the Safety Program".
* Condition: The safety program in assigned to an F-CPU.		

3.2 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. In *SIMATIC Manager*, select the F-CPU or its S7 program.
2. Select the **Options > Edit Safety Program** menu command.

   The "Safety Program" dialog will appear.
3. Click "Permission..." and enter the password for the safety program in the "New password" and "Confirm password" fields in the "Set up permission for safety program" dialog.

If you have not yet assigned a password for the safety program but you perform an action that triggers the password prompt for the safety program (see table for assignment and prompting of passwords), the "Set up permission for safety program" dialog for assigning the password for the safety program is displayed automatically.

Note

Make sure that you use identical online and offline passwords for the safety program by downloading the safety program to the F-CPU with the "Safety Program" dialog, as otherwise you cannot download it by means of *SIMATIC Manager* and *LAD/FBD Editor*.

⚠ WARNING

To optimize access protection, you must use different passwords for the F-CPU and the safety program.

⚠ 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 password protection 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 safety program before leaving the programming device or PC. If this is not strictly implemented, a screen saver equipped with a password accessible only to authorized personnel must also be used.

Changing the Password for the Safety Program

The "Set up permission for safety program" dialog is also used to change a password for the safety program. This is done using the same procedure as in Windows by entering the old password and then entering the new password twice.

Procedure for Setting Up Access Permission for the Safety Program

To set up access permission for the safety program:

1. In *SIMATIC Manager*, select the F-CPU or its S7 program.
2. Select the **Options > Edit Safety Program** menu command.

   The "Safety Program" dialog will appear.
3. Click "Permission..." and enter the password for the safety program in the "Old password" field in the "Set up permission for safety program" dialog.

If you have not yet set up any access permission but you perform an action that triggers the password prompt for the safety program (see table for assignment and prompting of passwords), the "Password for safety program" dialog for assigning the password is displayed automatically.

Validity of Access Permission for a Safety Program

A set-up access permission for a safety program enables exclusive access by the Windows user who was the current user at the time of the setup . In addition, the access permission for a safety program acts only in the context of the project in which the safety program was located at the time of the setup. Once the correct password has been entered, access is authorized for 1 hour or until permission is revoked.

Within this hour, the validity period of the online password is reset to 1 hour with each online action triggering the password prompt (see table for assignment and prompting of passwords) and, likewise, the validity period of the offline password is reset to 1 hour with each offline action triggering the password prompt.

If the validity of an access permission expires and you are at that moment executing an action for which a password is required (e.g. editing an F-block), then you are prompted again for the current password when you save. If you do not enter a password, you cannot save the result of the action.

Revoking Access Permission for the Safety Program

You can revoke the access permission for the safety program in the "Set up permission for safety program" dialog by clicking the "Cancel" button.

You can also revoke the access permission for the safety program in the "Safety Program" dialog by clicking the drop-down arrow on the "Permission..." button.

The user will then be prompted to enter the password for the safety program again the next time an action requiring a password (see table for assignment and prompting of passwords) is performed. To "revoke" access permission when using the Modify function, the connection to the F-CPU must be terminated (for example, by closing the *STEP 7* applications).

The access permission for the safety program is reset automatically when all *STEP 7* applications that have *S7 Distributed Safety* opened (e.g., *SIMATIC Manager*, *FBD/LAD editor*) have been exited. If you reopen *STEP 7* after exiting these *STEP 7* applications and perform an action that requires a password, you are prompted to enter the password for the safety program again.

3.3 Read Accesses without Password for the Safety Program

Read Access without a Password

The "Password for Safety Program" dialog allows you to set up the password for the safety program.

Alternatively, you can opt for read accesses without a password.

You can use read access without a password to access F-relevant tabs and object properties dialogs for the F-CPU, F-I/O, F-blocks, and safety-related communication, F-blocks of the safety program, and the "Edit F-Runtime Groups" dialog.

"Password for Safety Program" Dialog

The "Password for Safety Program" dialog looks like this:

Read Access for all Other Actions

If you have specified read access for all other actions, the user is not prompted to enter the password for additional read accesses and is granted read-only access.

Exception: Read access is not permitted for the requested action and you would like to terminate read access.

Read access for all other actions is not time-limited. It applies only to the safety program for which it was activated and not to other safety programs on the same programming device or PC.

Terminating Read Access for all Other Actions

Read access for all other actions is terminated when you perform one of the following actions:

- You revoke access permission in the "Set Up Permission for Safety Program" dialog
- You revoke the access permission in the "Safety Program" dialog by clicking the drop-down arrow on the "Permission..." button
- You close all S7 applications that were processing the data of a safety program with read access for all other actions
- You have entered a password for the safety program for an action for which read access is not permitted (e.g., compiling the safety program)
- You restart the programming device or PC

Read-Only Access for this Access

If you have specified one-time read access, the user is prompted to enter a password again for the next read access and for all other actions requiring a password.

3.4 Access Permission for the F-CPU

Procedure for Assigning a Password for the F-CPU

You assign the password for the F-CPU when configuring the F-CPU (see Chapter "Configuring the F-CPU").

Changing the Password for the F-CPU

The password for the F-CPU can be changed only by modifying the configuration. You must switch the F-CPU to STOP mode to download the modified configuration.

Procedure for Setting up an Access Permission for the F-CPU

1. In *SIMATIC Manager*, select the F-CPU or its S7 program.
2. Select **PLC > Access Rights > Setup**. In the resulting dialog, enter the password for the F-CPU that you assigned when configuring the F-CPU in the "Protection" tab.

⚠ 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 PG/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. If this is not strictly implemented, a screen saver equipped with a password accessible only to authorized personnel must also be used.

After canceling access permission, you should check to determine whether the collective signature of all F-blocks with an F-attribute in the block container online is identical to the collective signature of all F-blocks with an F-attribute in the block container of the accepted safety program. If not, you must download the correct safety program to the F-CPU.

Validity/Revoking the Access Permission for the F-CPU

Access permission for the F-CPU is valid until *SIMATIC Manager* is closed, permission is revoked using the **PLC > Access Rights > Cancel** menu command, or the last S7 application is closed.

Transferring the Safety Program to Multiple F-CPUs

⚠WARNING

If **multiple F-CPUs** can be reached over a network (such as MPI) by **one programming device or PC**, you must take the following actions to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, e.g., a uniform password for the F-CPUs having the respective MPI address as an extension (max. 8 characters): "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 (analogous to assigning an MPI address to an F-CPU for the first time).
- Before downloading a safety program to an F-CPU for which access rights by means of an F-CPU password do not yet exist, you must first revoke existing access permission for any other F-CPU.

See also

Configuring the F-CPU (Page 26)

Programming

4

4.1 Overview of Programming

4.1.1 Overview of Programming

Introduction

A safety program consists of fail-safe blocks that you select from an F-library or create using the F-FBD or F-LAD programming languages and fail-safe blocks that are automatically added when the safety program is compiled. Fault control measures are automatically added to the safety program you create, and additional safety-related tests are performed.

Overview

This section contains a description of the following:

- Structure of the safety program in S7 Distributed Safety
- Fail-safe blocks
- Differences between the F-FBD/F-LAD programming languages and the standard FBD and LAD languages

Schematic Structure of a Project with Standard User Program and Safety Program

The figure below presents the schematic structure of a *STEP 7* project in the programming device/PC with a standard user program and a safety program for S7 Distributed Safety.

The *Distributed Safety* F-block library (V1) is supplied with the *S7 Distributed Safety* optional package for creating the safety program.

The F-library is located in the *step7/s7libs* directory.

Additional information about programming is provided in the following sections.

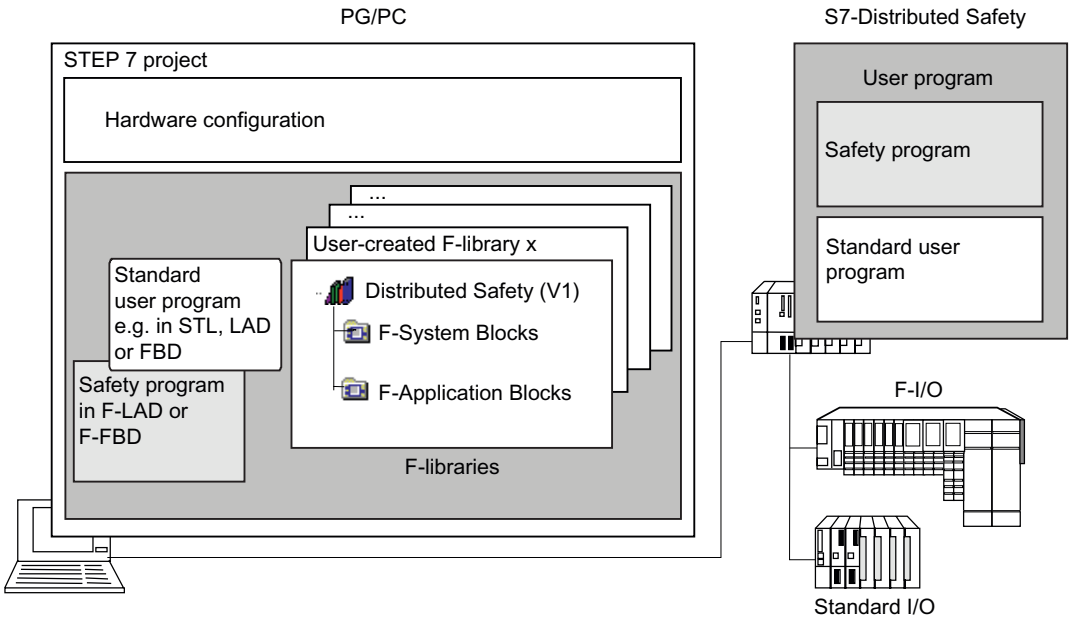


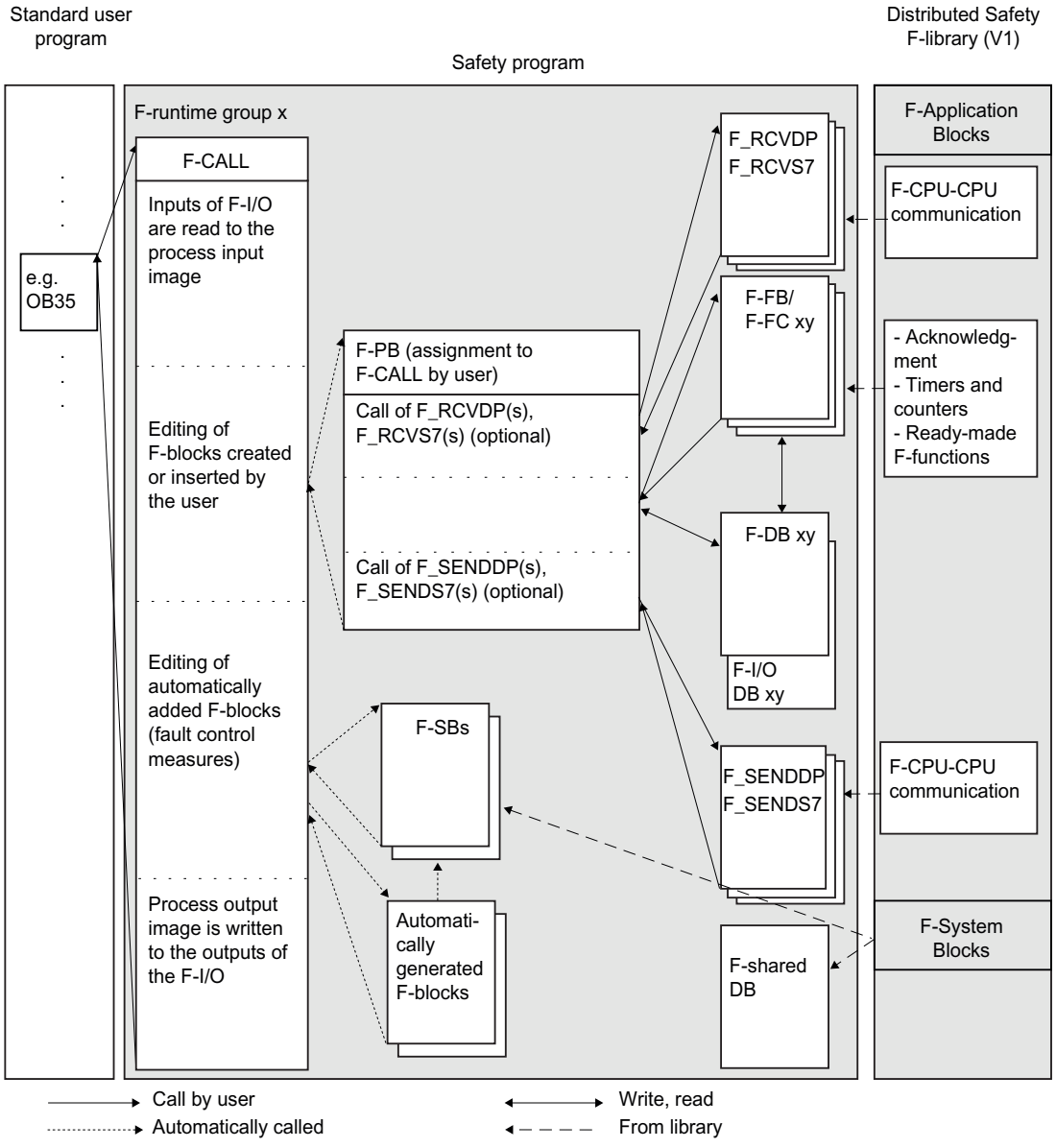
Figure 4-1 Configuration

4.1.2 Structure of the Safety Program in S7 Distributed Safety

Representation of Program Structure

The figure below shows the schematic structure of a safety program for *S7 Distributed Safety*. For structuring purposes, a safety program consists of one or two F-run-time groups. Each F-run-time group contains:

- F-blocks that you create or select from the *Distributed Safety* F-library (V1) or a user-created F-library
- F-blocks that are added automatically (F-system blocks, automatically generated F-blocks, and the F-shared DB)



Description of Program Structure

Entry into the safety program is made by calling F-CALL from the standard user program. Call the F-CALL directly in an OB, preferably in a cyclic interrupt OB (e.g., OB35).

The advantage of using cyclic interrupt OBs is that they interrupt the cyclic program execution in OB1 of the standard user program at fixed time intervals; that is, a safety program is called and executed at fixed time intervals in a cyclic interrupt OB.

Once the safety program is executed, the standard user program resumes.

F-Run-Time Groups

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

An F-run-time group consists of the following:

- One F-call block F-CALL
- One F-program block F-PB (an F-FB/F-FC that you assign to the F-CALL)
- Additional F-FBs or F-FCs that you program using F-FBD or F-LAD, as needed
- One or more F-DBs, as needed
- F-I/O DBs
- F-blocks of the *Distributed Safety* F-library (V1)
- F-blocks from user-created F-libraries
- F-system blocks F-SBs
- Automatically generated F-blocks

Structuring of the Safety Program in Two F-Run-Time Groups

You can divide your safety program into two F-run-time groups. By arranging for portions of your safety program (one F-run-time group) to run in a faster priority class, you achieve faster safety circuits with short response times.

See also

Rules for F-Run-Time Groups of the Safety Program (Page 86)

4.1.3 Fail-Safe Blocks

F-Blocks of an F-Runtime Group

You use the F-blocks in the table below in an F-runtime group:

F-Block	Function	Programming Language
F-CALL	F-block for calling the F-runtime group from the standard user program. The F-CALL contains the call for the F-program block and the calls for the automatically added F-blocks of the F-runtime group. You create the F-CALL, but you cannot edit it. It is possible to call the F-CALL in an OB or FB/FC that is called in an OB.	F-CALL
F-FB/F-FC, F-PB	You program the actual safety function using F-FBD or F-LAD. The starting point for F-programming is the F-program block. The F-PB is an F-FC or F-FB (with instance DB) that becomes the F-PB when assigned to the F-CALL. You can do the following in the F-PB: • Program the safety program with F-FBD or F-LAD • Call other created F-FBs/F-FCs for structuring the safety program • Insert F-blocks of the *F-Application Blocks* block container from the *Distributed Safety* F-library (V1). • Insert F-blocks from "user-created F-libraries" You define the call sequence of the F-blocks within the F-PB.	F-FBD/F-LAD
F-DB	Optional fail-safe data blocks that can be read/write accessed from within anywhere in the safety program	F-DB
F-I/O DB	An F-I/O DB is automatically created for each F-I/O during compilation in *HW Config*. You can or you must access the variables of the F-I/O DB in conjunction with F-I/O accesses.	–

F-Blocks of Distributed Safety F-Library (V1)

The *Distributed Safety* F-library (V1) contains:

- F-application blocks in the *F-Application Blocks*|*Blocks* block container
- F-system blocks and the F-shared DB in the *F-System Blocks*|*Blocks* block container

The F-blocks included in the block container are shown in the table below:

Block Container	... Purpose of F-Block	Function/F-Blocks
F-application blocks		This block container contains the F-application blocks that can be called by the user in the F-PB/F-FBs/F-FCs
	Safety-related CPU-CPU communication	F-application blocks for safety-related CPU-CPU communication: F_SENDDP, F_RCVDP, F_SENDS7, and F_RCVS7 for sending and receiving data during safety-related CPU-CPU communication
	Acknowledgment	F-application block F_ACK_OP for fail-safe acknowledgment by means of an operator control and monitoring system F-application block F_ACK_GL for a global acknowledgement of all F-I/O of an F-runtime group
	Timers and counters	F-application blocks F_TP, F_TON, F_TOF; F-application blocks F_CTU, F_CTD, F_CTUD
	Ready-made F-functions	F-application blocks for functions such as two-hand monitoring, muting, emergency STOP, safety door monitoring, and feedback loop monitoring
	Data conversion and scaling	F-application blocks F_SCA_I, F_BO_W, F_W_BO
	Copying	F-application blocks F_INT_WR, F_INT_RD
	Shift instructions	F-application blocks F_SHL_W, F_SHR_W
F-system blocks		This block container contains the F-system blocks (F-SBs) and the F-shared DB that are automatically inserted in the safety program
	F-system blocks	The F-system blocks (F-SBs) are automatically inserted by *S7 Distributed Safety* when the safety program is compiled in order to create an executable safety program from the user's safety program. You must not insert F-system blocks from the *F-System Blocks* block container in an F-PB/F-FB/F-FC. Likewise, you must not modify (rename) or delete F-system blocks in the *Distributed Safety* F-library (V1) or the block container of your user project.
	F-shared DB	Fail-safe block that contains all of the global data of the safety program and additional information needed by the F-system. When the hardware configuration is saved and compiled, the F-shared DB is automatically inserted and expanded. Using the symbolic name of the F-shared DB (i.e., F_GLOBDB), you can evaluate certain data of the safety program in the standard user program.

Note

A detailed description of the F-application blocks can be found in Chapter "Distributed Safety F-Library (V1)".

See also

F-I/O Access (Page 95)

Overview of Distributed Safety F-Library (V1) (Page 175)

Custom F-Libraries (Page 251)

4.1.4 Differences between the F-FBD and F-LAD programming languages and the standard FBD and LAD programming languages

Introduction

The user program in the F-CPU typically consists of a standard user program and a safety program. The standard user program is created in *STEP 7* using standard programming languages such as STL, LAD, or FBD.

The safety program for *S7 Distributed Safety* is programmed using F-FBD or F-LAD.

F-FBD and F-LAD Programming Languages

The F-FBD and F-LAD programming languages correspond fundamentally to the standard FBD/LAD languages. The standard *FBD/LAD Editor* in *STEP 7* is used for programming.

The primary differences between the F-FBD and F-LAD programming languages and their standard counterparts are limitations in the instruction set and in the data types and the address areas that can be used.

Supported Data and Parameter Types

The following elementary data types are supported in F-FBD/F-LAD:

- BOOL
- INT
- WORD
- TIME

Non-Permissible Data and Parameter Types

The following are **not** permitted:

- Elementary data types not listed above (for example, BYTE, DWORD, DINT, REAL)
- Complex data types (for example, STRING, ARRAY, STRUCT, UDT)
- Parameter types (for example, BLOCK_FB, BLOCK_DB, ANY)

Supported Address Areas

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

Note that you can only access data in F-FBD and F-LAD as follows:

- Data of data type BOOL, in bits
- Data of data type INT, in words
- Data of data type WORD, in words
- Data of data type TIME, in double words

This restriction does not apply when data are write-accessed from the standard user program (bit memory or process image of standard I/O).

Example: To access input channels of data type BOOL in the process input image of F-I/O, you must use the "input (bit)" unit.

For reasons of clarity, you should always access address areas in a safety program using symbolic names.

Address Area	Accessible Size Units:	S7 Notation	Description
Process input image			
• Of F-I/O			At the beginning of the F-runtime group (F-CALL), the F-CPU reads the inputs from the F-I/O and saves the values to the process input image. Input channels are read-only channels. Therefore, a transfer to IN_OUT parameters of an F-FB or F-FC is not permitted.
Channels of data type BOOL, such as digital channels	Input (bit)	I	Input channels of data type BOOL are read-only and can only be accessed using the "Input (bit)" unit. Access is not permitted, for example, with the "Input word" unit.

Address Area	Accessible Size Units:	S7 Notation	Description
Channels of data type INT (WORD), such as analog channels	Input word	IW	Input channels of data type INT (WORD) are read-only and can only be accessed using the "Input word" unit. Access to individual bits using the "Input (bit)" unit is not permitted.
• Of standard I/O			At the beginning of each OB 1 cycle, the F-CPU reads the inputs from the standard I/O and saves the values to the process input image. With the S7-400, also bear in mind the update times when using partial process images.
	Input (bit) Input word	I IW	Input channels of the standard I/O are read-only and can only be accessed using the indicated units. Therefore, a transfer to IN_OUT parameters of an F-FB or F-FC is not permitted. In addition, a process-specific validity check is required.
Process output image			
• Of F-I/O			In the F-PB, the safety program calculates the values for the outputs of the F-I/O and stores them in the process output image. At the end of the F-runtime group (F-CALL), the F-CPU writes the calculated output values to the outputs of the F-I/O. Output channels are write-only channels. Therefore, a transfer to IN_OUT parameters of an F-FB or F-FC is not permitted.
Channels of data type BOOL, such as digital channels	Output (bit)	A	Output channels of data type BOOL are write-only and can only be accessed using the "Output (bit)" unit. Access is not possible, for example, with the "output word" unit.
Channels of data type INT (WORD), such as analog channels	Output word	QW	Output channels of data type INT (WORD) are write-only and can only be accessed using the "Output word" unit. Access to individual bits using the "Output (bit)" unit is not permitted.
• Of standard I/O			In the F-PB, the safety program also calculates the values for the outputs of the standard I/O, if applicable, and stores them in the process output image. At the beginning of the next OB 1 cycle, the F-CPU writes the calculated output values to the outputs of the standard I/O. With the S7-400, also bear in mind the update times when using partial process images.
	Output (bit) Output word	A QW	Output channels of the standard I/O are write-only and can only be accessed using the indicated units. Therefore, a transfer to IN_OUT parameters of an F-FB or F-FC is not permitted.

Address Area	Accessible Size Units:	S7 Notation	Description
Bit memory	Bit memory (bit) Memory word	M MW	This area is used for data exchange with the standard user program. Memory can only be accessed using the indicated units. In addition, read access requires a process-specific validity check. For a memory bit, either read access or write access is possible in the safety program. Therefore, a transfer to IN_OUT parameters of an F-FB or F-FC is not permitted. Note that memory bits can only be used for connecting the standard user program and the safety program; they cannot be used as a buffer for F-data.
Data block			Data blocks store information for the program. They can either be defined such that all F-FBs, F-FCs, and F-PBs can access them (F-DBs) or assigned to a particular F-FB or F-PB (instance DB). They must be created with the "F-DB" programming language or as an instance DB of an F-FB or F-PB.
	Data bit Data word Data double word	DBX DBW DBD	Local data can only be accessed using the units corresponding to the data type in the declaration table.
Local data			This memory area accepts the temporary data of a block or an F-block while this block is being executed. The local data stack also provides memory for transferring block parameters and for saving intermediate results.
	Local data bit Local data word Local data double word	L LW LD	Local data can only be accessed using the units corresponding to the data type in the declaration table.

Non-Permissible Address Areas

Access using units other than those listed in the table above is **not** permitted, as is access to address areas not listed, in particular:

- Counters (fail-safe counters are implemented using F-application blocks from the *Distributed Safety* F-library (V1): F_CTU, F_CTD, F_CTUD)
- Timers (fail-safe timers are implemented using F-application blocks from the *Distributed Safety* F-library (V1): F_TP, F_TON, F_TOF)
- Data blocks of the standard user program
- Data blocks (F-DBs) using "OPN DI"
- Data blocks that were automatically added
  - Exception: certain data in the F-I/O DB and the F-shared DB of the safety program
- I/O area: Inputs
- I/O area: Outputs

Boolean Constants "0" and "1"

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

Local Data Address Area: Particularities

Note

Note when using the local data address area that the first access of a local data element in an F-PB, F-FB, or 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).

Initialization of a "local data bit" should be performed with the Assign ("=") instruction (F-FBD) or Output Coil ("--()") instruction (F-LAD). 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. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety program: internal CPU fault; internal error information: 404"

You can find out which address areas are possible for your F-CPU in the product information for the CPU you are using.

Address Areas for N, P, NEG, POS, S, R, SR; RS Instructions: Particularities

Note

The "process input image," "process output image," and "bit memory" address areas must not be used for edge memory bits of the RLO Edge Detection (N, P) or Address Edge Detection (NEG, POS) instructions or for the address of the Flip Flop (SR, RS) instructions.

If the "local data" address area is used for the edge memory bits of the RLO Edge Detection (N, P) or Address Edge Detection (NEG, POS) instructions or for the address of the Flip Flop (SR, RS), Set Output (S), or Reset Output (R) instructions, the local data bit must be initialized beforehand.

Supported Instructions

You can use the instructions listed in the table below in the safety program:

Instruction		Function	Description
F-FBD	**F-LAD**		
>=1	-	Bit logic instruction	OR logic operation
&	-	Bit logic instruction	AND logic operation
XOR	-	Bit logic instruction	EXCLUSIVE OR logic operation
---|	-	Bit logic instruction	Insert binary input
---o|	-	Bit logic instruction	Negate binary input
=	-	Bit logic instruction	Assign
-	---| |---	Bit logic instruction	Normally open contact
-	---| / |---	Bit logic instruction	Normally closed contact
-	---|NOT|---	Bit logic instruction	Invert power flow
-	---( )	Bit logic instruction	Output coil
#	---(#)---	Bit logic instruction	Midline output
S	---(S)	Bit logic instruction	Set output
R	---(R)	Bit logic instruction	Reset output
SR	SR	Bit logic instruction	Set-reset flip flop
RS	RS	Bit logic instruction	Reset-set flip flop
N	---(N)---	Bit logic instruction	Negative RLO edge detection
NEG	NEG	Bit logic instruction	Address negative edge detection
P	---(P)---	Bit logic instruction	Positive RLO edge detection
POS	POS	Bit logic instruction	Address positive edge detection
WAND_W	WAND_W	Word logic instruction	(Word) AND Word
WOR_W	WOR_W	Word logic instruction	(Word) OR Word
WXOR_W	WXOR_W	Word logic instruction	(Word) Exclusive OR Word
ADD_I	ADD_I	Integer function	Add integer
DIV_I	DIV_I	Integer function	Divide integer
MUL_I	MUL_I	Integer function	Multiply integer
SUB_I	SUB_I	Integer function	Subtract integer
CMP ? I	CMP ? I	Comparison instruction	Compare integer (CMP==I, CMP<>I, CMP>I, CMP<I, CMP>=I, CMP<=I)
NEG_I	NEG_I	Conversion instruction	Create twos complement integer (16 Bit)
OPN	---(OPN)	DB instruction	Open data block
MOVE	MOVE	Move instruction	Assign a value
CALL_FC (call FC as box)	CALL_FC (call FC as box)	Program control	Call F-FCs unconditionally (EN = 1, no interconnection of EN!)
CALL_FB (call FB as box)	CALL_FB (call FB as box)	Program control	Call F-FBs unconditionally (EN = 1, no interconnection of EN!)
vRET	---(RET)	Program control	Return (exit block)

Instruction		Function	Description
Call multiple instances	Call multiple instances	Program control	Call multiple instances
JMP	---(JMP)	Jump instruction	Unconditional jump in block Jump in block if 1 (conditional)
JMPN	---(JMPN)	Jump instruction	Jump in block if 0 (conditional)
OV	OV ---| |---	Status bit	Evaluate exception bit overflow (OV bit in status word)

S Instruction: Particularities

Note

The Set Output (S) instruction is only executed if it is applied to an output of an F-I/O that is passivated (e.g., during startup of the F-system). For this reason, you should only attempt to access outputs of F-I/O with the Assign ("=") (F-FBD) or Output Coil ("--( )") (F-LAD) instruction.

You can evaluate whether an F-I/O or channels of an F-I/O are passivated in the associated F-I/O DB.

S, R, SR, RS, N, NEG, P, POS Instructions: Particularities

Note

If you wish to use a formal parameter of an F-FB/F-FC for the edge memory bits of the RLO Edge Detection (N, P) or Address Edge Detection (NEG, POS) instructions or for the address of the Flip Flop (SR, RS), Set Output (S), or Reset Output (R) instructions, it must be declared as an in/out parameter.

The F-CPU can go to STOP if this caution is not observed. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety program: internal CPU fault; internal error information: 404"

ADD_I, SUB_I, MUL_I, NEG, DIV_I, OV Instructions: Particularities

Note

If the result of an ADD_I, SUB_I, MUL_I, or NEG_I instruction or the quotient of a DIV_I instruction is outside the permitted range for integers (16 bits), the F-CPU goes to STOP mode if the result/quotient is used in an output to an F I/O or to a partner F-CPU by means of safety-related CPU-CPU communication. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety program: internal CPU fault; internal error information: 404"

Therefore, you should take appropriate steps when programming to comply with the permissible range for integers (16 bits), or evaluate the OV bit.

A warning is issued if you have not programmed an OV bit scan for ADD_I, SUB_I, MUL_I, NEG_I, and DIV_I instructions.

By evaluating the OV bit, you can identify an overflow without the F-CPU going to STOP mode in the case of an overflow. The result/quotient behaves like the analogous instruction in a standard user program.

Note

An OV bit scan is only permitted in the network following the network with the instruction affecting the OV bit.

The network with the OV bit scan must not be the destination of a jump instruction; in other words, it must not contain a jump label.

If an OV bit scan is programmed in the network following the instruction affecting the OV bit, the execution time of the instruction affecting the OV bit is increased (see also *Excel File for Response Time Calculation s7fcotia.xls*).

Note

If the divisor (input IN2) of a DIV_I instruction = 0, the quotient of the division (result of division at output OUT) = 0. The result behaves like the corresponding instruction in a standard user program. The F-CPU does **not** go to STOP mode. This is the response regardless of whether an OV-bit scan is programmed in the next network.

OPN DB Instruction: Particularities

Note

Keep in mind when using the "OPN DB" 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 "OPN DB" 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.

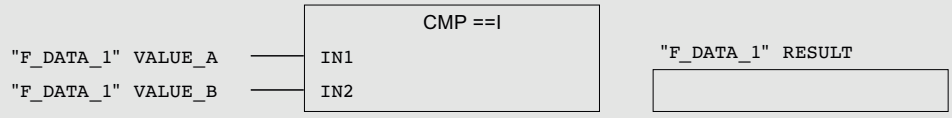
If you still want to use the "OPN DB" instruction, you must ensure that the DB register is restored by repeating the "OPN DB" instruction following calls of F-FB/F-FC and "fully qualified DB accesses." Otherwise, an error could result.

"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 "OPN DB" instruction. This also applies to the initial access to data of a data block after a jump label.

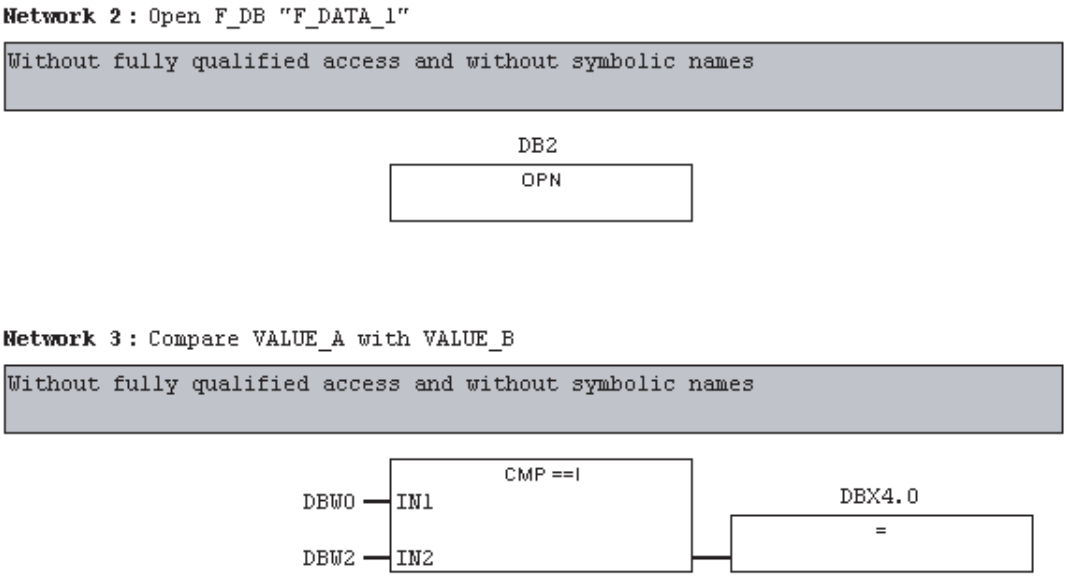
Example of "Fully Qualified DB Access":

```
FB5:  Title
Comment:

Network 1 : Compare VALUE_A with VALUE_B
With fully qualified access and with symbolic names.

You have to assign a symbolic name, for example "F_DATA_1" for the F-DB, and
instead of the absolute addresses, use the names assigned in the declaration of
the F-DB.
```




```
Symbol information
DB2.DBW0                "F_DATA_1" VALUE_A
DB2.DBW2                "F_DATA_1" VALUE_B
DB2.DBX4.0              "F_DATA_1" RESULT
```

Example of "Not Fully Qualified DB Access":



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 data in instance DBs of other F-FBs.

Make sure that "Report Cross References as Error" is not selected in the "General" dialog (**Options > Settings**) in the *FBD/LAD Editor*. Otherwise, instance DBs cannot be accessed.

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.

MOVE Instruction: Particularities

Note

The MOVE operation is permitted if the data types at the input and output are the same or between data with the INT and WORD data types.

For data from the standard user program, the length of the data types at the input and output must match.

Call Multiple Instances: Particularities

Note

You must not declare the F_SENDS7 and R_RCVS7 F-application blocks as multiple instances, even if they have the "multiple instance-capable" property.

Accesses to static data of a multiple instance within the F-FB in which the multiple instance is declared are not permitted.

Accesses to inputs and outputs of a multiple instance outside the F-FB in which the multiple instance is declared are not permitted.

JMP, JMPN, RET Instructions: Particularities

Note

You are not permitted to program an F_SENDDP or F_SENDS7 call between a jump instruction and the associated destination of the jump instruction.

You are not permitted to program a RET instruction prior to an F_SENDDP or F_SENDS7 call.

Non-Permissible Instructions

All instructions that are not listed in the table above are **not** permitted, in particular:

- Counter instructions (fail-safe counters are implemented using F-application blocks from the *Distributed Safety* F-library (V1): F_CTU, F_CTD, F_CTUD)
- Timer instructions (fail-safe timers are implemented using F-application blocks from the *Distributed Safety* F-library (V1): F_TP, F_TON, F_TOF)
- Shift and Rotate instructions (shift instructions are implemented using F-application blocks from the *Distributed Safety* F-library (V1): F_SHL_W, F_SHR_W)
- The following program control instructions:
  - Call standard blocks (FBs, FCs)
  - CALL: Call FC/SFC without parameters
  - Call F-FBs, F-FCs conditionally (interconnection of EN and EN = 0)
  - Call SFBs, SFCs

  Note

  In fail-safe programming, you must not interconnect, assign "0" to, or evaluate the enable input EN or the enable output ENO.

See also

F-I/O Access (Page 95)
Data Transfer from the Safety Program to the Standard User Program (Page 123)
Data Transfer from the Standard User Program to the Safety Program (Page 125)

4.2 Creating the Safety Program

4.2.1 Basic Procedure for Creating the Safety Program

Software Requirements

The software requirements are described in Chapter "Installing/Removing the *S7 Distributed Safety* V5.4 SP4 Optional Package".

Additional Requirements

- A project structure must be created in *SIMATIC Manager*.
- The hardware components of the project - in particular, the F-CPU and the F-I/O - must have been configured prior to programming.
- The safety program must be assigned to an F-CPU, such as a CPU 315F-2 DP.

Steps for Creating an S7 Distributed Safety Program

The primary steps for creating the safety program are as follows:

Step	Action	Reference
1	Save and compile hardware configuration in *HW Config* and download it to the F-CPU, if necessary.	Configuration
2	Define the program structure	Defining the Program Structure
3	Create F-FBs and F-FCs with the F-FBD or F-LAD programming language in *SIMATIC Manager*	Creating F-Blocks in F-FBD/F-LAD
4	Edit and save F-FBs and F-FCs in the *FBD/LAD Editor*	Creating F-Blocks in F-FBD/F-LAD
5	Specify one or two F-runtime groups: For each F-runtime group: • Assign a previously programmed F-FB or F-FC to the F-CALL of the F-runtime group (assignment causes F-FB or F-FC to become the F-PB) • If the F-PB is a function block, assign an instance DB	Defining F-Runtime Groups
	• Set the maximum cycle time of the F-runtime group	*Safety Engineering in SIMATIC S7* system manual
	• If one F-runtime group is to provide data for evaluation to another F-runtime group of the safety program, assign a DB for F-runtime group communication.	Defining F-Runtime Groups
6	Compile safety program in the "Safety Program" dialog	Compiling the Safety Program

Step	Action	Reference
7	Call F-CALL blocks directly in OBs (cyclic interrupt OBs, to the extent possible)	Defining F-Runtime Group
8	Download the entire user program (standard user program and safety program) to the F-CPU in the "Safety Program" dialog	Downloading the Safety Program

See also

Installing/Removing the S7 Distributed Safety V5.4 SP4 Optional Package (Page 17)

Overview of Configuration (Page 23)

Defining the Program Structure (Page 74)

Creating F-Blocks in FFBD/FLAD (Page 76)

Rules for F-Run-Time Groups of the Safety Program (Page 86)

Compiling Safety Program (Page 258)

Downloading the Safety Program (Page 260)

4.2.2 Defining the Program Structure

Structuring of the Safety Program in Two F-Run-Time Groups

You can divide your safety program into two F-run-time groups. By arranging for portions of your safety program (one F-run-time group) to run in a faster priority class, you achieve faster safety circuits with short response times.

Note

You can better structure your safety program by dividing it into two F-run-time groups. However, note that the following actions cannot be performed for individual F-run-time groups, but only for the safety program as a whole:

- Specifying a password for the safety program
- Compiling the safety program
- Downloading the safety program
- Deactivating safety mode
- Comparing safety programs
- Printing a safety program

The collective signatures are formed using all F-blocks of the safety program.

Rules for the Program Structure

You must keep the following rules in mind when designing a safety program for S7 Distributed Safety:

- F-blocks must not be called directly in an OB; rather, they must be inserted into one or two F-run-time groups.
- The safety program consists of one or two F-run-time groups, each with one F-CALL. A maximum of one F-program block can be assigned to each F-CALL.
- The channels of an F-I/O can only be accessed from one F-run-time group.
- Variables of the F-I/O DB of an F-I/O can only be accessed from one F-run-time group and only from the F-run-time group from which the channels of this F-I/O are accessed (if access is made).
- For optimal use of local data, you must call the F-CALL blocks (the F-run-time groups) directly in OBs (cyclic interrupt OBs, to the extent possible); you should not declare any additional local data in these cyclic interrupt OBs.
- Certain resources must be reserved for the safety program. This is done during configuration of the F-CPU in *HW Config* in the Object Properties dialog for the F-CPU. If you do not make any settings explicitly, meaningful default values are used.
- Create your program according to the general *STEP 7* rules. Consider, for example, the data flow.

Note

You can improve performance by writing section 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 test.

See also

Overview of Configuration (Page 23)

Differences between the F-FBD and F-LAD programming languages and the standard FBD and LAD programming languages (Page 61)

Rules for F-Run-Time Groups of the Safety Program (Page 86)

Safety Program Acceptance Test (Page 297)

4.3 Creating F-Blocks in F-FBD/F-LAD

4.3.1 Creating F-Blocks in F-FBD/FLAD

Overview

This section describes how to create a safety program in F-FBD or F-LAD using F-FBs, F-FCs, and/or F-DBs you have created. The basic procedure is the same as for the standard user program; therefore, only the deviations from programming a standard user program are presented below.

You will find an explanation of how F-blocks are represented in *SIMATIC Manager* in Chapter ""Safety Program" Dialog".

Creating Individual F-Blocks without Assignment to an F-CPU

Note

It is possible to create individual F-blocks directly in an S7 program that is not assigned to any F-CPU. This allows you to create safety programs for different F-CPUs irrespective of the hardware used. However, keep in mind that F-addresses and the validity of F-I/O accesses are not checked in this case.

See also

"Safety Program" Dialog (Page 253)

4.3.2 Creating and editing F-FB/F-FC

Procedure for Creating and Editing an F-FB/F-FC

1. Go to the block container of *SIMATIC Manager*, and select the **Insert > S7 Block > Function** (or function block) menu command. You can also use the "Insert New Object" shortcut menu.

   Note

   You must not use the FB numbers in the band of numbers you reserved for automatically added F-function blocks ("F-function blocks" parameter in the object properties for the F-CPU).

2. In the "General - Part 1" tab of the "Properties - Function" window, enter the name of the F-FB/F-FC. Select "F-FBD" or "F-LAD" as the programming language. Click "OK" to confirm. Enter the password for the safety program (password prompts will no longer be mentioned in operating procedures below).

   The block symbol displayed in *SIMATIC Manager* is highlighted in yellow.

   The created F-block can then be opened and edited with the *FBD/LAD Editor*.

3. Double-click the F-FB/F-FC in *SIMATIC Manager*. The *FBD/LAD Editor* is displayed.

4. You should select "Type Check of Addresses" in the "LAD/FBD" dialog in the *FBD/LAD Editor* (**Options > Settings**).

   Note

   Only the following elements are displayed in the *F-Program Elements Catalog*:
   - Supported instructions
   - F-FBs and F-FCs from the block container of your S7 program
   - F-blocks from F-libraries, e.g., F-application blocks of *Distributed Safety* F-library (V1)
   - Multi-instances of the edited F-block

5. Edit your F-FB/F-FC block.

   The data types are checked during editing. Any errors detected are output in the *FBD/LAD Editor*, same as when creating a standard user program.

   Note

   An F-FB/F-FC called in the F-CALL (which then becomes the F-PB) cannot have any parameters because they cannot be initialized (see Defining F-runtime Groups).

   Note

   F-FBs/F-FCs must not call themselves.

   Note

   When switching from F-FBD to F-LAD, graphic representation of certain F-FBD networks might not be possible in F-LAD; rather, these networks are displayed in STL. The STL code they contain must not be changed.

   Rule: STL networks are not permitted in the F-FBD representation. STL networks in F-LAD must be represented again as F-FBD networks when there is a switch to F-FBD.

⚠WARNING

Editing the instance DB of F-FBs is not permitted online or offline and can cause the F-CPU to go to STOP mode.

Note

Accesses to static parameters of instance DBs of other F-FBs are not permitted.

Note

Note when using F-FCs that the first access of output parameters of F-FCs must be a write access. This initializes the output parameters. Output parameters from F-FCs must always be initialized.

The F-CPU can go to STOP if this is not observed. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety program: internal CPU fault; internal error information: 404"

Note

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

Note

Variable names in F-FBs/F-FCs can contain a maximum of 22 characters.

Note

Note that access to the input parameters in an F-FB/F-FC is read-only, while access to the output parameters is write-only.

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

6. Save the F-FB/F-FC block.

Note

When an F-FBD/F-LAD block is saved in the *FBD/LAD Editor,* only a local consistency check is performed for the F-block. A safety program is not yet generated.

Note

Occasionally, certain networks that you have edited in F-FBD are represented in STL (for example, upstream interconnections with edge memory bits and branches) when you try to save the F-block. Such F-blocks cannot be saved. You must delete the STL network and replace the upstream interconnection with your own networks, in which you direct the upstream interconnection to a temporary variable. You can then use this temporary variable as an address.

Note

For greater clarity, assign unique symbolic names to the F-FBs/F-FCs you have created. These symbolic names appear in the "Details" view of *SIMATIC Manager*, in the "Safety Program" dialog, and in the symbol table. Symbolic names are assigned in the same way as in standard programming.

See also

Configuring the F-CPU (Page 26)

Overview of Access Protection (Page 47)

Differences between the F-FBD and F-LAD programming languages and the standard FBD and LAD programming languages (Page 61)

Compiling Safety Program (Page 258)

4.3.3 Creating and Editing F-DB

F-DBs

Similarly to F-FBs or F-FCs, you can also create and edit F-DBs (with the F-DB programming language) whose parameters can be read/write accessed within one F-runtime group of the safety program.

The data types are checked during editing. Any errors detected are output in the *FBD/LAD Editor*, same as when creating a standard user program.

Note

You must not use the DB numbers in the band of numbers you reserved for automatically added F-data blocks ("F-data blocks" parameter in the object properties for the F-CPU; see Chapter "Configuring the F-CPU").

Note

When an F-DB is saved in the *FBD/LAD Editor,* only a local consistency check is performed for the F-block. A safety program is not yet generated.

Note

For greater clarity, assign unique symbolic names to the F-DBs you have created. These symbolic names appear in the "Details" view of *SIMATIC Manager*, in the "Safety Program" dialog, and in the symbol table. Symbolic names are assigned in the same way as in standard programming.

Variable names in F-DBs can contain a maximum of 22 characters.

Options for Data Blocks: "Unlinked" and "DB is Write-Protected in the PLC"

Note

The available option "Unlinked" in the object properties for a DB must not be set for F-DBs and instance DBs of F-blocks.

The available option "DB is write-protected in the PLC" in the object properties for a DB must not be set for F-DBs and instance DBs of F-blocks.

If you have selected either of these options, the selection will be corrected when the safety program is compiled.

F-Communication DB for Safety-Related CPU-CPU Communication via S7 Connections

For safety-related CPU-CPU communication via S7 connections, you must create an F-communication DB on the sender side and another on the receiver side. F-communication DBs are F-DBs that you create and edit in the same way as other F-DBs in *SIMATIC Manager*.

Special requirements for F-communication DBs are described in Chapter "Programming Safety-Related CPU-CPU Communication via S7 Connections".

DB for F-Runtime Group Communication

For safety-related communication between F-runtime groups of a safety program, you must create a "DB for F-runtime group communication" for each F-runtime group that is to provide data for another F-runtime group.

The procedure for creating DBs for F-runtime group communication and the special requirements for these DBs are described in Chapter "Defining F-Runtime Groups".

See also

Creating and editing F-FB/F-FC (Page 77)

4.3.4 Know-How Protection for User-Created F-FBs, F-FCs, and F-DBs

Know-How Protection

A block with know-how protection is a protected block that cannot be edited.

You can furnish user-created F-FBs, F-FCs, and F-DBs (except instance DBs) with know-how protection.

The protected F-FBs/F-FCs/F-DBs can no longer be modified.

You can read the block properties of protected F-FBs/F-FCs/F-DBs, but the instruction portion remains hidden.

Using Know-How Protection

Use know-how protection if you want to protect the knowledge contained in an F-FB/F-FC/F-DB or if you want to prevent unintentional manipulation of the F-FBs, F-FCs, and F-DBs (except instance DBs).

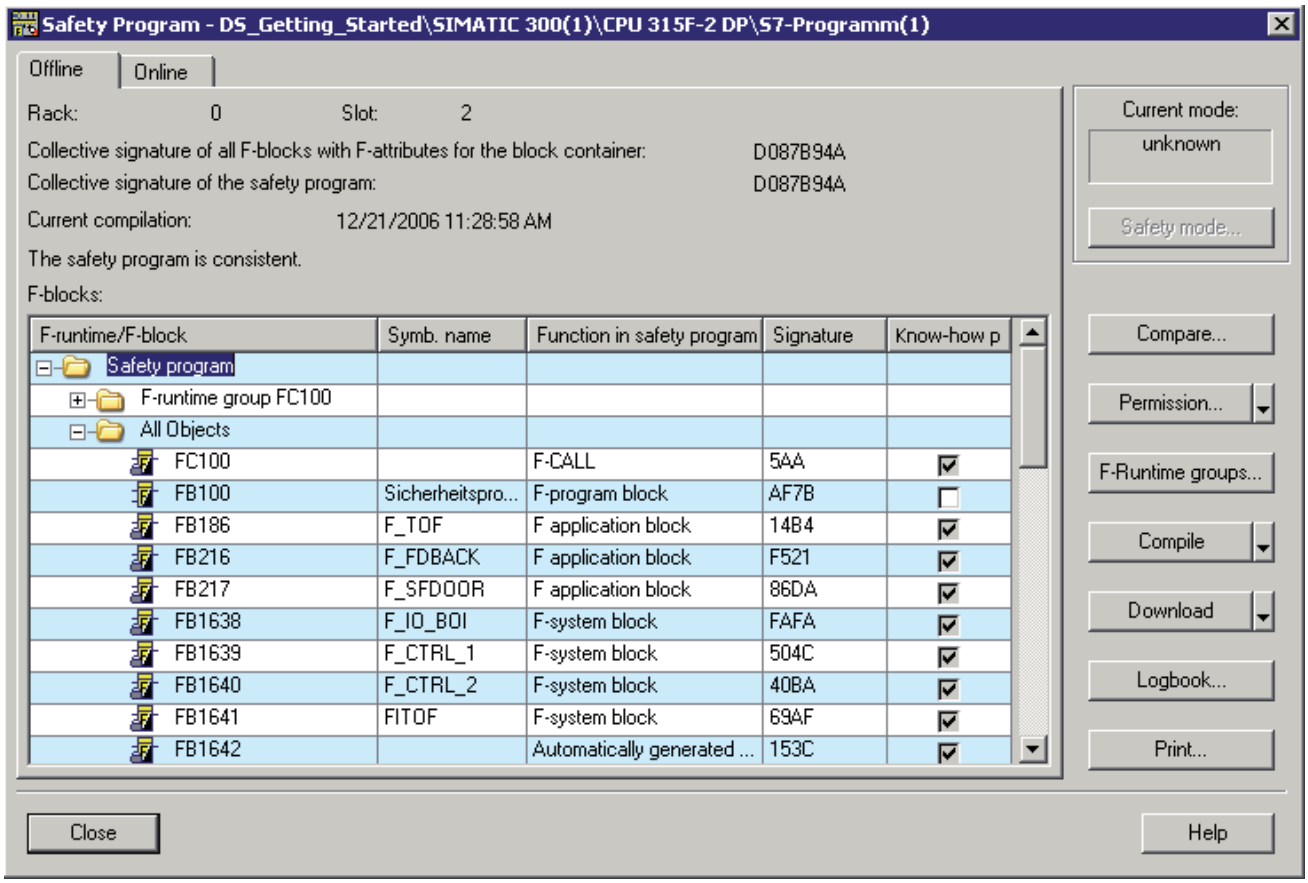
Requirements

You have created F-FBs, F-FCs, or F-DBs whose know-how you want to protect. The F-FBs/F-FCs/F-DBs you want to protect are not open in the *FBD/LAD Editor*.

Procedure for Setting Know-How Protection

Follow the steps outlined below:

1. Open the "Safety Program" dialog in *SIMATIC Manager*.
2. You set know-how protection for F-FBs/F-FCs/F-DBs in the offline safety program. For this purpose, select the "Offline" tab.



3. Select the relevant check box for the F-FBs, F-FCs, and F-DBs in the "Know-how protection" column.

   Result: A dialog for creating a backup copy opens automatically for every F-FB/F-FC/F-DB you want to protect.

4. Remember the following when you save the backup copy:

   Note

   Assign the name to the backup copy explicitly, so that you can relate the F-FB/F-FC/F-DB to the protected F-FB/F-FC/F-DB later (e.g., same name, comments regarding F-FB/F-FC/F-DB).

   Do not store the backup copy in the project containing the protected F-FB/F-FC/F-DB (otherwise, a non-protected copy of the F-FB/F-FC/F-DB will be available).

   If you want to store the backup copy in an F-library, make sure that the F-library is a user-created F-library in *S7 Distributed Safety*. The *FBD/LAD Editor* displays only F-libraries for *S7 Distributed Safety*.

5. Save the backup copy of the F-FB/F-FC/F-DB.

   Result: The check box in the "Know-how protection" column of the "Safety Program" dialog is selected and cannot be cleared.

   The block symbol in the "Block" column is shown with a padlock. The F-FB, F-FC, or F-DB is protected.

6. Follow the same procedure until all the F-FBs/F-FCs/F-DBs you want to protect are protected.

Modifying Protected F-FBs/F-FCs/F-DBs

Note

You cannot cancel the know-how protection of F-FBs/F-FCs/F-DBs.

If you want to modify a protected F-FB/F-FC/F-DB, proceed as follows:

1. Delete the protected F-FB/F-FC/F-DB from your project.
2. Copy the backup copy of the F-FB/F-FC/F-DB into your project.
3. Edit the unprotected F-FB/F-FC/F-DB in the *FBD/LAD Editor*.
4. If required, set know-how protection for the F-FB/F-FC/F-DB (see above).

See also

Custom F-Libraries (Page 251)

4.3.5 "Check Block Consistency" Function for User-Created F-FBs, F-FCs, and F-DBs

"Check Block Consistency" Function

The "Check block consistency" function can be found in *SIMATIC Manager* in the "Edit" menu, if you have selected a block container.

The "Check block consistency" function rectifies many of the time stamp conflicts and block inconsistencies. You can use this function in your safety program for F-FBs, F-FCs, and F-DBs without know-how protection. The procedure is the same as in standard programming. The "Go To" functionality is not supported.

You can select the **Program > Compile** and **Program > Compile All** menu commands for the "Check block consistency" function.

The complete safety program is then compiled as follows:

- If you select **Program > Compile**, the safety program is recompiled only if it was changed.
- If you select **Program > Compile All**, the safety program is recompiled regardless of whether it was modified.

4.3.6 "Compile and Download Objects" Function

"Compile and Download Objects" Function

You cannot use the "Compile and download objects" function in *SIMATIC Manager* to compile safety programs or download them to the F-CPU.

4.3.7 "Store Write-Protected" Function for User-Created F-FBs, F-FCs, and F-DBs

Storing an F-Block as a Write-Protected Block

You can use the "Store write-protected" function for F-blocks. If you execute the **File > Store write-protected** menu command for the F-block currently open in the FBD/LAD Editor, a write-protected copy of the F-block is created in any block container.

4.3.8 "Rewiring" Function for F-FBs and F-FCs

"Rewiring" Function

You can use the *STEP 7* "Rewiring" function for F-FBs and F-FCs in the offline safety program.

After successful rewiring, an appropriate entry is made in the logbook of the safety program.

The automatic consistency tests that are performed when F-blocks are saved are not performed for "Rewiring". A consistent safety program is not generated.

⚠ WARNING
"Rewiring" of F-blocks constitutes a change in the safety program and, thus, causes the collective signature to change. For this reason, the safety program must undergo acceptance testing again.

4.4 Defining F-Runtime Groups

4.4.1 Rules for F-Runtime Groups of the Safety Program

Requirements

You must have created your safety program.

Rules

⚠WARNING

Note the following:

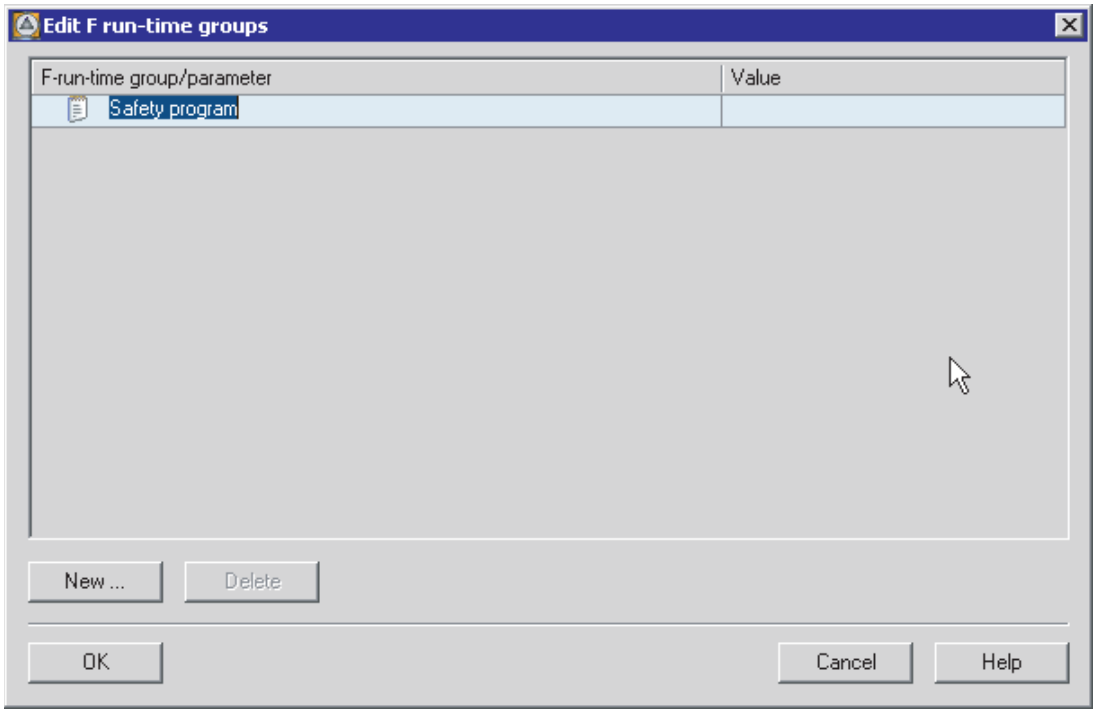
- The channels of an F-I/O can only be accessed from one F-run-time group.
- Variables of the F-I/O DB of an F-I/O can only be accessed from one F-run-time group and only from the F-run-time group from which the channels of this F-I/O are accessed (if access is made).
- An F-program block must not be used in more than one F-run-time group.
- F-FBs can be used in more than one F-run-time group but they must be called with different instance DBs.
- Instance DBs can only be accessed from the F-run-time group in which the associated F-FB is called.
- Individual parameters of F-DBs (except the F-shared DB) can only be used in one F-run-time group (however, an F-DB can be used in more than one F-run-time group).
- A DB for run-time group communication can be read- and write-accessed by the F-run-time group for which you furnished the DB, but only read-accessed by the "receiver" F-run-time group.
- The F-communication DB can only be accessed from one F-run-time group.

- F-blocks must not be called directly in an OB; rather, they must be inserted into one or two F-run-time groups.
- For optimal use of local data, you must call the F-CALL blocks (the F-run-time groups) directly in OBs (cyclic interrupt OBs, to the extent possible); you should not declare any additional local data in these cyclic interrupt OBs.
- Within a cyclic interrupt OB, the F-CALL (the F-run-time group) should be executed **before** the standard user program; that is, it should be at the very beginning of the OB, so that the F-run-time group is always called at fixed time intervals, regardless of how long it takes to process the standard user program.
- An F-CALL can only be called once. Multiple calls are not permitted and can cause the F-CPU to go to STOP mode.
- The process input and output images from standard I/O and memory bits can be accessed from more than one F-run-time group.
- F-FCs can generally be called in more than one F-run-time group.

4.4.2 Procedure for Defining an F-Runtime Group

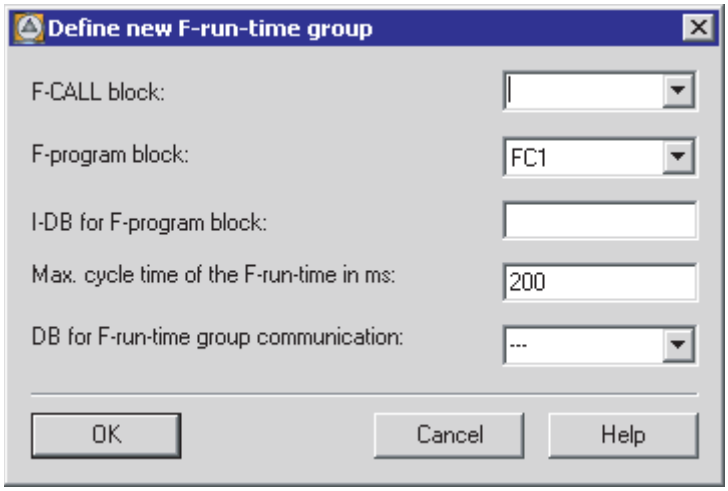
Procedure

1. In *SIMATIC Manager,* select the **Options > Edit Safety Program** menu command. The "Safety Program" dialog will appear. Activate the "F-Runtime Groups..." button to open the "Edit F-Runtime Groups" dialog.



2. In the "Edit F-Runtime Groups" dialog, select "New...".

   The "Define New F-Runtime Group" dialog is displayed.



3. From the drop-down list, select the FC that you want to define as the F-CALL for the new F-runtime group, or specify another FC. This FC is automatically created as soon as you exit the "Edit F-Runtime Groups" dialog with "OK".

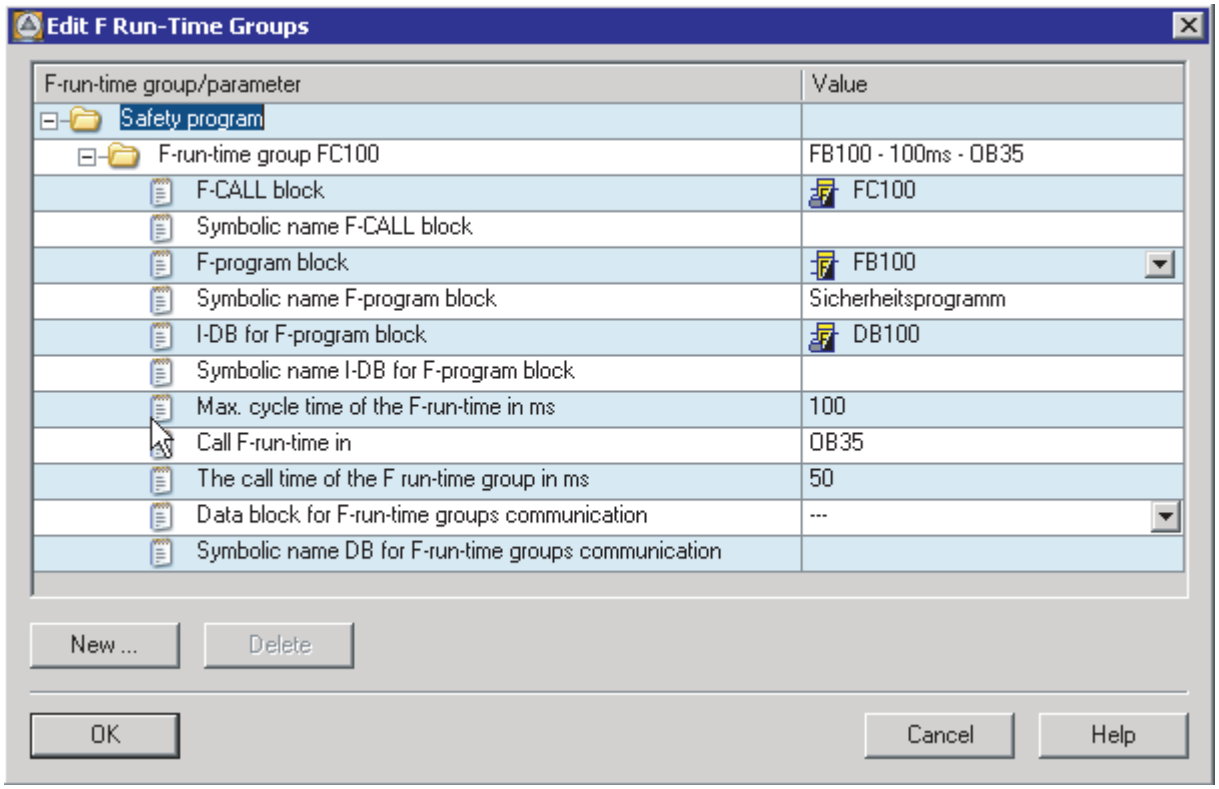
4. Define the F-program block of the F-runtime group by selecting the F-FB or F-FC from the drop-down list that you want to define as the F-PB for the new F-runtime group (symbolic entry possible). Only F-FBs/F-FCs without parameters can be specified. If the block to be assigned is an F-block of type "FB", you must specify an instance DB (e.g., "DB10") for "I-DB for F-program block" (symbolic entry possible). This I-DB is automatically created as soon as you exit the "Edit F-Runtime Groups" dialog with "OK". The number of the I-DB must not come from the range reserved in *HW Config*. If you specify an existing I-DB, it must be suitable for the selected F-program block.
5. The F-CPU monitors the F-cycle time in the F-runtime group. For "Max. Cycle Time of F-Runtime Group in ms", enter the maximum permissible time between two calls of this F-runtime group (maximum of 120,000 ms); see *Safety Engineering in SIMATIC S7* system manual.

⚠ WARNING

The F-runtime group call interval is monitored relative to the maximum value; that is, monitoring is performed to determine whether the call is executed often enough, but not whether it is executed too often. For this reason, fail-safe timers must be implemented using F-application blocks from the *Distributed Safety* F-library (V1) and not counters (OB calls).

6. If this F-runtime group is to provide data to another F-runtime group, select an F-DB for "DB for F-runtime group communication" from the drop-down list or specify another F-DB (symbolic entry possible). This F-DB is automatically created as soon as you exit the "Edit F-Runtime Groups" dialog with "OK".

   After the "OK" button is activated, the entries in the "Edit F-Runtime Groups" dialog undergo an internal validity check and are then applied.



   This dialog also displays:

   – The symbolic names of the newly defined F-blocks
   – The block of the standard user program in which the F-runtime group is called
   – Call time for the F-runtime group

   That is the execution time of the cyclic interrupt OB in which the F-CALL is called. You configured this time in *HW Config* (object properties for the F-CPU, "Cyclic interrupts" tab, "Execution time" parameter of the corresponding OB).

7. Repeat steps 2 to 6 to create a second F-runtime group.
8. Once the "OK" button is activated in the "Edit F-Runtime Groups" dialog, the entries are saved and, following a prompt, any non-existing F-blocks are automatically created.

4.4.3 Safety-Related Communication between F-Runtime Groups of a Safety Program

Safety-Related Communication between F-Runtime Groups

Safety-related communication can take place between the two F-runtime groups of a safety program. That is, fail-safe data that are provided by one F-runtime group in an F-DB are read in another F-runtime group.

You have the following options for creating the "DB for F-runtime group communication":

- In the "Define New F-Runtime Group" dialog
- In the "Edit F-Runtime Groups" dialog
- In *SIMATIC Manager* (see "Creating a DB for F-Runtime Group Communication in *SIMATIC Manager*" below)

Note

A DB for F-runtime group communication can be read- and write-accessed by the F-runtime group for which you furnished the F-DB, 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 data as possible are exchanged between the F-runtime groups.

Creating a DB for F-Runtime Group Communication in SIMATIC Manager

You can create the DB for F-runtime group communication in *SIMATIC Manager* in the same way as other F-DBs (see Chapter "Creating and Editing an F-DB").

Note the following when creating the DB for F-runtime group communication in *SIMATIC Manager*:

Assign the "RTG_DB" identifier in the "Family" box in the "General - Part 2" tab of the object properties for the F-DB. This identifier designates the F-DB as a DB for F-runtime group communication. Assign a symbolic name for the DB for F-runtime group communication.

Up-to-Dateness of Data When Reading from Another F-Runtime Group

Note

The data read from another F-runtime group are as up-to-date as they were when the F-runtime group furnishing the data was last processed before the start of the F-runtime group reading the data.

If the furnished data undergo multiple changes while the F-runtime group furnishing the data is being processed, the F-runtime group reading the data always receives the last change.

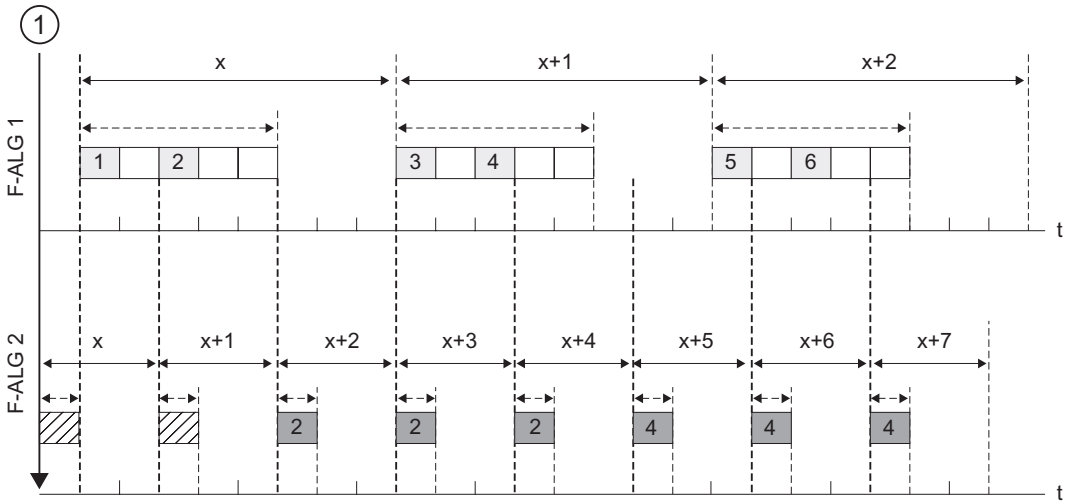
Assignment of fail-safe values

After a startup of the F-system, fail-safe values are made available to the F-runtime group having read access to data in the DB for F-runtime group communication of another F-runtime group (for example, F-runtime group 2). The values you specified in the DB for F-runtime group communication of F-runtime group 1 are made available as fail-safe values (presetting of the DB for F-runtime group communication).

F-runtime group 2 reads the fail-safe values the first time it is called. The second time F-runtime group 2 is called, it reads the latest data 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.

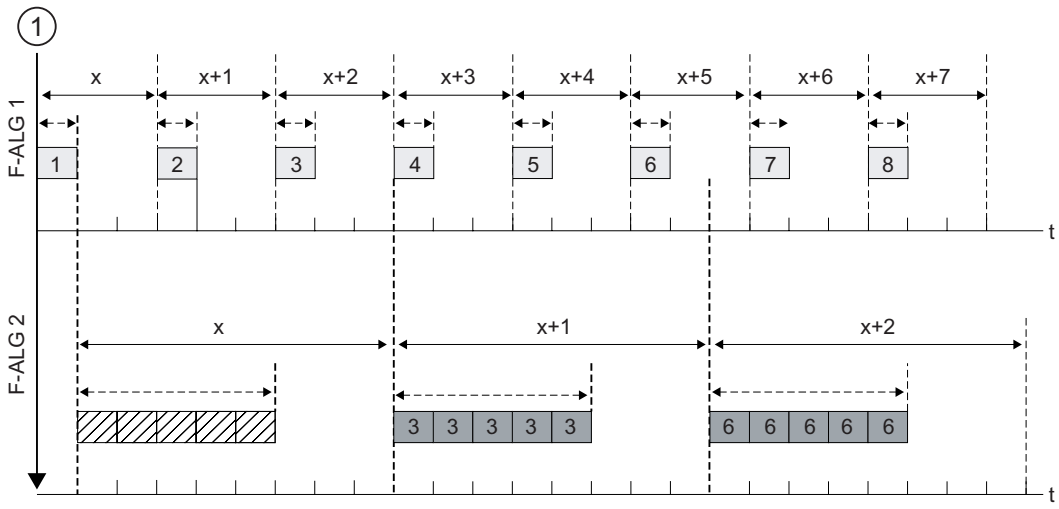
The behavior is illustrated in the two figures below.

Reading data from F-runtime group 1 that has a longer OB cycle and lower priority than F-runtime group 2



(1)	Startup of F-system
←→	Cycle time of the OB in which the F-runtime group is called.
←→	Runtime of the F-runtime group
1	... Data of F-runtime group 1, written to DB for F-runtime group communication of F-runtime group 1
2	Data of F-runtime group 2, read in DB for F-runtime group communication of F-runtime group 1
▨	Presetting in the DB for F-runtime group communication

Reading of data from F-runtime group 1 that has a shorter OB cycle and higher priority than F-runtime group 2



(1)	Startup of F-system
←→	Cycle time of the OB in which the F-runtime group is called.
←→	Runtime of the F-runtime group
1	... Data of F-runtime group 1, written to DB for F-runtime group communication of F-runtime group 1
2	... Data of F-runtime group 2, read in DB for F-runtime group communication of F-runtime group 1
	Presetting in the DB for F-runtime group communication

F-runtime group providing the data is not processed

Note

If the F-runtime group whose DB for F-runtime group communication supplies the data to be read is not processed (F-CALL of the F-runtime group is not called in an OB or FB), 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 F-CALL block (of F-runtime group that is not processed)
- Current cycle time in ms: "0"

See also

Creating and Editing F-DB (Page 79)

Procedure for Defining an F-Run-Time Group (Page 87)

4.4.4 Deleting F-Run-Time Groups

Deleting F-Run-Time Groups

1. In the "Edit F-Run-Time Groups" dialog, select the folder of the F-run-time group to be deleted.
2. Activate the "Delete" button.
3. Close the dialog with "OK."

The assignment of the F-blocks to an F-run-time group is deleted. However, the F-blocks continue to exist.

Note

If you want to delete your safety program, delete all yellow-highlighted F-blocks offline in *SIMATIC Manager*.

See also

Procedure for Defining an F-Run-Time Group (Page 87)

4.4.5 Changing F-Run-Time Groups

Changing F-Run-Time Groups

You can make the following changes for each F-run-time group of your safety program in the "Edit F-Run-Time Groups" dialog:

- Define a different FB/FC as the F-program block (select an FB/FC from the drop-down list)
- Enter a different or new I-DB for the F-program block
- Change the value of the maximum cycle time of the F-run-time group
- Define a different F-DB as the data block for F-run-time group communication (select an F-DB from the drop-down list or enter a new one)

Once the "OK" button is activated, the changes are saved and, following a prompt, any non-existing F-blocks are created automatically.

See also

Procedure for Defining an F-Run-Time Group (Page 87)

4.5 Programming Startup Protection

Introduction

⚠WARNING

When an F-CPU is switched from STOP to RUN mode, the standard user program starts up in the normal way. When the safety program is started up, all data blocks with an F-attribute 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.

A data handling 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: Process data outputs must be blocked until manually enabled. These outputs must not be enabled until it is safe to do so and faults have been corrected.

Example of Restart/Startup Protection

In order to apply restart/startup protection, it must be possible to detect a startup. To detect a startup, you declare a variable of data type BOOL with an initial/actual value of "1" in an F-DB.

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

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

See also

F-I/O DB (Page 98)

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Implementing User Acknowledgment in the Safety Program of a I-Slave F-CPU (Page 120)

F-I/O Access

5

5.1 F-I/O Access

Overview

This section describes how to 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, F-I/O (e.g., S7-300 F-SMs) are accessed via the **process image** (PII and PIQ). Direct I/O access is not permitted. The channels of an F-I/O can only be accessed from one F-runtime group.

The process input image is updated at the start of the F-runtime group, before the F-program block is processed. The process output image is updated at the end of the F-runtime group, after the F-program block is processed (see figure in Chapter "Structure of Safety Program in *S7 Distributed Safety*").

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

Due to the special safety protocol, the F-I/O occupy a larger area of the process image than is required for the channels that are actually present on the F-I/O. To find out the area of the process image where the channels (user data) 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 are permitted to be accessed.

Note that for certain F-I/O (such as S7-300 F-SMs and ET 200S 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.

Signal Charts

The signal charts presented in the "Signal Chart ..." figures in the following sections represent typical signal charts for the indicated behavior.

Actual signal charts and, in particular, the relative position of the status change of individual signals can deviate from the given signal charts within the scope of known distortion for cyclic program execution, depending on the following:

- Which F-I/O are being used
  (F-I/O with inputs, F-I/O with outputs, F-I/O with inputs and outputs, S7-300 F-SMs, ET200S F-modules, ET 200eco F-modules, ET 200pro F-modules, or fail-safe DP standard slaves/standard I/O devices, version of PROFIsafe bus profile for the F-I/O and F-CPU)
- The cycle time of the 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 charts refer to the status of signals in the user's safety program. If the signals are evaluated in the standard user program before or after the safety program is called in the same OB, the status change of the signals can be displaced by one cycle.

Contrary to what is shown in the signal charts, status changes between process data and fail-safe values that are transmitted to the fail-safe outputs ("To Outputs" signal chart) can occur before the status change of the associated QBAD signal, if necessary. The timing of the status change is dependent on whether F I/O with outputs or F I/O with inputs and outputs were used.

See also

Structure of the Safety Program in S7 Distributed Safety (Page 57)

F-I/O Access for Safety-Related I-Slave-Slave Communication (Page 163)

5.2 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 of data type BOOL and (analog) channels of data type INT (WORD), as follows:

- 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 Output for F-I/O/Channels of an F-I/O

In the case of **F-I/O with inputs**, when **passivation** occurs the F-system provides fail-safe values (0) instead of the process data pending in the PII to the safety program.

The F-system recognizes an overflow or underflow of a channel of the **SM 336; AI 6 x 13Bit** or the **SM 336; F-AI 6 x 0/4 ... 20 mA HART** 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 in the case of F-I/O with inputs you want to process other fail-safe values besides "0" **for analog channels of data type INT (WORD)** in the safety program, you can specify individual fail-safe values when QBAD/QBAD_I_xx/QBAD_O_xx = 1.

⚠ WARNING
For F-I/O with inputs, the fail-safe value "0" provided in the PII must be further processed for digital channels of data type BOOL in the safety program.

When passivation occurs in a F-I/O module 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:

- Cause of passivation of the F-I/O/channels of the F-I/O
- Parameters you have to assign for the F-I/O DB (see below)

Note

Note that channel-level passivation is possible for the faulty channel in the event of a channel fault in the F-I/O. If configured accordingly in *HW Config*, the fail-safe value (0) is output for the affected channel. If you have configured channel-level passivation for the F-I/O, the relevant channels are reintegrated once the fault is corrected; any faulty channels remain passivated.

See also

Configuring the F-I/O (Page 36)

5.3 F-I/O DB

Introduction

An F-I/O DB is automatically created for each F-I/O during compilation in *HW Config*. This F-I/O DB contains variables that you can evaluate in the safety program, or that you can or must describe (except for the DIAG variable, which can only be evaluated in the standard user program). The initial values or actual values of the variables cannot be changed directly in the F-I/O DB because the F-I/O DB is know-how protected.

Use of Access to an F-I/O DB

You access variables 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 as a function of particular states of the safety program (for example, group passivation)
- For reassignment of parameters for fail-safe DP standard slaves/ standard I/O devices
- If you want to evaluate whether fail-safe values or process data should be output

Variables of an F-I/O DB

The following table presents the variables of an F-I/O DB:

	Variable	Data type	Function	Default
Variables that Can or Must be Described	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	Variable for parameter reassignment of fail-safe DP standard slaves/standard I/O devices or SM 336; F-AI 6 x 0/4 ... 20 mA HART for enabling HART communication	0
Variablesthat Can Be Evaluated:	PASS_OUT	BOOL	Passivation output*	1
	QBAD	BOOL	1=Fail-safe values are output*	1
	ACK_REQ	BOOL	1=acknowledgment requirement for reintegration	0
	IPAR_OK	BOOL	Variable for parameter reassignment of fail-safe DP standard slaves/standard I/O devices or 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	1
	QBAD_O_xx	BOOL	1=fail-safe values are output to output channel xx	1
* For a description, see information in "PASS_OUT/QBAD/QBAD_I_xx/QBAD_O_xx"				

PASS_ON

The PASS_ON variable allows you to enable passivation of an F-I/O, for example, as a function of particular states in your safety program.

Using the PASS_ON variable in the F-I/O DB, you can only passivate the entire F-I/O; channel-level passivation is not possible.

As long as PASS_ON equals 1, the associated F-I/O are **passivated**.

ACK_NEC

If an F-I/O fault is detected by the F-I/O, the relevant F-I/O are **passivated**. If channel faults are detected, the relevant channels are passivated if channel-level passivation is configured. 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, the relevant F-I/O are **reintegrated**, depending on ACK_NEC:

- With ACK_NEC = 0, you can program **automatic reintegration**.
- With ACK_NEC = 1, you can program **reintegration** through a **user acknowledgment**.

⚠WARNING

ACK_NEC = 0 can be assigned only if automatic reintegration is permissible for the relevant process from a safety standpoint.

Note

By default, ACK_NEC = 1 after creation of the F-I/O DB. If you do not require automatic reintegration, you do not need to describe ACK_NEC.

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 are passivated. If channel faults are detected, the relevant channels are passivated if channel-level passivation is configured. 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 the fault has been eliminated requires a **user acknowledgment** with a positive edge at variable ACK_REI of the F-I/O DB:

- After every communication error
- After F-I/O faults or channel faults when ACK_NEC = 1 is assigned

Reintegration after channel faults reintegrates all channels whose faults were eliminated.

Acknowledgment is only possible when ACK_REQ = 1.

In your safety program, you must provide for a user acknowledgment by means of ACK_REI for each F-I/O.

⚠WARNING

For the user acknowledgement, you must interconnect the ACK_REI variable of the F-I/O DB with a signal generated by an operator input. An interconnection with an automatically generated signal is not allowed.

Note

Alternatively, you can use the FB 219 "F_ACK_GL" F-application block to carry out reintegration of the F-I/O following communication errors or F-I/O/channel faults (see Chapter "FB 219 "F_ACK_GL: Global Acknowledgment of all F-I/O of an F-Runtime Group").

IPAR_EN

The IPAR_EN variable corresponds to the iPar_EN_C variable in the PROFIsafe bus profile, PROFIsafe Specification V1.20 and higher.

Fail-safe DP standard slaves/standard I/O devices

To find out when this variable has to be set or reset when parameters of fail-safe DP standard slaves are reassigned, consult the PROFIsafe specification V1.20 or higher or the documentation for the fail-safe DP standard slave/standard I/O device.

⚠WARNING

Note that IPAR_EN = 1 does **not** trigger passivation of the relevant F-I/O.

If passivation should continue to occur when IPAR_EN = 1, you must also set variable PASS_ON = 1.

HART communication with SM 336; F-AI 6 x 0/4 ... 20 mA HART

If you set variable IPAR_EN to "1" when parameter "HART_GATE" = "switchable", the HART communication is enabled for the SM 336; F-AI 6 x 0/4 ... 20 mA HART. Setting this variable to "0" disables the HART communication. The F-SM acknowledges the enabled or disabled HART communication with variable IPAR_OK = 1 or 0.

The HART communication is not enabled until the status of your system allows the parameters of the associated HART field device to be safely reassigned.

If you want to evaluate the "Enable HART communication" status in your safety program, e.g., for the purpose of programming interlocks, you must generate 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 IPAR_EN variable when evaluating the status if there is no passivation due to a communication error or F-I/O/channel fault (PASS_OUT = 0).

Example of enabling HART communication

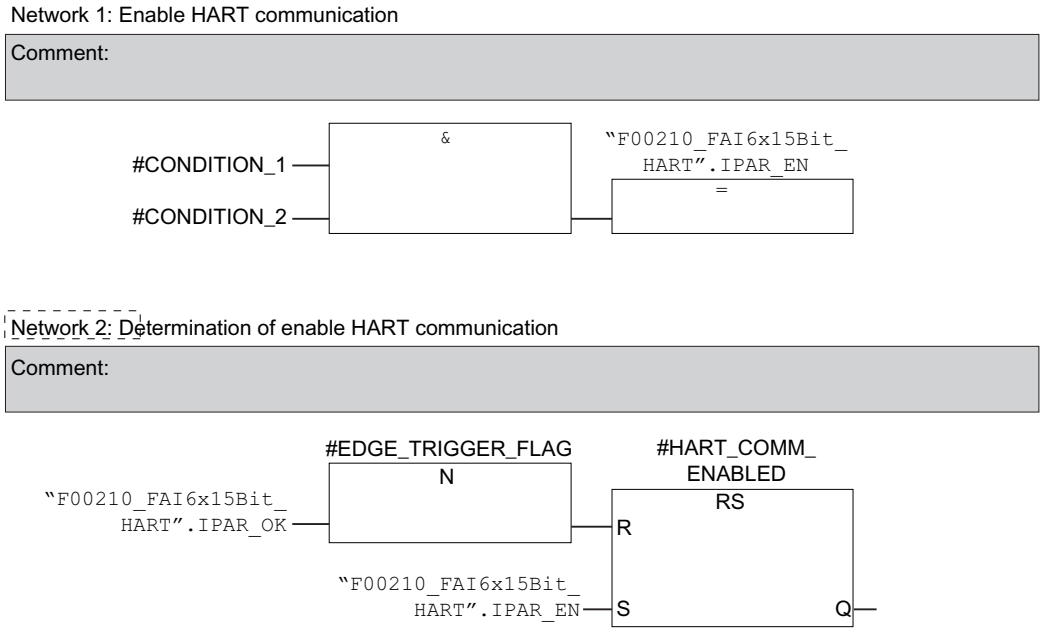


Figure 5-1 Example of enabling HART communication

You can find additional information on HART communication with SM 336; F-AI 6 x 0/4 ... 20 mA HART in the *S7-300, Fail-Safe Signal Modules* manual and in the object properties of this F-SM in the online help for *HW Konfig*.

PASS_OUT/QBAD/QBAD_I_xx/QBAD_O_xx

If you have configured channel-level passivation for the F-I/O, PASS_OUT = 1 and QBAD = 1 indicate that at least one channel was passivated. QBAD_I_xx and QBAD_O_xx indicate the input and output channels that were passivated.

If you have configured passivation of the entire F-I/O, the PASS_OUT = 1 and QBAD = 1 variables indicate that the entire F-I/O is passivated.

The **F-system** sets PASS_OUT, QBAD, QBQD_I_xx, and QBAD_O_xx = 1, as long as fail-safe 0 values are used instead of process data for the associated F-I/O or individual channels of the F-I/O.

However, if **you** enable passivation by setting PASS_ON = 1, only QBAD, QBAD_I_xx, and QBAD_O_xx = 1 is set. PASS_OUT does not change value in the event of passivation is enabled with PASS_ON = 1. For this reason, PASS_OUT can be used for group passivation of additional F-I/O.

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-level passivation, the F-system sets ACK_REQ = 1 as soon as the channel fault is corrected. User acknowledgement 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 faults or channel faults may only be possible minutes after the fault has been eliminated due to necessary test signal inputs (see *F-I/O manuals*).

IPAR_OK

The IPAR_OK variable corresponds to the iPar_OK_S variable in the PROFIsafe bus profile, PROFIsafe Specification V1.20 and higher.

Fail-safe DP standard slaves/standard I/O devices

To find out how to evaluate this variable when parameters of fail-safe DP standard slaves or standard I/O devices are reassigned, consult the PROFIsafe specification V1.20 or higher or the documentation for the fail-safe DP standard slave/standard I/O device.

HART communication with SM 336; F-AI 6 x 0/4 ... 20 mA HART

See Section "IPAR_EN"

DIAG

The DIAG variable 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 necessary, it can be evaluated in your standard user program. DIAG bits are saved until you perform an acknowledgment at ACK_REI or until automatic reintegration takes place.

Note

Access to this variable in the safety program is not permitted.

Structure of DIAG

Bit No.	Assignment	Possible Causes of Problems	Remedies
Bit 0	Timeout detected by F-I/O	The PROFIBUS/PROFINET connection between F-CPU and F-I/O is faulty. The monitoring time of the F-I/O in *HW Config* is set too low. The F-I/O is receiving invalid parameter assignment data. or	• Check the PROFIBUS/PROFINET connection and ensure that there are no external sources of interference. • Check the parameter assignment of the F-I/O in *HW Config*. 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. • Check the diagnostics buffer of the F-I/O. • Turn the power of the F-I/O off and back on.
		Internal F-I/O fault 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	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	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	Reserved	-	-

See also

Configuring the F-I/O (Page 36)
Passivation and Reintegration of F-I/O after F-I/O Faults and Channel Faults (Page 110)
Group passivation (Page 114)

5.4 Accessing F-I/O DB Variables

Symbolic Name of the F-I/O DB

During compilation in *HW Config*, an F-I/O DB is automatically created for each F-I/O, and a symbolic name is entered for the F-I/O DB in the symbol table.

The symbolic name is generated by combining the fixed prefix "F," the start address of the F-I/O, and the names (maximum 17 characters) entered in the object properties for the F-I/O in *HW Config* (example: F00005_4_8_F_DI_24VDC).

For F-I/O accessed via I-slave-slave communication, an X is added after the start address of the F-I/O (e.g. F00005_X_4_8_F_DI_DC24V).

Rule for Accessing Variables of F-I/O DB

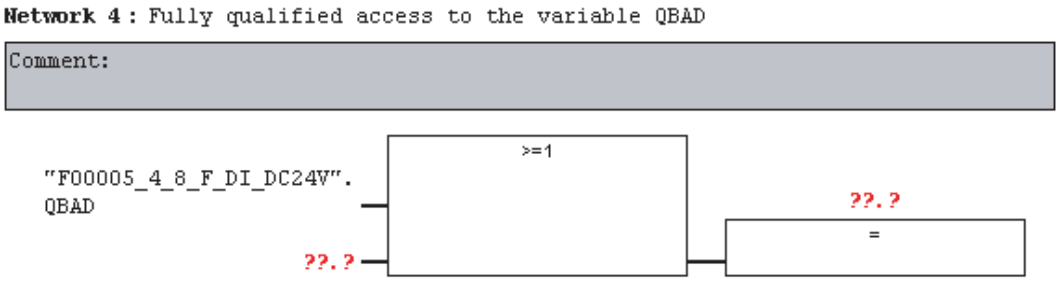
Variables of the F-I/O DB of an F-I/O can only be accessed from one F-runtime group and only from the F-runtime group from which the channels of this F-I/O are accessed (if access is made).

Fully Qualified DB Access

You can access the variables of the F-I/O DB with "fully qualified DB access" (that is, by specifying the symbolic name of the F-I/O DB and by specifying the name of the variable).

Make sure that "Report Cross References as Error" is not selected in the "General" dialog (**Options > Settings**) in the *FBD/LAD Editor*. Otherwise, the variables of the F-I/O DBs cannot be accessed.

Example of Evaluating the QBAD Variable



See also

Assigning Symbolic Names (Page 44)

5.5 Passivation and Reintegration of F-I/O after F-System Startup

Behavior after Startup

After a startup of the F-system, communication between the F-CPU and F-I/O must be established in accordance with the PROFIsafe safety protocol. During this time, the entire F-I/O are **passivated**.

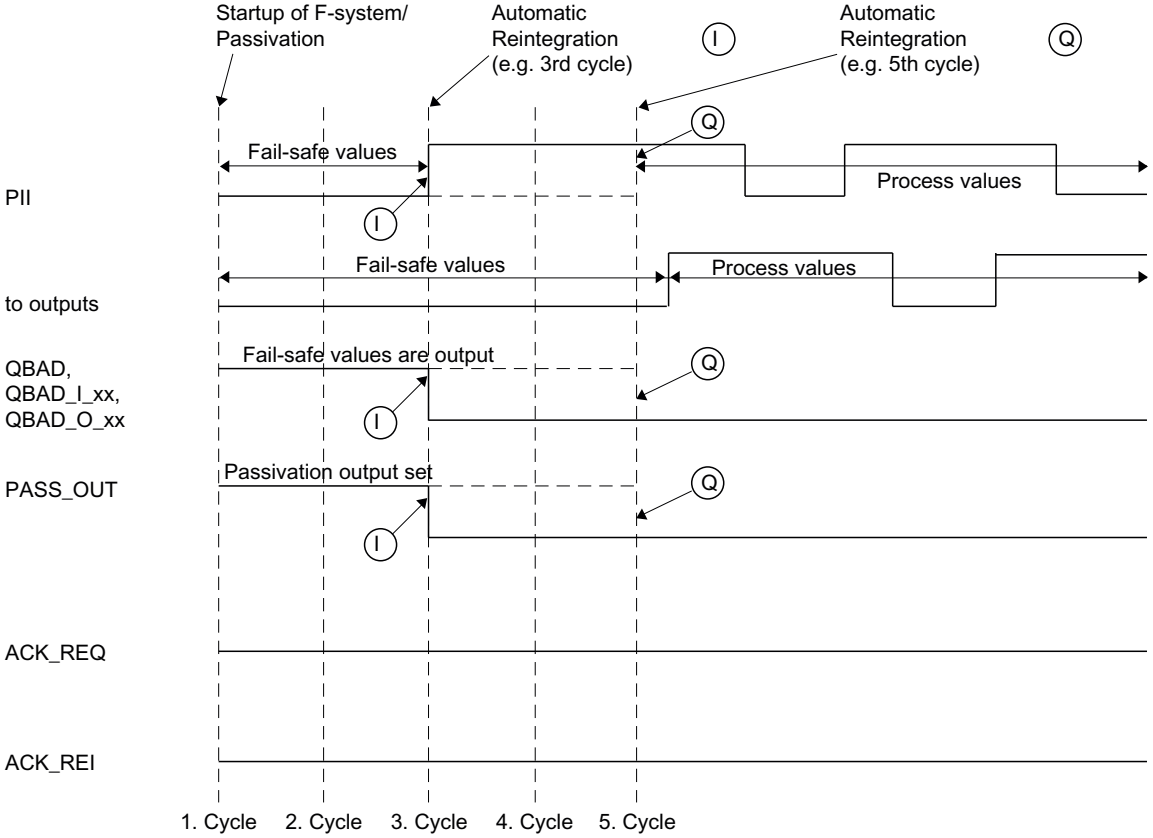
While fail-safe values (0) are being used, variables QBAD, PASS_OUT, QBAD_I_xx, and QBAD_O_xx = 1.

Reintegration of F-I/O

Reintegration of the F-I/O, that is, the provision of process data in the PII or the transfer of process data provided in the PIQ to the fail-safe outputs, takes place **automatically**, starting at the **earliest** with the second cycle of the F-run-time group after startup of the F-system; this happens regardless of the setting at variable ACK_NEC. Depending on the F-I/O you are using and the cycle time of the F-run-time group and PROFIBUS DP/PROFINET IO, several cycles of the F-run-time group can elapse before reintegration occurs.

If communication between the F-CPU and F-I/O takes longer to establish than the monitoring time set in the object properties for the F-I/O in *HW Config*, automatic reintegration does not take place.

Signal Chart 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

If you do not want automatic reintegration to take place after startup of the F-system, you must program startup protection.

See also

Programming Startup Protection (Page 94)

Passivation and Reintegration of F-I/O after Communication Errors (Page 108)

5.6 Passivation and Reintegration of F-I/O after Communication Errors

Behavior after Communication Errors

If the F-system detects an error during safety-related communication (communication error) between the F-CPU and an F-I/O in accordance with the PROFIsafe safety protocol, the relevant F-I/O are **passivated**.

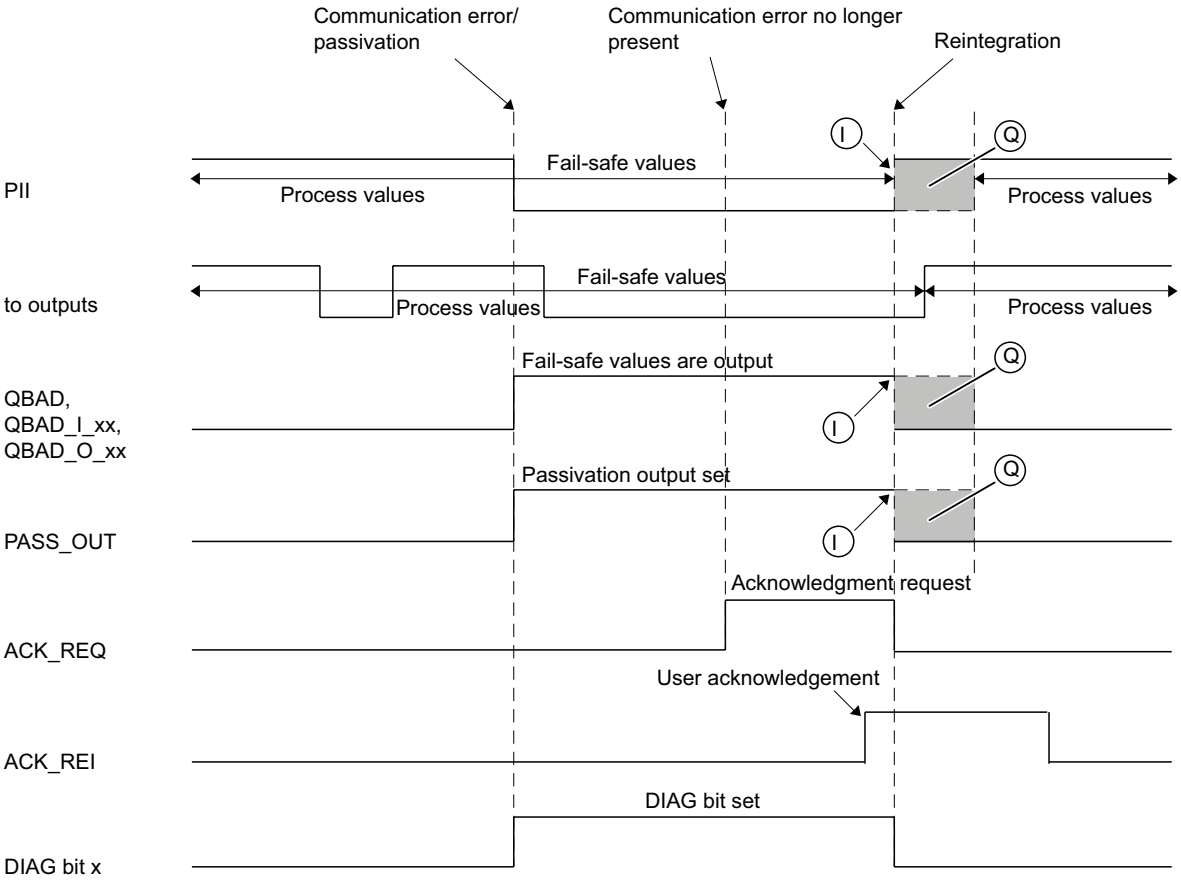
The QBAD, PASS_OUT, QBAD_I_xx, and QBAD_O_xx variables are set to "1" when fail-safe values (0) are being used.

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 ACK_REQ = 1
- A **user acknowledgment** with a positive edge has occurred:
  - On the ACK_REI variable of the F-I/O DB or
  - On the ACK_REI_GLOB input of the FB 219 "F_ACK_GL" F-application block (see Chapter *9.1.2.18*)

Signal Chart 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 pattern dependent on the F-I/O used)

See also

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Implementing User Acknowledgment in the Safety Program of a I-Slave F-CPU (Page 120)

5.7 Passivation and Reintegration of F-I/O after F-I/O Faults and Channel Faults

Behavior after F-I/O Faults

If the F-system detects an F-I/O fault (programming errors, excess temperature, for example) it passivates the corresponding F-I/O.

The QBAD, PASS_OUT, QBAD_I_xx, and QBAD_O_xx variables are set to "1" when fail-safe values (0) are being used.

Behavior after Channel Faults

If the F-system detects a channel fault (e.g., short circuit, overload, discrepancy error, or wire break), the response of the F-system depends on how the "Behavior after Channel Faults" parameter for the F-I/O is configured in HW Config.

If you have configured channel-specific passivation, the relevant channels of the F-I/O are passivated. While fail-safe values (0) are being used, variables QBAD, PASS_OUT or QBAD_I_xx and QBAD_O_xx of the relevant channels = 1.

If you have configured passivation of the entire F-I/O, passivation occurs just like after F-I/O errors (see above).

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 faults or channel faults have been eliminated.

If you have configured channel-specific passivation for the F-I/O, the relevant channels are reintegrated once the fault is corrected; any faulty channels remain passivated.

Reintegration takes place as follows, depending on your setting for ACK_NEC:

- When ACK_NEC = 0, **automatic reintegration** takes place as soon as the F-system detects that the fault has been eliminated. 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 place several minutes after completion of necessary test signal inputs, which are used by the F-I/O to determine that the fault has been eliminated.
- With ACK_NEC = 1, reintegration takes place only as a result of a user acknowledgement with a positive edge on the ACK_REI variable of the F-I/O DB or on the ACK_REI_GLOB input of the FB 219 "F_ACK_GL" F-application block. Acknowledgment can be made as soon as the F-system detects that the fault has been eliminated and it has set ACK_REQ = 1.

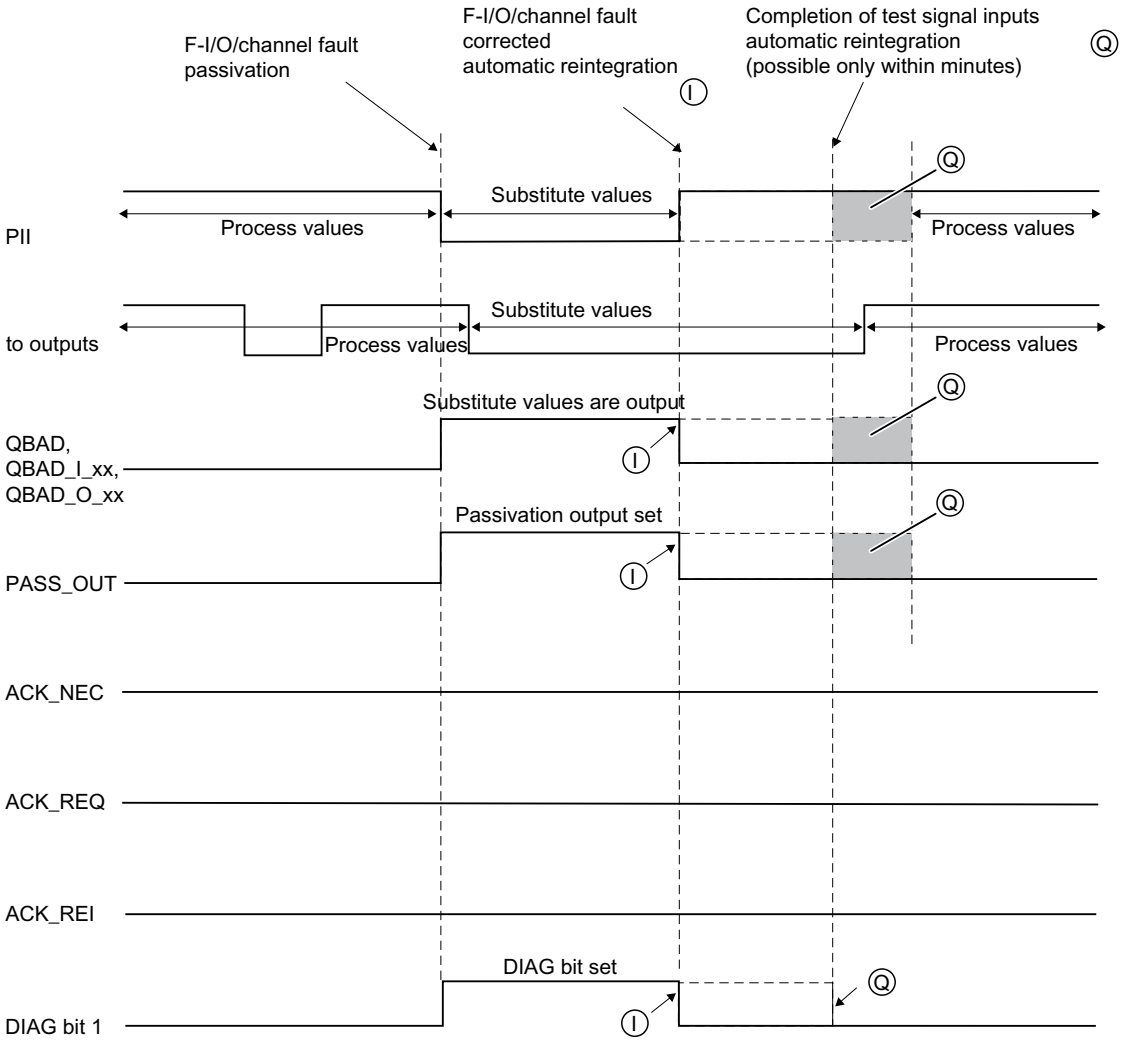
⚠WARNING

Following a power failure of the F-I/O lasting shorter than the specified monitoring time for the F-I/O in *HW Config* (see *Safety Engineering in SIMATIC S7* system manual), automatic reintegration can occur regardless of your setting for ACK_NEC, as described for the case when ACK = 0.

If for this case, automatic reintegration is not permissible for the relevant process, you must program startup protection by evaluating variables QBAD or QBAD_I_xx and QBAD_O_xx or PASS_OUT.

In the event of a power failure of the F-I/O lasting longer than the specified monitoring time for the F-I/O in HW Config, the F-system detects a communication error.

Signal Sequence for Passivation and Reintegration of F-I/O after F-I/O Faults and Channel Faults When ACK_NEC = 0 (for Passivation of Entire F-I/O after Channel Faults)



(I) For F-I/O with inputs

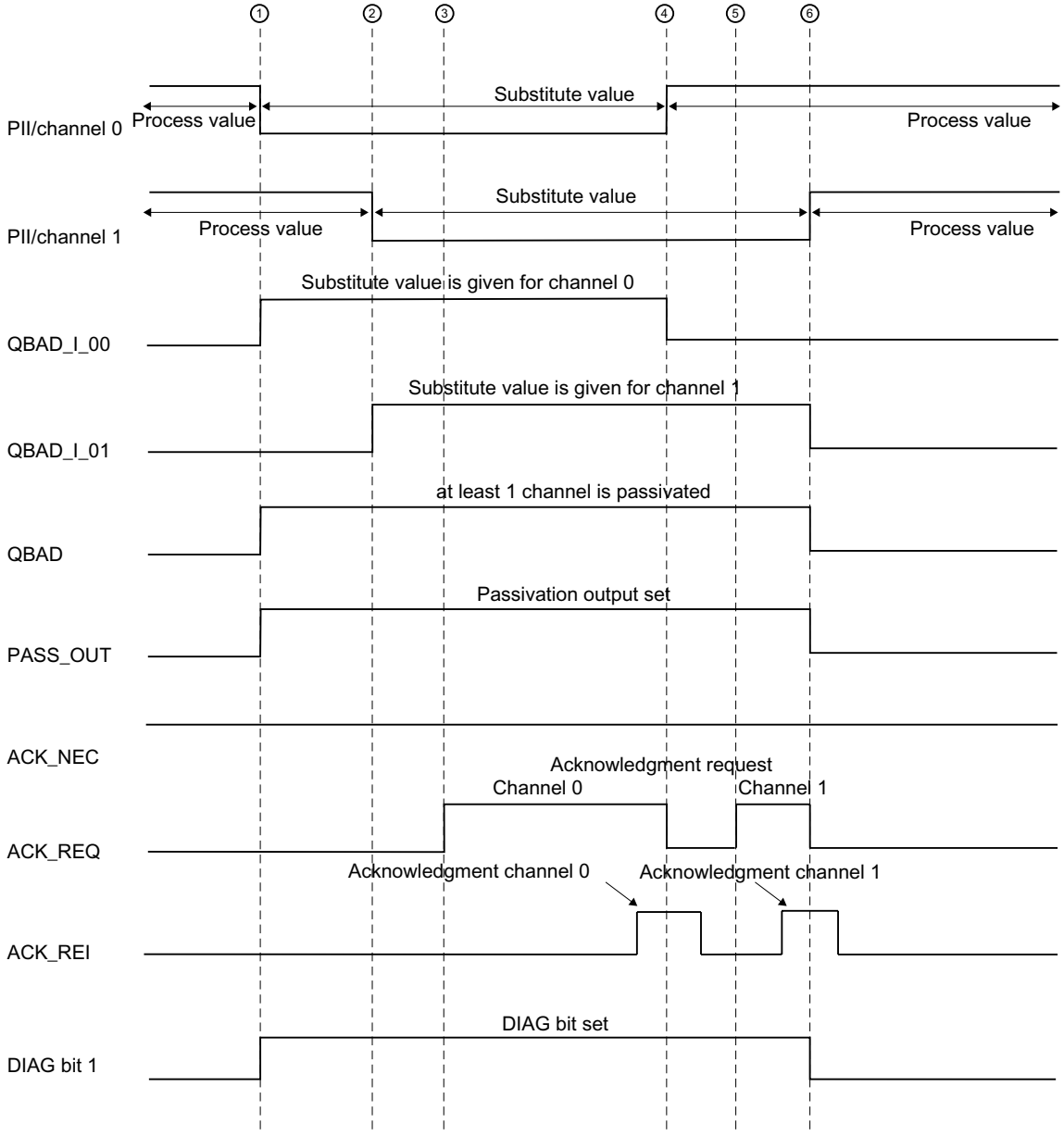
(Q) For F-I/O with outputs and F-I/O with inputs and outputs (signal pattern dependent on the F-I/O in use)

Signal Sequence for Passivation and Reintegration of F-I/O after F-I/O Faults and Channel Faults when ACK_NEC = 1 (for Passivation of Entire F-I/O after Channel Faults)

For the signal sequence for passivation and reintegration of the F-I/O after F-I/O faults or channel faults when ACK_NEC = 1 (default), see Chapter "Passivation and Reintegration of the F-I/O after Communication Errors".

Signal Chart for Passivation and Reintegration of F-I/O after Channel Faults when ACK_NEC = 1 (for channel-specific passivation)

Example of an F-I/O with inputs:



① Channel fault for channel 0/passivation channel 0
② Channel fault for channel 1/passivation channel 1
③ Channel fault for channel 0 was corrected
④ Reintegration channel 0
⑤ Channel fault for channel 1 was corrected
⑥ Reintegration channel 1

See also

Configuring the F-I/O (Page 36)

Programming Startup Protection (Page 94)

Passivation and Reintegration of F-I/O after Communication Errors (Page 108)

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Implementing User Acknowledgment in the Safety Program of a I-Slave F-CPU (Page 120)

5.8 Group passivation

Programming a Group Passivation

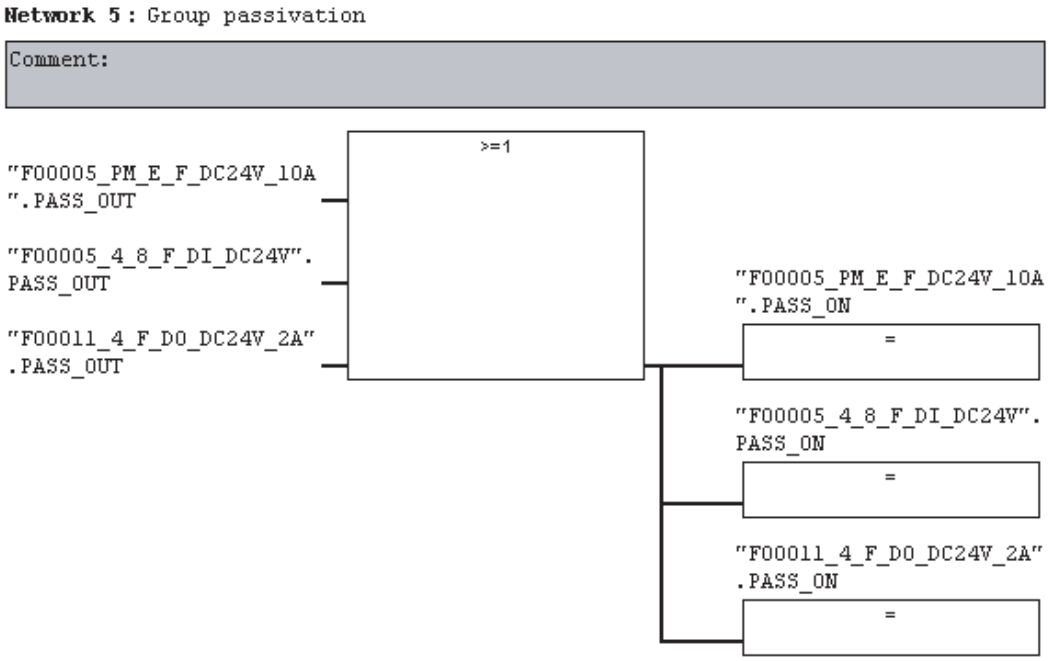
If you want to enable passivation of additional F-I/O when an F-I/O or a channel of an F-I/O is passivated by the F-system, you can use the PASS_OUT/PASS_ON variables to perform a **group passivation** of associated F-I/O.

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 variables of the F-I/O in the group and assign the result to all PASS_ON variables of the F-I/O in the group.

While fail-safe values (0) are being applied due to group passivation using PASS_ON = 1, the QBAD, QBAD_I_xx, and QBAD_O_xx variables of the F-I/O in the group are set to 1.

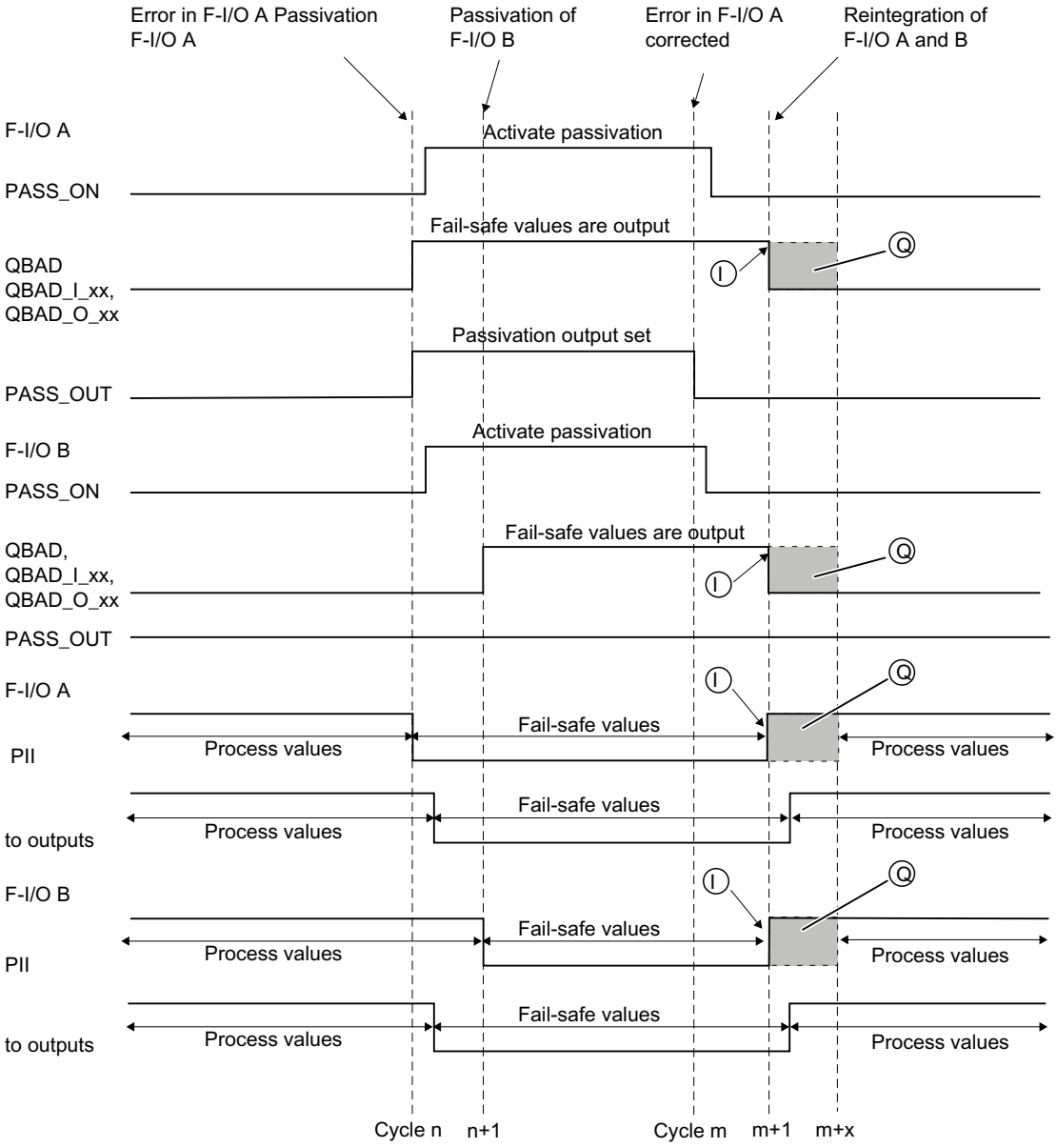
Example of Group Passivation



Reintegration of F-I/O

Reintegration of F-I/O passivated by group passivation takes place **automatically**, if reintegration of the F-I/O that triggered the group passivation takes place (either **automatically** or **through user acknowledgment**) (PASS_OUT = 0).

Signal Chart for Group Passivation



Ⓘ for F-I/O with inputs

Ⓠ for F-I/O with outputs and F-I/O with inputs and outputs
(signal pattern dependent on the F-I/O used)

Implementation of user acknowledgment

6

6.1 Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller

Options for User Acknowledgment

You can implement a user acknowledgment in one of the following ways:

- By means of an acknowledgment key that you connect to an F-I/O with inputs
- By means of an operator control and monitoring system

User Acknowledgment by Means of Acknowledgment Key

Note

If you use the option of user acknowledgment by means of an acknowledgment key, and a communication error, an F-I/O fault, or a channel fault occurs at 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 remedied 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 operator control and monitoring system for the acknowledgment for reintegration of an F-I/O to which an acknowledgment key is connected.

User Acknowledgment by Means of an Operator Control and Monitoring System

User acknowledgment by means of an operator control and monitoring system requires the F_ACK_OP F-application block from the *Distributed Safety* F-library (V1).

Procedure for Programming User Acknowledgment by Means of an Operator Control and Monitoring System

1. Call the "F_ACK_OP" F-application block in your safety program. The acknowledgment signal for evaluating user acknowledgments is provided at output OUT of F_ACK_OP.
2. On your operator control and monitoring system, set up a field for manual entry of an "acknowledgment value" of "6" (first step in acknowledgment) and an "acknowledgment value" of "9" (second step in acknowledgment) in the instance DB of F_ACK_OP (input IN).

   or

   Assign function key 1 to transfer an "acknowledgment value" of "6" (first step in acknowledgment) and function key 2 to transfer an "acknowledgment value" of "9" (second step in acknowledgment) in the instance DB of F_ACK_OP (input IN).
3. Optional: On your operator control and monitoring system, evaluate input Q in the instance DB of F_ACK_OP to indicate the time frame within which the second step in acknowledgment must occur or to indicate that the first step in acknowledgment has already occurred.

If you should only be able to perform a user acknowledgment from one programming device or PC using the "Monitor/Modify Variable" function, and you do not want to deactivate safety mode, then you must transfer an address (memory word) at input IN when calling the F_ACK_OP F-block. You can then transfer "acknowledgment values" "6" and "9" on the programming device or PC by modifying the memory word. The memory word must not be described by the program.

Note

If you interconnect input IN to a memory word, it may only be an input at F_ACK_OP in one F-runtime group.

⚠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 one program and using one function key to trigger them.
The two separate acknowledgment steps will also prevent your non-fail-safe operator control and monitoring system from triggering an erroneous acknowledgment.

⚠ WARNING

If your operator control and monitoring system can access multiple F-CPUs that use F_ACK_OP for fail-safe acknowledgment, or if you have networked operator control and monitoring systems and F-CPUs (with F_ACK_OP F-application blocks), you must be sure that the correct F-CPU is in fact being addressed **before** executing the two acknowledgment steps:

- In each F-CPU, store a network-wide unique name for the F-CPU in a DB of your standard user program.
- In your operator control and monitoring 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 operator control and monitoring 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.

Example of Procedure for Programming a User Acknowledgment for Reintegrating an F-I/O

1. Optional: Set the ACK_NEC variable in the respective F-I/O DB to "0" if automatic reintegration (without user acknowledgment) is to take place after an F-I/O fault or a channel fault.

   ⚠ WARNING

   ACK_NEC = 0 can only be assigned if automatic reintegration is permissible for the relevant process from a safety standpoint.

2. Optional: Evaluate the QBAD or QBAD_I_xx and QBAD_O_xx or DIAG variables in the respective F-I/O DB to trigger an indicator light, if applicable, in the event of an error, and/or generate error messages on your operator control and monitoring system in your standard user program by evaluating QBAD or QBAD_I_xx and QBAD_O_xx or DIAG; 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 variable in the respective F-I/O DB, for example, in the standard user program or on the operator control and monitoring system, to query or to indicate whether user acknowledgment is required.
4. Assign the input of the acknowledgment key or the OUT output of F_ACK_OP to the ACK_REI variable in the respective F-I/O DB or the ACK_REI_GLOB input of the FB 219 "F_ACK_GL" F-application block (see above).

See also

F-I/O DB (Page 98)

FB 187 "F_ACK_OP": Fail-Safe Acknowledgment (Page 189)

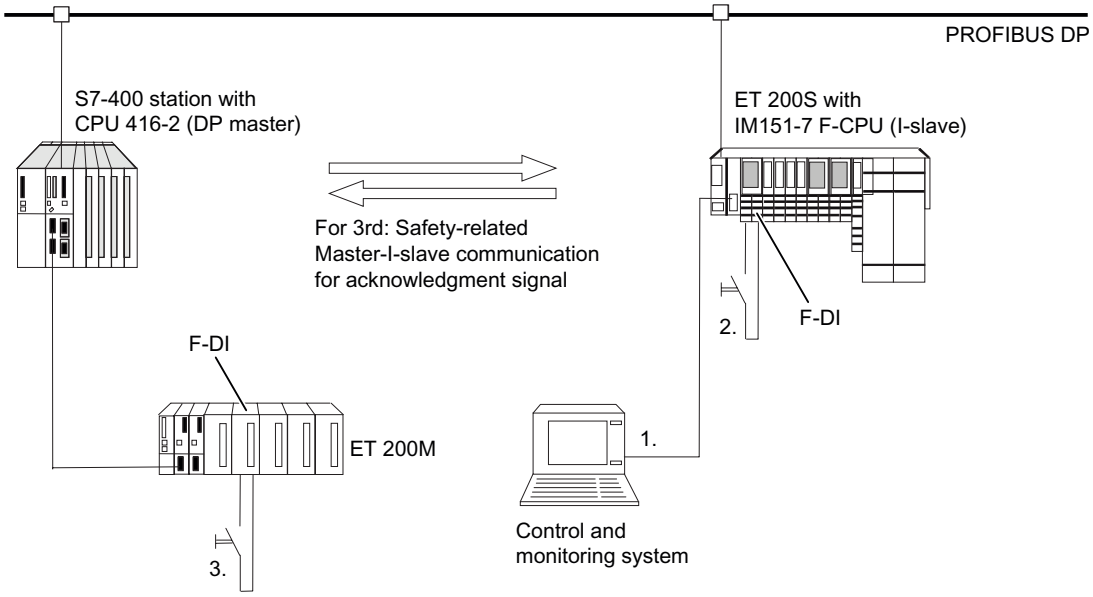
6.2 Implementing User Acknowledgment in the Safety Program of a I-Slave F-CPU

Options for User Acknowledgment

You can implement a user acknowledgment in one of the following ways:

- By means of an operator control and monitoring system that you can use to access the F-CPU of the I-slave
- By means of an acknowledgment key that you connect to an F-I/O with inputs that is assigned to the F-CPU of the I-slave
- By means of an acknowledgment key that you connect to an F-I/O with inputs that is assigned to the F-CPU of the DP master

These three options are illustrated in the figure below.



1. User Acknowledgment by Means of an Operator Control and Monitoring System that You Can Use to Access the F-CPU of the I-Slave

To implement a user acknowledgment by means of an operator control and monitoring system that you can use to access the F-CPU of the I-slave, you need the F_ACK_OP F-application block from the *Distributed Safety* F-library (V1).

Programming Procedure

Follow the procedure described in Chapter "Procedure for Programming User Acknowledgment by Means of an Operator Control and Monitoring System" under "Implementing User Acknowledgment in Safety Program of F-CPU of DP Master".

From your operator control and monitoring system, you can then access the instance DB of F_ACK_OP in the I-slave directly.

2. User Acknowledgment by Means of an Acknowledgment Key at an F-I/O with Inputs Assigned to the F-CPU of the I-Slave

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 "block" can only be removed by a STOP-to-RUN transition of the F-CPU of the I-slave.

Consequently, it is recommended that you also provide for an acknowledgment by means of an operator control and monitoring system that you can use to access the F-CPU of the I-slave for the acknowledgment for 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 Assigned to the F-CPU of the DP Master

If you want to use the acknowledgment key that is assigned to the F-CPU on the DP master for a user acknowledgment in the safety program of the F-CPU of an I-slave, you must transmit the acknowledgment signal from the safety program in the F-CPU of the DP master to the safety program in the F-CPU of the I-slave by means of safety-related master-I-slave communication.

Programming Procedure

1. Call the F_SENDDP F-application block in the safety program in the F-CPU of the DP master.
2. Call the F_RCVDP F-application block in the safety program in the F-CPU of the I-slave.
3. Supply an input SD_BO_xx of the F_SENDDP block with the input of the acknowledgment key.
4. The acknowledgment signal for evaluating user acknowledgments is now available at the corresponding output RD_BO_xx of the F_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, "Name F_RCVDP1".RD_BO_02). To enable this, you must first assign a symbolic name ("Name F_RCVDP1" in the example) for the instance DB of F_RCVDP in the symbol table.
5. Supply the corresponding input SUBBO_xx of the F_RCVDP with the fail-safe value "RLO0," so that an unintentional user acknowledgment is not triggered before communication is established the first time after startup of the sending and receiving F-system, or in the event of a safety-related communication error. RLO 0 is available in the F-shared-DB. At input SUBBO_xx, enter "F_GLOBDB".RLO0 fully qualified.

Note

If a communication error, an F-I/O fault, 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 "block" can only be removed by a STOP-to-RUN transition of the F-CPU of the DP master.

Consequently, it is recommended that you also provide for an acknowledgment by means of an operator control and monitoring system that you can use to access the F-CPU of the DP master for the acknowledgment for reintegration of the F-I/O to which an acknowledgment key is connected.

If a safety-related master-I-slave communication error occurs, the acknowledgment signal cannot be transmitted, and an acknowledgment for reintegration of safety-related communication is no longer possible.

This "block" can only be removed by a STOP-to-RUN transition of the F-CPU of the I-slave.

Consequently, it is recommended that you also provide for an acknowledgment by means of an operator control and monitoring system that you can use to access the F-CPU of the I-slave for the acknowledgment for reintegration of the safety-related communication for transmission of the acknowledgment signal (see 1).

See also

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Overview of safety-related communication (Page 127)

FB 187 "F_ACK_OP": Fail-Safe Acknowledgment (Page 189)

FB 223 "F_SENDDP" and FB 224 "F_RCVDP": Send and Receive Data via PROFIBUS DP (Page 229)

Data Exchange between Standard User Programs and Safety Program

7

7.1 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 out all data of the safety program, for example, through symbolic (fully qualified) accesses to the following:

- Instance DBs of the F-FBs
- F-DBs (for example, "Name F_DB".Signal_1)
- Process input image and process output image of F-I/O (for example, "Emergency_Stop_Button_1" (I 5.0))

Note

The process input image for F-I/O is updated not only at the start of an F runtime group prior to execution of the F-program block, but also by the standard operating system.

To find out the standard operating system update times, refer to "Process image of inputs/outputs" in the *STEP 7 Online Help*. With F-CPUs that support partial process images, also bear in mind the update times when using partial process images. For this reason, when accessing the process input image for F-I/O in the standard user program, you can 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 may access the process input image in the standard program only after execution of an F-runtime group. In this case, you can also evaluate the QBAD or QBAD_I_xx variable in the associated F-I/O DB in the standard user program to find out whether the process input image is receiving fail-safe values (0) or process data. When using partial process images, make sure as well that the process image is not updated by the standard operating system or by SFC 26 UPDAT_PI between execution of an F-runtime group (F-CALL) and evaluation of the process input image in the standard user program.

F-Shared DB

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

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

You use fully qualified access to access these variables (e.g., "F_GLOBDB".MODE). The number and symbolic name of the F-shared DB and the absolute addresses of variables are indicated in the printout of the safety program.

Bit Memory

You can also write to memory bits in the safety program to enable intermediate results of the safety program to be used by the standard user program without having to pass through F-data blocks. However, these memory bits must not be read in the safety program itself.

Process Output Image

The process output image (PIQ) of standard I/O can also be written to in the safety program, e.g., for display purposes. These values must not be read in the safety program, either (see table of supported address areas in Chapter "Differences between the F-FBD/F-LAD Programming Languages and the Standard FBD/LAD Languages").

See also

Differences between the F-FBD and F-LAD programming languages and the standard FBD and LAD programming languages (Page 61)

7.2 Data Transfer from the Standard User Program to the Safety Program

Data Transfer from Standard User Program to Safety Program

As a basic principle, only fail-safe data or fail-safe signals from fail-safe I/O and other safety programs (in other F-CPUs) can be processed in the safety program, since standard data and signals are not safe.

If you nevertheless have to process data from the standard user program in the safety program, you can evaluate either memory bits from the standard user program or the process input image (PII) for standard I/O in the safety program (see table of supported address areas in Chapter "Differences between the F-FBD/F-LAD Programming Languages and the Standard FBD/LAD Languages").

⚠ WARNING

Because these data 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 a memory bit or input of standard I/O is used in both F-runtime groups, you must perform the validity check separately in each F-runtime group.

To facilitate the checks, all signals from the standard user program that are evaluated in the safety program are included when the safety program is printed out.

Note

Data from the standard user program (bit memory or PII of standard I/O) cannot be used for edge memory bits of the RLO Edge Detection (N, P) or Address Edge Detection (NEG, POS) instructions or for the address of the Flip Flop (SR, RS) instructions, since these data are read and written to by the instruction.

Note

When F-blocks are being edited in F-FBD/F-LAD in the *FBD/LAD Editor*, all addresses that are **not** fail-safe are shown by default with a yellow background.

Example: Programming Validity Checks

- Use comparison instructions to check whether unsafe data from the standard user program exceed or fall below permitted upper and lower limits. You can then influence your safety function with the result of the comparison.
- With unsafe signals from the standard user program, for example, only allow a motor to be switched off, but not to be switched on using Set, Reset, or Flip-flop instructions.
- For starting cycles, gate unsafe signals from the standard user program, for example, using AND-gating with starting conditions that you derive from fail-safe signals.

If you want to process unsafe data in the safety program, bear in mind that a sufficiently simple method of checking validity does not exist for all unsafe data.

Reading Data from the Standard User Program When Changes to the Data are Possible during Runtime of an F-Runtime Group

You must use dedicated memory bits if you want to read data from the standard user program (bit memory or PII of standard I/O) in the safety program and these data can be changed by the standard user program or an operator control and monitoring system during runtime of the F-runtime group in which the data are read - for example, because your standard user program is being executed by a higher priority cyclic interrupt. You must write the data from the standard user program to these memory bits immediately before calling the F-runtime group. You can then only access these memory bits in the safety program.

Note, too, that **clock memory** that you defined when configuring your F-CPU (in *HW Config,* in the object properties for the F-CPU) 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 the information above is not observed. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety Program: internal CPU fault; internal error information: 404"

See also

Differences between the F-FBD and F-LAD programming languages and the standard FBD and LAD programming languages (Page 61)

Compiling Safety Program (Page 258)

Configuring and Programming Communication

8

8.1 Overview of safety-related communication

Introduction

This section provides an overview of the following options for safety-related communication in *S7 Distributed Safety* F-systems:

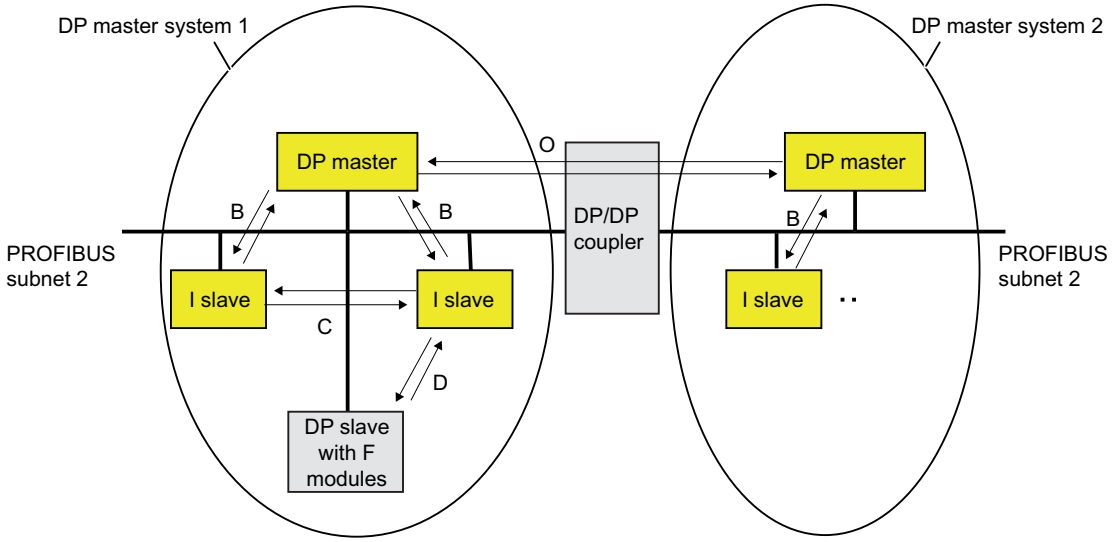
- Safety-related I-slave-slave communication (via PROFIBUS DP)

Safety-related CPU-CPU communication:

- Safety-related master-master communication (via PROFIBUS DP)
- Safety-related master-I-slave communication (via PROFIBUS DP)
- Safety-related I-slave-I-slave communication (via PROFIBUS DP)
- Safety-related IO controller-IO controller communication (via PROFINET IO)
- Safety-related communication by means of S7 connections (via Industrial Ethernet)
- Safety-related communication between *S7 Distributed Safety* and *S7 F Systems*

Overview of Safety-Related Communication via PROFIBUS DP

The figure below presents an overview of the four options for safety-related communication via PROFIBUS DP in *S7 Distributed Safety* F-systems.



A Safety-related master-master communication (via DP/DP coupler)
B Safety-related master-I-slave communication
C Safety-related I-slave-I-slave communication
D Safety-related I-slave-slave communication

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 data types BOOL and INT is transmitted in a fail-safe manner between the safety programs in F-CPUs of DP masters/I-slaves or IO controllers.

The data transmission makes use of F-application blocks F_SENDDP for sending and F_RCVDP for receiving. The data are stored in configured address areas of the DP/DP coupler/DP master/I-slave or PN/PN coupler.

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, e.g., with all ET 200S F-modules and with all S7-300 fail-safe signal modules with IM 153-2, order no. 6ES7 153-2BA01-0XB0 or higher, firmware version > V4.0.0.

Safety-related communication between the safety program of the F-CPU of an I-slave and F-I/O of a slave takes place using direct data exchange – same as in standard programs. The process image (PII and PIQ) is used to access the channels of the F-I/O in the safety program of the F-CPU of the I-slave.

Use of IE/PB Link

You can use the IE/PB Link to link the four options for safety-related communication via PROFIBUS DP in *S7 Distributed Safety* F-systems to PROFINET IO, as well (see also the documentation on PROFINET IO and IE/PB Link).

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 *Excel File for Response Time Calculation s7cotib.xls* for *S7 Distributed Safety*).

Note that this Excel file does not support all of the conceivable configurations.

Safety-Related CPU-CPU Communication via Industrial Ethernet

Safety-related CPU-CPU communication via Industrial Ethernet is possible by means of configured S7 connections. Communication from and to the following CPUs is possible:

- CPU 315F-2 PN/DP (only via the CPU PN interface)
- CPU 317F-2 PN/DP (only via the CPU PN interface)
- CPU 319F-3 PN/DP (only via the CPU PN interface)
- CPU 416F-2, **firmware version V4.0 and higher**
- CPU 416F- 3 PN/DP

In safety-related communication via S7 connections, a specified amount of fail-safe data of data types BOOL, INT, WORD, or TIME is transferred in a fail-safe manner between the safety programs of the F-CPUs linked by means of the S7 connection.

The data transfer makes use of the F-application blocks F_SENDS7 for sending and F_RCVS7 for receiving. Data are exchanged using one F-DB ("F-communication DB") each on the sender and receiver sides.

In addition, safety-related communication between *S7 Distributed Safety* and *S7 F Systems* is possible.

8.2 Safety-Related Master-Master Communication

8.2.1 Configuring Address Areas (Safety-Related Master-Master Communication)

DP/DP coupler

Safety-related communication between safety programs of the F-CPUs of DP masters takes place via a DP/DP coupler (Order No. 6ES7158-0AD01-0XA0).

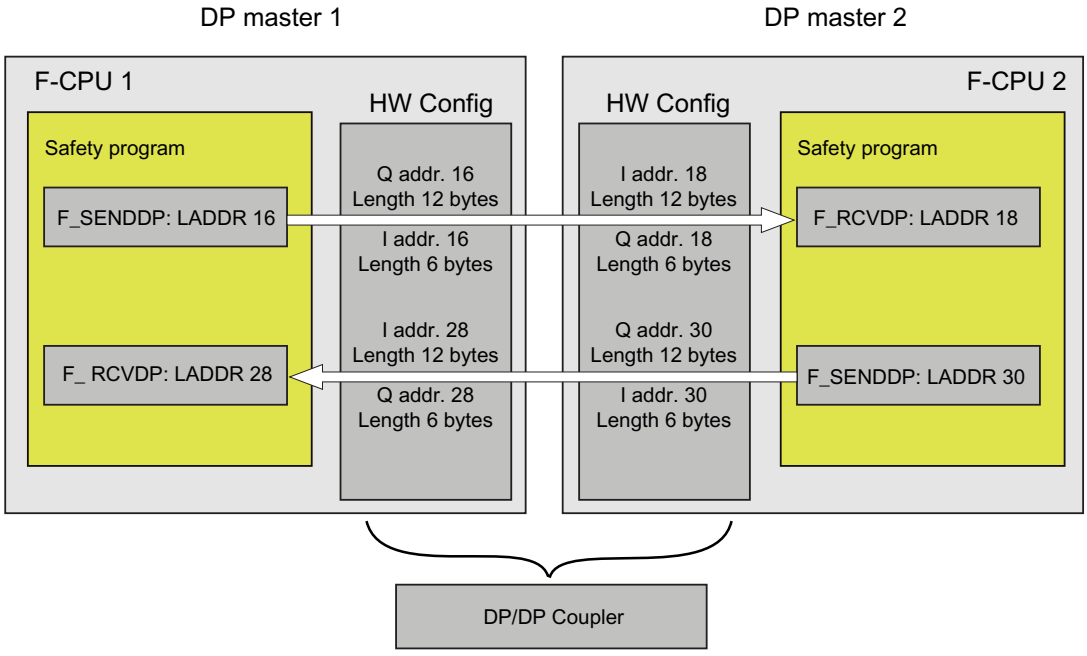
Each F-CPU is linked to the DP/DP coupler by means of its PROFIBUS DP interface.

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 Address Areas

You must configure one address area for output data and another address area for input data in the DP/DP coupler in *HW Config* for each connection between two F-CPUs via DP/DP coupler. In the figure below, each of the two F-CPUs will be able to send and receive data (bidirectional communication).



Rules for Defining the Address Areas

The output data address area for **data to be sent** must begin with the same start address as the associated input data address area. A total of 12 bytes (consistent) is required for the output data address area, while 6 bytes (consistent) are required for the input data address area.

The input data address area for **data to be received** must begin with the same start address as the associated output data address area. A total of 12 bytes (consistent) is required for the input data address area, while 6 bytes (consistent) are required for the output data address area.

8.2.2 Configuring Safety-Related Master-Master Communication

Requirements

You have created two stations with one DP master system each in *HW Config*.

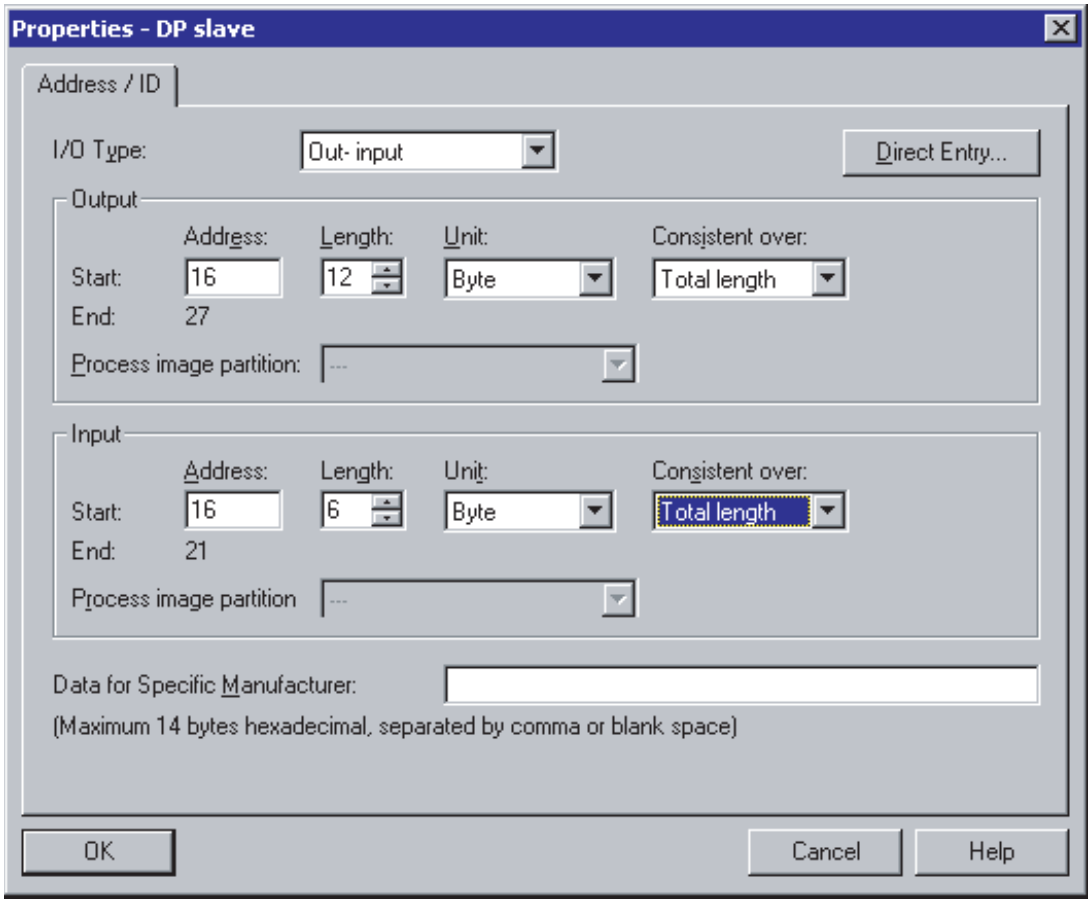
Procedure for Configuring Master-Master Communication (example with bidirectional communication)

1. Open the station with F-CPU 1.
2. Select the DP/DP coupler from the hardware catalog "PROFIBUS DP\Additional field devices\Gateway\DP/DP coupler". Place the DP/DP coupler on the DP master system of your F-CPU.
3. An available PROFIBUS address is automatically assigned in the shortcut menu. You can change this in address area 1 to 125. This address must be set via a switch on the DP/DP coupler: either directly on the DP/DP coupler by means of the DIP switch or using *STEP 7* (see DP/DP Coupler manual). You can insert the name of the subnet, the subnet ID, the author, and a comment using the "Properties" menu command. In the "Network Settings" tab, you should set the transmission rate to at least "1.5 Mbps". You must select "DP" as the profile.
4. In order for safety-related communication between CPUs to be able to be established **consistently** and for any address and length settings to be possible, you must use **universal modules**. Select "DP/DP" on the DP master system, and insert a universal module from the DP/DP Coupler folder.

   Use two universal modules for each F-CPU for bidirectional connections, that is, each F-CPU will send and receive data.

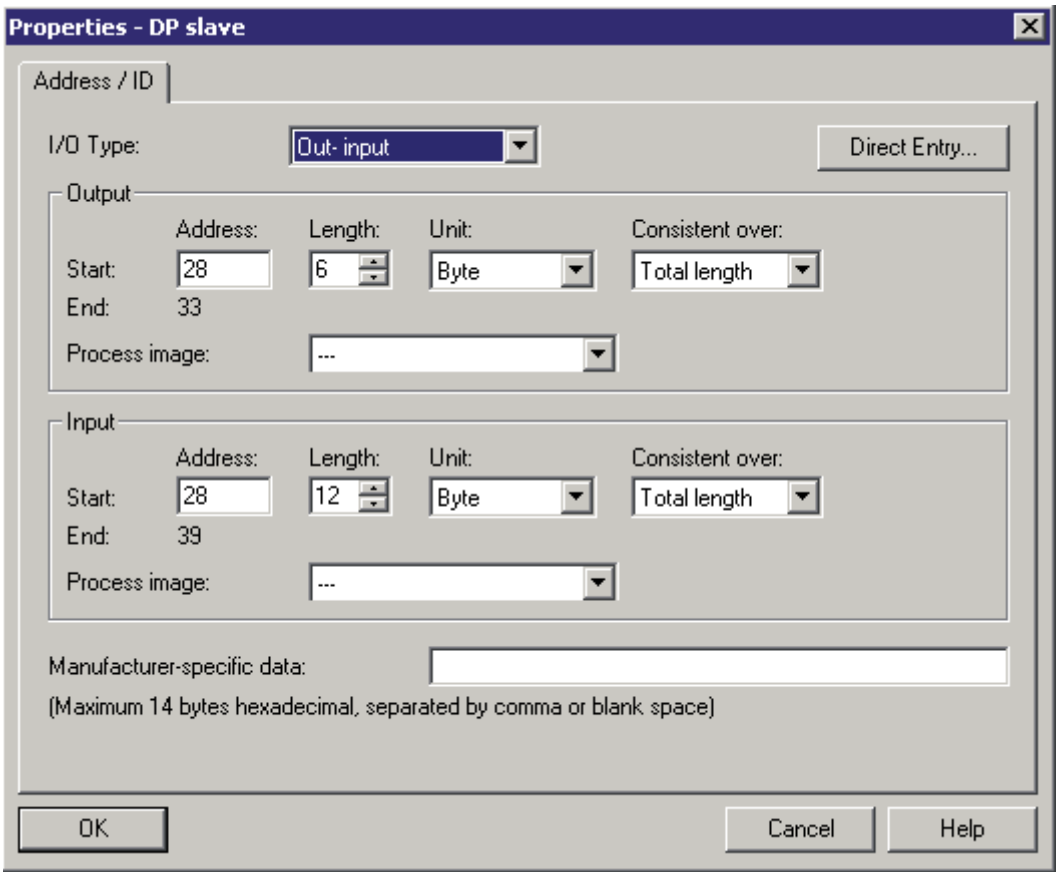
5. Select the first universal module, and select the **Edit > Object Properties** menu command.

   The object properties dialog appears.



6. In the object properties for the first universal module, select "Out input" as the I/O type.
7. Enter the associated values for the output data address area. In our example, enter "16" as "Start Address", "12" as "Length", "Byte" as "Unit", and "Total Length" as "Consistent".
8. Enter the associated values for the input data address area. In our example, enter "16" as "Start Address", "6" as "Length", "Byte" as "Unit", and "Total Length" as "Consistent".
9. Click "OK" to confirm.
10. Select the second universal module, and select the **Edit > Object Properties** menu command.

The object properties dialog appears.



11. In the object properties for the second universal module, select "Out input" as the I/O type.
12. Enter the associated values for the output data address area. In our example, enter "28" as "Start Address", "6" as "Length", "Byte" as "Unit", and "Total Length" as "Consistent over".
13. Enter the associated values for the input data address area. In our example, enter "28" as "Start Address", "12" as "Length", "Byte" as "Unit", and "Total Length" as "Consistent over".
14. Click "OK" to confirm. This completes the configuration of the master-master communication for F-CPU 1.

    Perform steps 1 to 14 for F-CPU 2. Note that you have to adjust the addresses accordingly (see figure in Chapter "Configuring the Address Areas (Safety-Related Master-Master Communication)".

Note

Make sure that the values you assign for the start addresses of the output and input data address areas are identical.

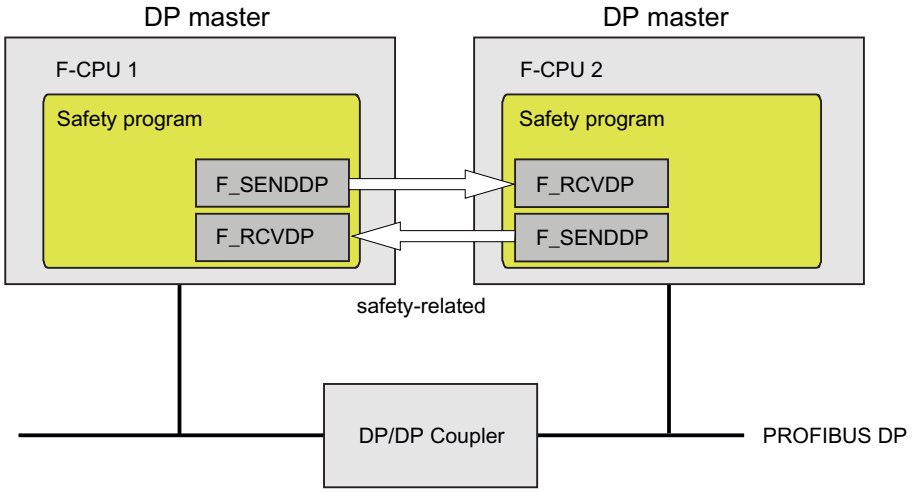
Always select the "Consistent over total length" option for all input and output data address areas.

Additional Information

The DP/DP coupler is described in the DP/DP Coupler manual.

8.2.3 Communication by Means of F_SENDDP and F_RCVDP (Safety-Related Master-Master Communication)

Communication by Means of F_SENDDP and F_RCVDP



Safety-related communication makes use of the F-application blocks F_SENDDP for sending and F_RCVDP for receiving. They can be used to transfer a *fixed* amount of fail-safe data of data types BOOL and INT in a fail-safe manner.

You can find these F-application blocks in the *F-application blocks* block container in the *Distributed Safety* F-library (V1). The F_RCVDP **must** be called at the start of the F-PB. The F_SENDDP **must** be called at the end of the F-PB.

Note that the send signals are sent only after the F_SENDDP call at the end of the relevant F-runtime group execution.

For a detailed description of the F_SENDDP and F_RCVDP F-application blocks, refer to Chapter "FB 223 "F_SENDDP" and FB 224 "F_RCVDP": Sending and Receiving Data via PROFIBUS DP".

See also

FB 223 "F_SENDDP" and FB 224 "F_RCVDP": Send and Receive Data via PROFIBUS DP (Page 229)

8.2.4 Programming Safety-Related Master-Master Communication

Requirements

The following requirements must be met prior to programming:

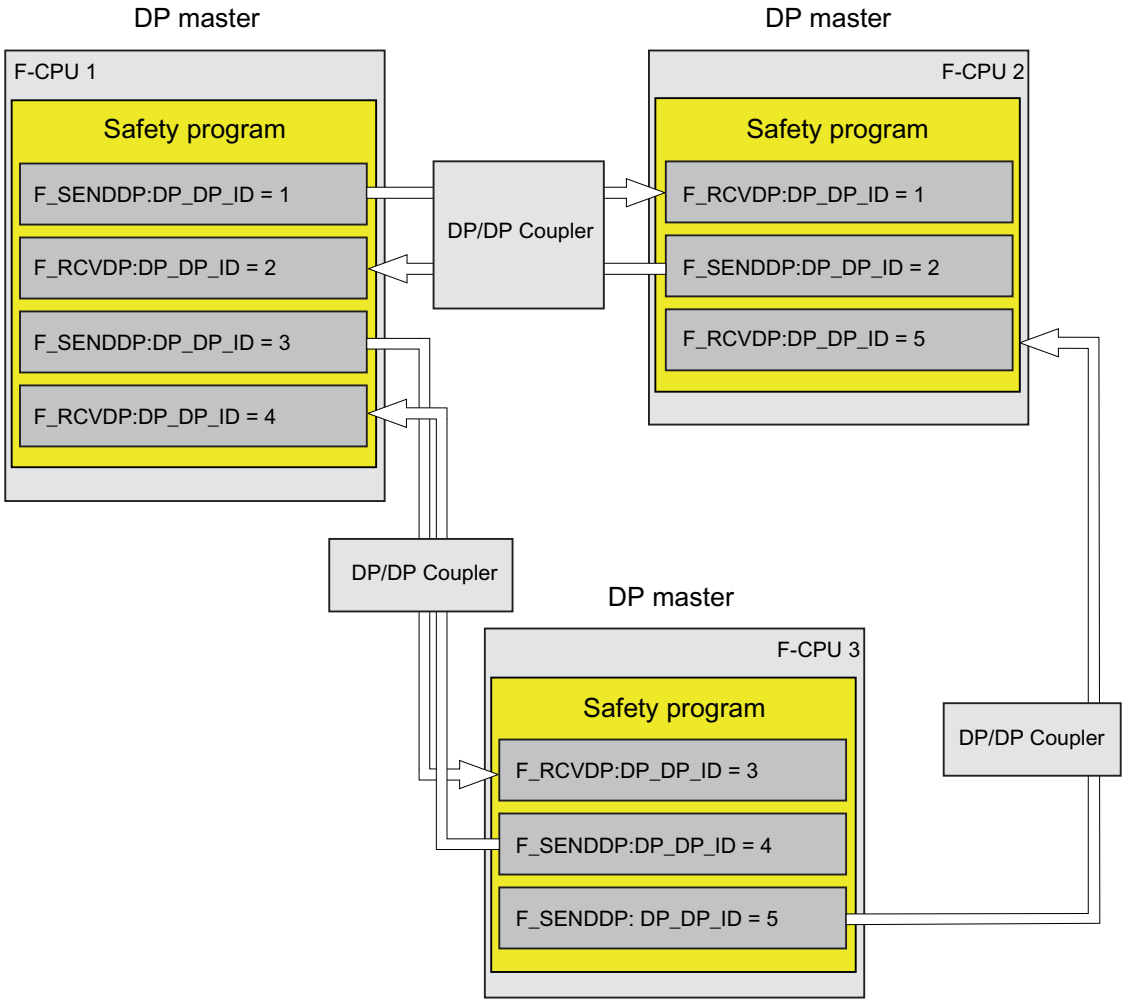
- The address areas for input and output data for the DP/DP coupler must be configured in *HW Config*.
- Both CPUs must be configured as F-CPUs:
  - "CPU contains safety program" option must be selected
  - The password for the F-CPU must be entered

Programming Procedure

1. In the safety program from which data are to be sent, call the F_SENDDP F-application block for sending at the end of the F-PB.
2. In the safety program in which data are to be received, call the F_RCVDP F-application block for receiving at the start of the F-PB.
3. Assign the start addresses of the output and input data address areas of the DP/DP coupler configured in *HW Config* 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 association to the DP_DP_ID inputs. This establishes the association between an F_SENDDP in one F-CPU and an F_RCVDP in the other F-CPU: The associated fail-safe blocks receive the same value for DP_DP_ID.



WARNING

The value for each address association (input parameter DP_DP_ID; data type: INT) is user-defined; however, it must be unique from all other safety-related communication connections in the network.

Note

A separate instance DP must be used for each call of an F SENDDP or F_RCVDP.

The input and output parameters of the F_RCVDP must not be supplied with local data of the F-program block.

You must not use an actual parameter for an output parameter of an F_RCVDP, if it is already being used for an input parameter of the same F_RCVDP call or another F_RCVDP or F_RCVS7 call. The F-CPU can go to STOP if this is not observed. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety program: internal CPU fault; internal error information: 404"

5. Provide the SD_BO_xx inputs of the F_SENDDP with the send signals. To cut down on intermediate signals when transferring block parameters, you can write the value directly to the instance DB of the F_SENDDP using symbolic, fully qualified access (for example, "Name F_SENDDP1".SD_BO_02) before calling the F-SENDDP.
6. Supply the RD_BO_xx outputs of the F-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 (for example, "Name F_RCVDP1".RD_BO_02).
7. Provide the SUBBO_xx and SUBI_xx inputs of the F_RCVDP with the fail-safe values that are to be output by F_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 at input SUBI_xx. If you want to specify constant fail-safe values for data of data type BOOL, use variables "RLO0" or "RLO1" from the F-shared DB. Then, at input SUBBO_xx, enter "F_GLOBDB".RLO0 with fully qualified access if you want to specify a fail-safe value of "0" and "F_GLOBDB".RLO1 if you want to assign a fail-safe value of "1".
   - Specification of dynamic fail-safe values:

     If you want to specify dynamic fail-safe values, define a variable that you can change dynamically through your safety program in an F-DB and declare this variable with fully qualified access at input SUBI_xx or SUBBO_xx.

     ⚠WARNING

     Note that your safety program for dynamically changing a variable for a dynamic fail-safe value can only be processed after the call of the F_RCVDP, because prior to the F_RCVDP call there can be no network in the F-PB and at most there can be one other F_RCVDP. You must therefore assign appropriate initial/actual values for these variables to be output by F_RCVDP in the first cycle after a startup of the F-system.

8. Configure the TIMEOUT inputs of the F_RCVDPs and F_SENDDPs with the required monitoring time.

> **⚠ WARNING**
>
> It can be ensured (from a fail-safe standpoint) that a signal level to be transferred will be captured on the sender side and transferred to the receiver only if the signal is pending for at least as long as the assigned monitoring time (TIMEOUT).
>
> For information on calculating the monitoring times, refer to the *Safety Engineering in SIMATIC S7* system manual.

9. Optional: Evaluate the ACK_REQ output of the F_RCVDP, for example, in the standard user program or on the operator control and monitoring system in order to query or to indicate whether user acknowledgment is required.
10. Provide the ACK_REI input of the F_RCVDP with the signal for the acknowledgment for reintegration.
11. Optional: Evaluate the SUBS_ON output of the F_RCVDP or the F_SENDDP in order to query whether the F_RCVDP is outputting the fail-safe values assigned at the SUBBO_xx and SUBI_xx inputs of the F_RCVDP.
12. Optional: Evaluate the ERROR output of the F_RCVDP or the F_SENDDP, for example, in the standard user program or on the operator control and monitoring system in order to query or to indicate whether a communication error has occurred.
13. Optional: Evaluate the SENDMODE output of the F_RCVDP in order to query whether the F-CPU with the associated F_SENDDP is in deactivated safety mode.

> **⚠ WARNING**
>
> If the F-CPU with the associated F_SENDDP is in deactivated safety mode, you can no longer assume that the data received from this F-CPU were 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 received data. Alternatively, you must output fail-safe values instead of the received data in the F-CPU with the F_RCVDP by evaluating SENDMODE.

See also

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Deactivating Safety Mode (Page 286)

8.2.5 Limits for Data Transfer (Safety-Related Master-Master Communication)

Note

If the data quantities to be transmitted exceed the capacity of the F_SENDDP/F_RCVDP block pair, a second (or third) F_SENDDP/F_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.

8.3 Safety-Related Master-I-Slave Communication

8.3.1 Configuring Address Areas (Safety-Related Master-I-Slave Communication)

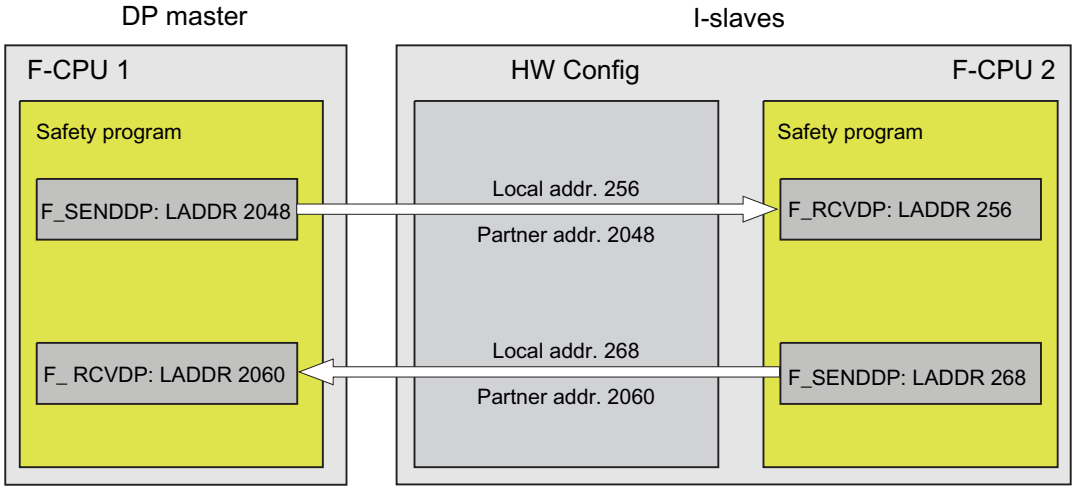
Introduction

Safety-related communication between the safety program of the F-CPU of the 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, as in standard systems.

You do not need any additional hardware for the master-I-slave communication.

Configuring Address Areas

For every communication connection between two F-CPUs, you must configure address areas in *HW Config*. In the figure below, each of the two F-CPUs will be able to send and receive data (bidirectional communication).



You configure the following in the object properties for the I-slave:

- A local address (I-slave) and a partner address (DP master) for sending data to the DP master
- A local address (I-slave) and a partner address (DP master) for receiving data from the DP master

You assign the configured addresses to the LADDR parameter of the corresponding F_SENDDP and F_RCVDP F-application blocks in the safety programs.

Assigned Address Areas

Each of the local and partner addresses represents a start address of an address area of input and output data. Once the local and partner addresses are configured, the address areas are automatically assigned. The assigned address areas for a send connection and a receive connection are shown in the following table:

Communication Connection	Assigned Address Area in the F-CPU of the ...
Send: I-slave to DP master	I-slaves: 12 bytes of output data and 6 bytes of input data
	DP masters: 12 bytes of input data and 6 bytes of output data
Receive: I-slave from DP master	I-slaves: 12 bytes of input data and 6 bytes of output data
	DP masters: 12 bytes of output data and 6 bytes of input data

Note

We recommend that you use addresses outside the process image as the local and partner addresses, since the process image should be reserved for the address areas of modules.

8.3.2 Configuring Safety-Related Master-I-Slave Communication

Requirements

You have created a project in *STEP 7*.

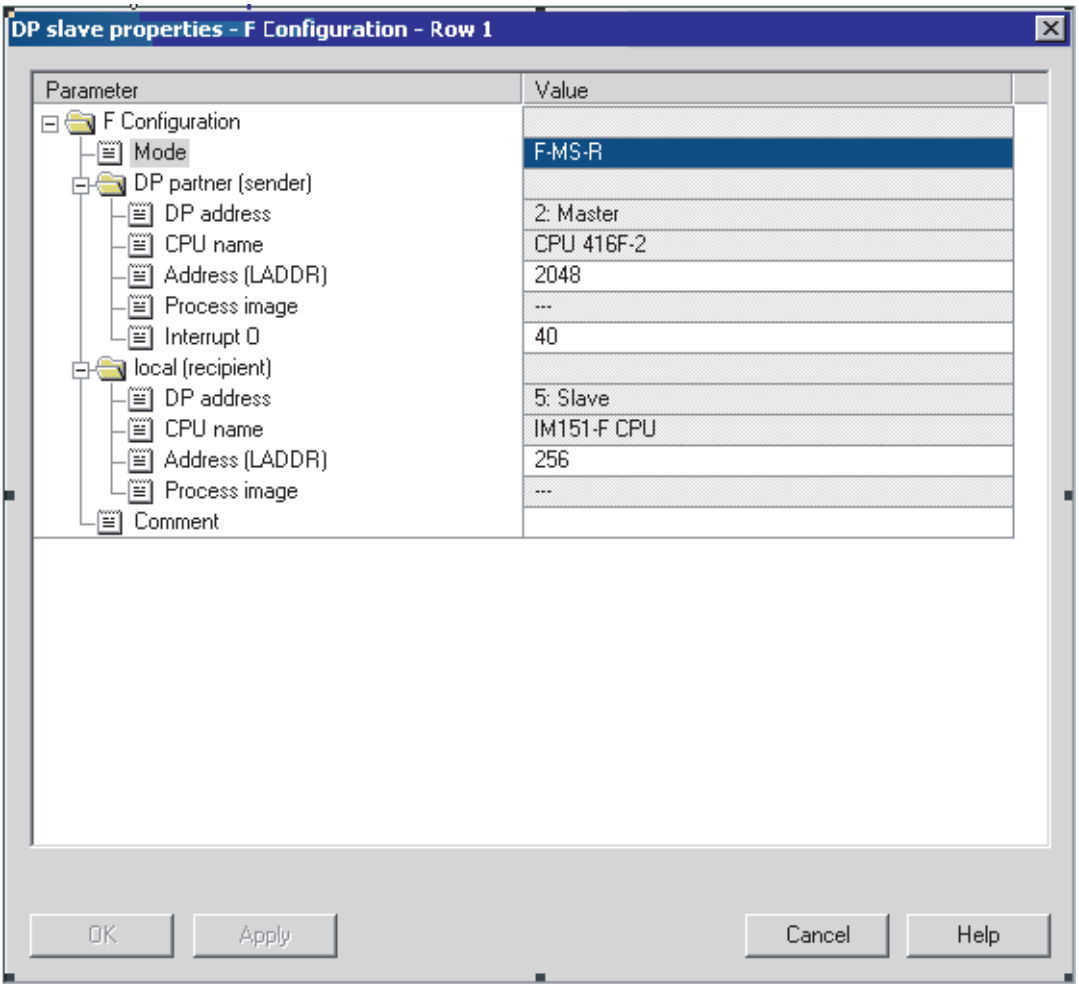
Procedure for Configuring Master-I-Slave Communication (Example with Bidirectional Communication)

1. Create a station in your project (in *SIMATIC Manager*, for example, an S7-300 station).
2. Assign an F-CPU to this station (from the hardware catalog in *HW Config*).
3. Configure this CPU as a DP slave (in *HW Config*, in the "Operating Mode" tab of the object properties for the DP interface of the CPU).
4. Create another station, and assign an F-CPU (see steps 1 and 2).
5. Configure this CPU as a DP master (in *HW Config*, in the "Operating Mode" tab of the object properties for the DP interface of the CPU).
6. In the hardware catalog under "Configured stations," select the station type of the I-slave (for example, "CPU 31x") and place it on the DP master system.
7. Link the I-slave to the DP master in the Connection dialog, which opens automatically.

   Now you can define the address areas for safety-related master-I-slave communication:
8. In the "F-Configuration" tab of the object properties for the I-slave, select "New".

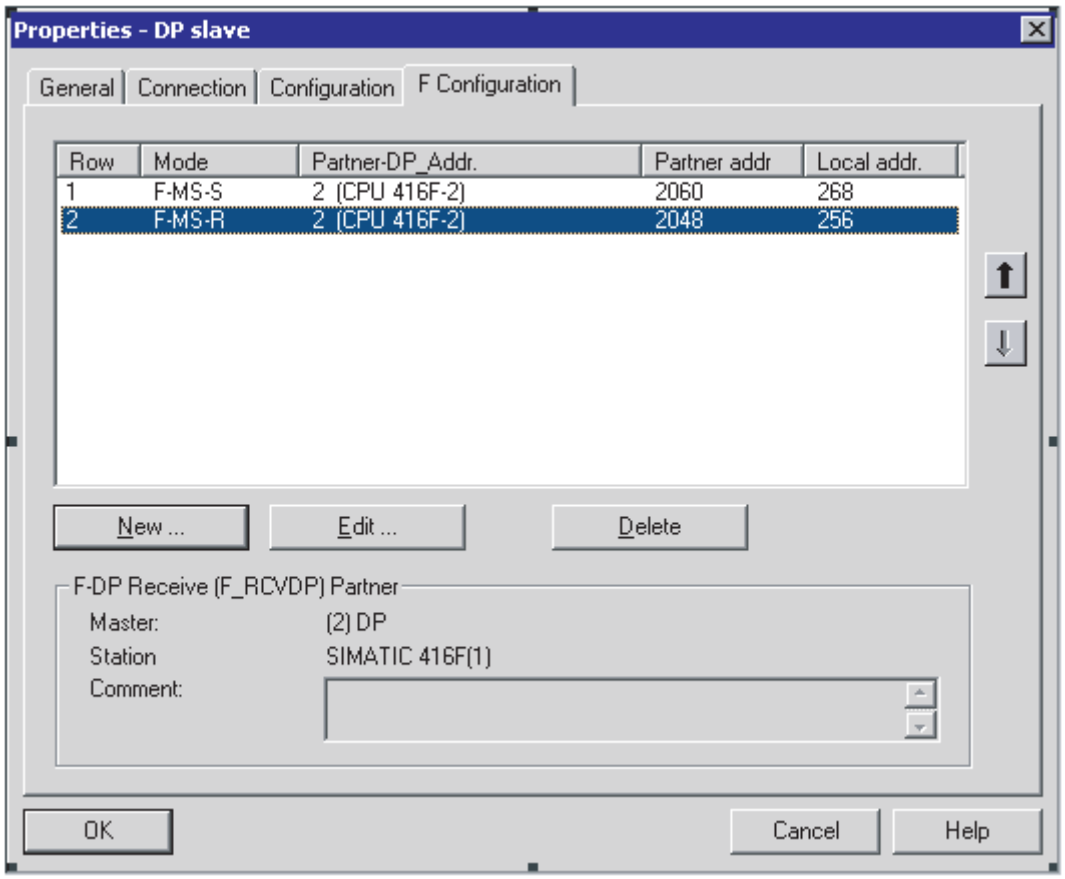
9. In the next dialog, make the following entries for the receive connection from the DP master for our example:
   - For "Mode: F-MS-R" (receive via fail-safe master-I-slave communication)
   - For "DP partner (sender): Address (LADDR): 2048"
   - For "Local (receiver): Address (LADDR): 256"
   - Accept the defaults for the other parameters in the dialog box.

   The dialog box has the following appearance:



10. Confirm your entries with "OK".
11. In the "F-Configuration" tab of the object properties for the I-slave, select "New".
12. In the next dialog, make the following entries for the send connection to the DP master for our example:
    - For "Mode: F-MS-R" (send via fail-safe master-I-slave communication)
    - For "DP partner (receiver): Address (LADDR): 2060"
    - For "Local (sender): Address (LADDR): 268"

13. Confirm your entries with "OK".
    This results in two configuration lines for this example:



Note

Entries are automatically made in the "Configuration" tab in the object properties for the I-slave based on the configuration in the "F-Configuration" tab. These entries must not be modified. Otherwise, safety-related master-I-slave communication is not possible.

You can obtain the assigned address areas in the DP master and I-slave in the "Configuration" tab.

Disabling Active Coupling of an I-Slave

Before you can disable "active coupling" of an I-slave, you must delete all safety-related communication connections to other F-CPUs or F-modules in the "F-Configuration" tab.

Additional Information

You will find a description of the parameters in the *context-sensitive online help for the "F-Configuration" tab*.

For more information on master-I-slave communication, refer to the *STEP 7 Online Help*.

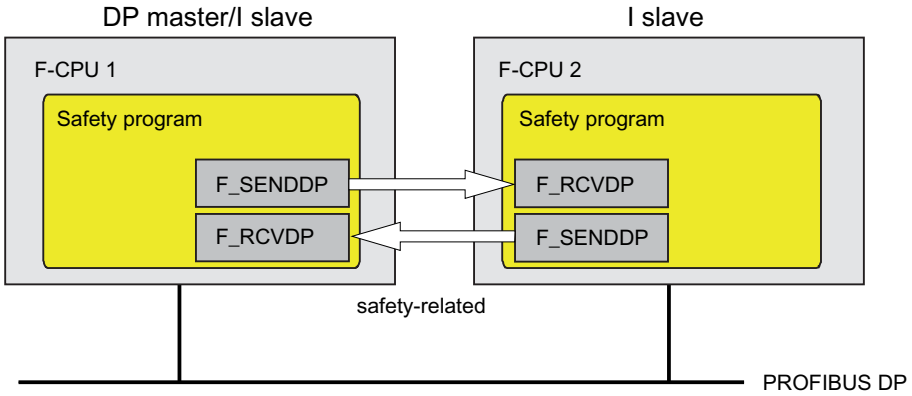
For information on address areas, partial process images, and supported interrupt OBs, refer to the *technical specifications for the F-CPU you are using*.

8.3.3 Communication by Means of F_SENDDP and F_RCVDP (Safety-Related Master-I-Slave/I-Slave-I-Slave Communication)

Introduction

The procedure for programming safety-related master-I-slave communication or safety-related I-slave-I-slave communication is the same as for programming safety-related master-master communication. For this reason, only the differences are described in the following section.

Communication by Means of F_SENDDP and F_RCVDP



For safety-related communication between the F-CPUs of the DP master and an I-slave or between the F-CPUs of several I-slaves, you make use of the F application blocks F_SENDDP for sending and F_RCVDP for receiving. They can be used to transfer a *fixed* amount of fail-safe data of data types BOOL and INT in a fail-safe manner.

You can find these F-application blocks in the *F-application blocks* block container in the *Distributed Safety* F-library (V1). The F_RCVDP **must** be called at the start of the F-PB. The F_SENDDP **must** be called at the end of the F-PB.

Note that the send signals are sent only after the F_SENDDP call at the end of the relevant F-runtime group execution.

For a detailed description of the F_SENDDP and F_RCVDP F-application blocks, refer to Chapter "FB 223 "F_SENDDP" and FB 224 "F_RCVDP": Sending and Receiving Data via PROFIBUS DP".

Assigning F-CPUs to F_SENDDP/F_RCVDP

Assign the F-CPUs to F_SENDDPs/F_RCVDPs as follows:

- Configure the address areas (local and partner addresses) for the DP master and the I-slave(s) in *HW Config*.
- Specify the following addresses for master-I-slave communication in the safety program of the F-CPU of the DP master:
  - At F_SENDDP at input parameter LADDR, the partner address for sending ("F-Configuration" tab: row Mode: "F-MS-R")
  - At F_RCVDP at input parameter LADDR, the partner address for receiving ("F-Configuration" tab: row Mode: "F-MS-S")
- Specify the following addresses for master-I-slave or I-slave-I-slave communication in the safety program of the F-CPU of an I-slave:
  - At F_SENDDP at input parameter LADDR, the local address for sending ("F-Configuration" tab: row Mode: "F-MS-S" or "F-DX-S")
  - At F_RCVDP at input parameter LADDR, the local address for receiving ("F-Configuration" tab: row Mode: "F-MS-R" or "F-DX-R")

Make these assignments for each F-CPU involved.

Note

Thus, the following always applies for safety-related master-I-slave and I-slave-I-slave communication:

- At the F_SENDDP/F_RCVDP of the **DP master**, always enter **the partner addresses** for the communication connections (from *HW Config*, "F-Communication" tab of the I-slave).
- At the F_SENDDP/F_RCVDP of a **DP slave** always enter **the local addresses** for the communication connections (from *HW Config*, "F-Communication" of the I-slave).

See also

Programming Safety-Related Master-Master Communication (Page 135)

FB 223 "F_SENDDP" and FB 224 "F_RCVDP": Send and Receive Data via PROFIBUS DP (Page 229)

8.3.4 Programming Safety-Related Master-I-Slave and I-Slave-I-Slave Communication

Requirements

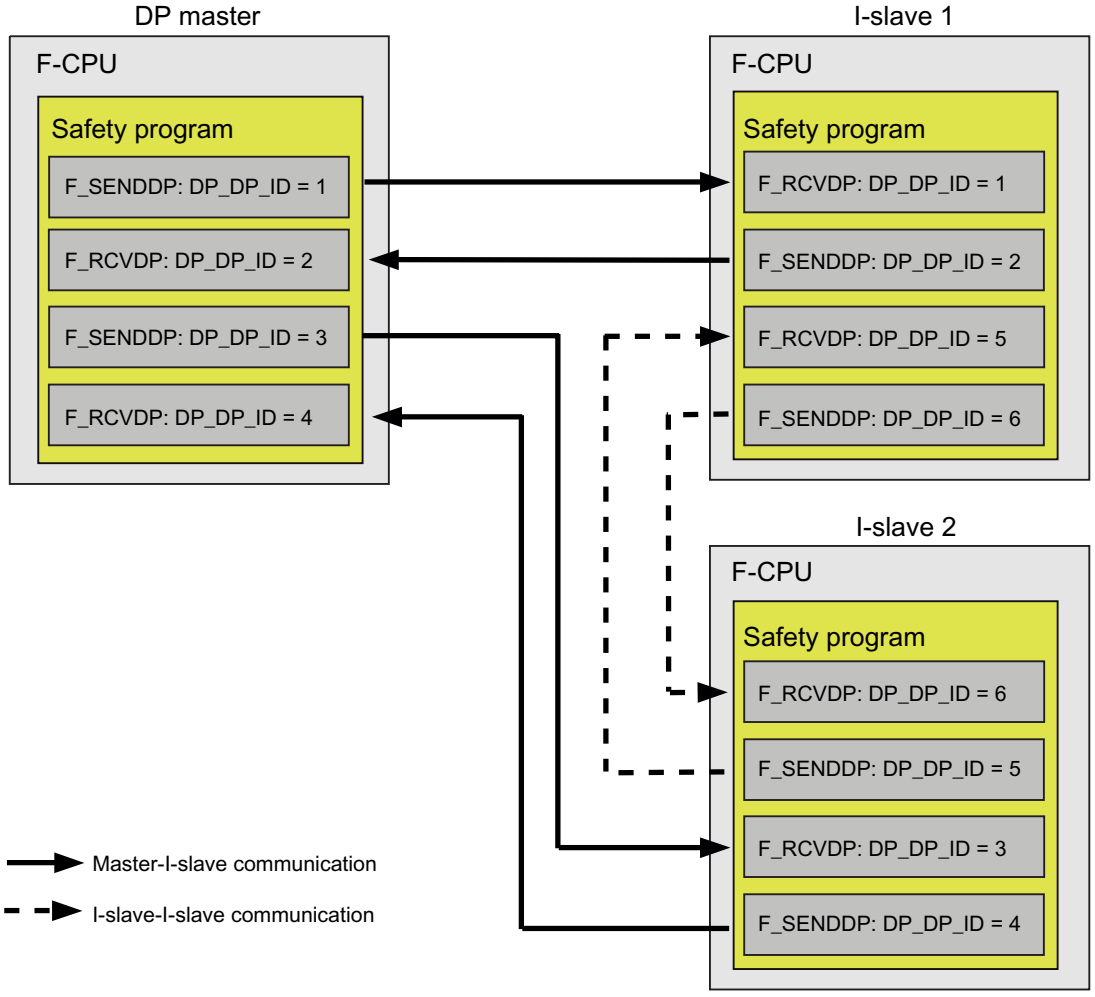
The following requirements must be met prior to programming:

- The address areas (local and partner addresses) for the DP master and the I-slave(s) must be configured in *HW Config*.
- Both CPUs must be configured as F-CPUs:
  - "CPU contains safety program" option must be selected
  - The password for the F-CPU must be entered

Programming Procedure

The procedure for programming safety-related master-I-slave communication or I-slave-I-slave communication is the same as for programming safety-related master-master communication.

The figure below contains an example of how to specify the address relationships at the inputs of F application blocks F_SENDDP and F_RCVDP for two safety-related master-I-slave communication connections and one I-slave-I-slave communication connection.



⚠WARNING

The value for each address association (input parameter DP_DP_ID; data type: INT) is user-defined; however, it must be unique from all other safety-related communication connections in the network.

Note

A separate instance DP must be used for each call of an F SENDDP or F_RCVDP.

The input and output parameters of the F_RCVDP must not be supplied with local data of the F-program block.

You must not use an actual parameter for an output parameter of an F_RCVDP if it is already being used for an input parameter of the same F_RCVDP call or another F_RCVDP or F_RCVS7 call. The F-CPU can go to STOP if this is not observed. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety Program: internal CPU fault; internal error information: 404"

⚠WARNING

If the F-CPU with the associated F_SENDDP is in deactivated safety mode, you can no longer assume that the data received from this F-CPU were 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 received data. Alternatively, you must output fail-safe values instead of the received data in the F-CPU with the F_RCVDP by evaluating SENDMODE.

See also

Programming Safety-Related Master-Master Communication (Page 135)

Deactivating Safety Mode (Page 286)

8.3.5 Limits for Data Transfer (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 an F_SENDDP/F_RCVDP block pair, you can use additional F_SENDDP/ F_RCVDP calls. Configure additional communication connections for this purpose. Remember the maximum limit of 244 bytes of input and 244 bytes of output data for transfer between an I-slave and a DP master.

The following table shows you the amount of output data and input data that is assigned for safety-related communication connections:

Safety-Related Communication	Communication Connection	Assigned Input and Output Data			
		Between I-Slave 1 and DP Master		Between I-Slave 2 and DP Master	
		Output Data	Input Data	Output Data	Input Data
Master to I-slave	Send: I-slave 1 to DP master	12 bytes	6 bytes	-	-
	Receive: I-slave 1 from DP master	6 bytes	12 bytes	-	-
I-slave-I-slave	Send: I-slave 1 to I-slave 2	12 bytes	-	6 bytes	-
	Receive: I-slave 1 from I-slave 2	6 bytes	-	12 bytes	-

If necessary, you should also taken into account fail-safe I-slave-I-slave communication (F-DX-modules), master-slave connections (MS), or direct data exchange connections (DX) used to exchange data within your standard user program as part of the maximum limit of 244 bytes of input data and 244 bytes of output data for transmission between an I-slave and a DP master.

You can check whether you are within the maximum limit of 244 bytes of input data and 244 bytes of output data for all configured safety-related and standard communication connections in the "Configuration" tab in the object properties for the I-slave. Include all lines with MODE "MS" in the "Configuration" tab. The lines with MODE "DX" are not included.

8.4 Safety-Related I-Slave-I-Slave Communication

8.4.1 Configuring Address Areas (Safety-Related I-Slave-I-Slave Communication)

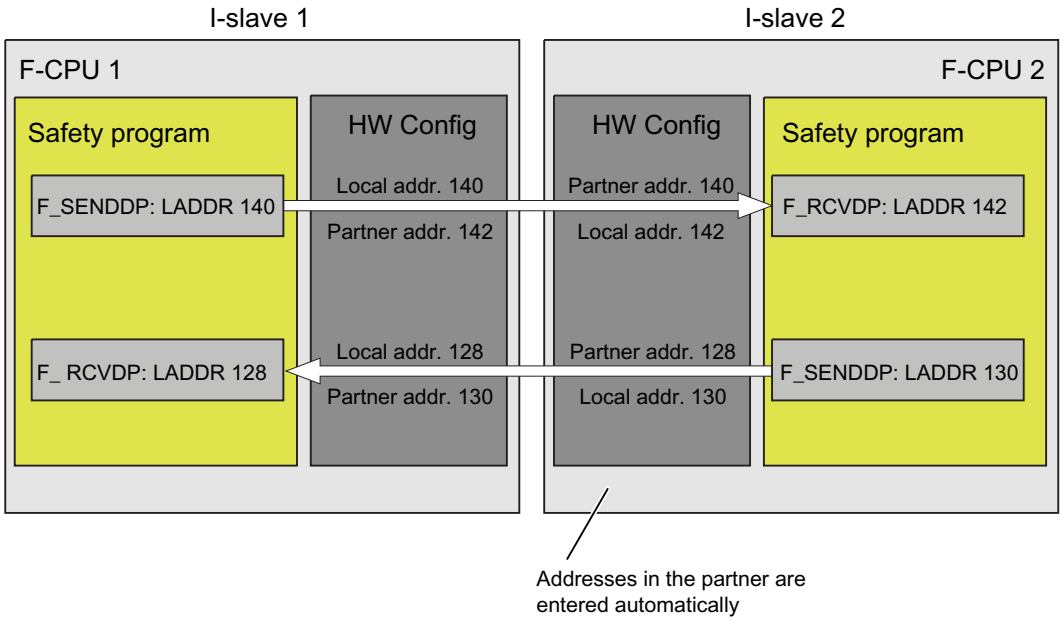
Introduction

Safety-related communication between the safety program of the F-CPUs of I-slaves takes place using direct data exchange – same as in standard programs.

You do not need any additional hardware for I-slave-I-slave communication.

Configuring Address Areas

For every communication connection between two F-CPUs, you must configure address areas in *HW Config*. In the figure below, each of the two F-CPUs will be able to send and receive data (bidirectional communication).



You configure the following in the object properties for I-slave 1:

- For sending to I-slave 2, a local address (I-slave 1) and a partner address (I-slave 2)
- For receiving from I-slave 2, a local address (I-slave 1) and a partner address (I-slave 2)

No further configuration of communication is necessary in the object properties for I-slave 2. The addresses are entered automatically in the object properties for I-slave 2.

You assign the configured addresses to the LADDR parameter of the corresponding F_SENDDP and F_RCVDP F-application blocks in the safety programs.

Assigned Address Areas

Each of the local and partner addresses represents a start address of an address area of input and output data. Once the local and partner addresses are configured, the address areas are automatically assigned. The assigned address areas for a send connection and a receive connection are shown in the following table:

Communication Connection	Assigned Address Areas in the F-CPU* of the ...
Send: I-slave 1 to I-slave 2	I-slave 1: 12 bytes of output data and 6 bytes of input data
	I-slave 2: 12 bytes of input data and 6 bytes of output data
	DP masters: 12 + 6 bytes of input data
Receive: I-slave 1 from I-slave 2	I-slave 1: 12 bytes of input data and 6 bytes of output data
	I-slave 2: 12 bytes of output data and 6 bytes of input data
	DP masters: 12 + 6 bytes of input data
* The CPU of the DP master can be an F-CPU or a standard CPU. Whether or not the PROFIBUS DP interface of the standard CPU supports direct data exchange can be found in the info text for the relevant CPU in the hardware catalog in *HW Config*.	

Note

We recommend that you use addresses outside the process image as the local and partner addresses, since the process image should be reserved for the address areas of modules.

8.4.2 Configuring Safety-Related I-Slave-I-Slave Communication

Requirements

You have created a project in *STEP 7*.

Procedure for Configuring I-Slave-I-Slave Communication (Example with Bidirectional Communication)

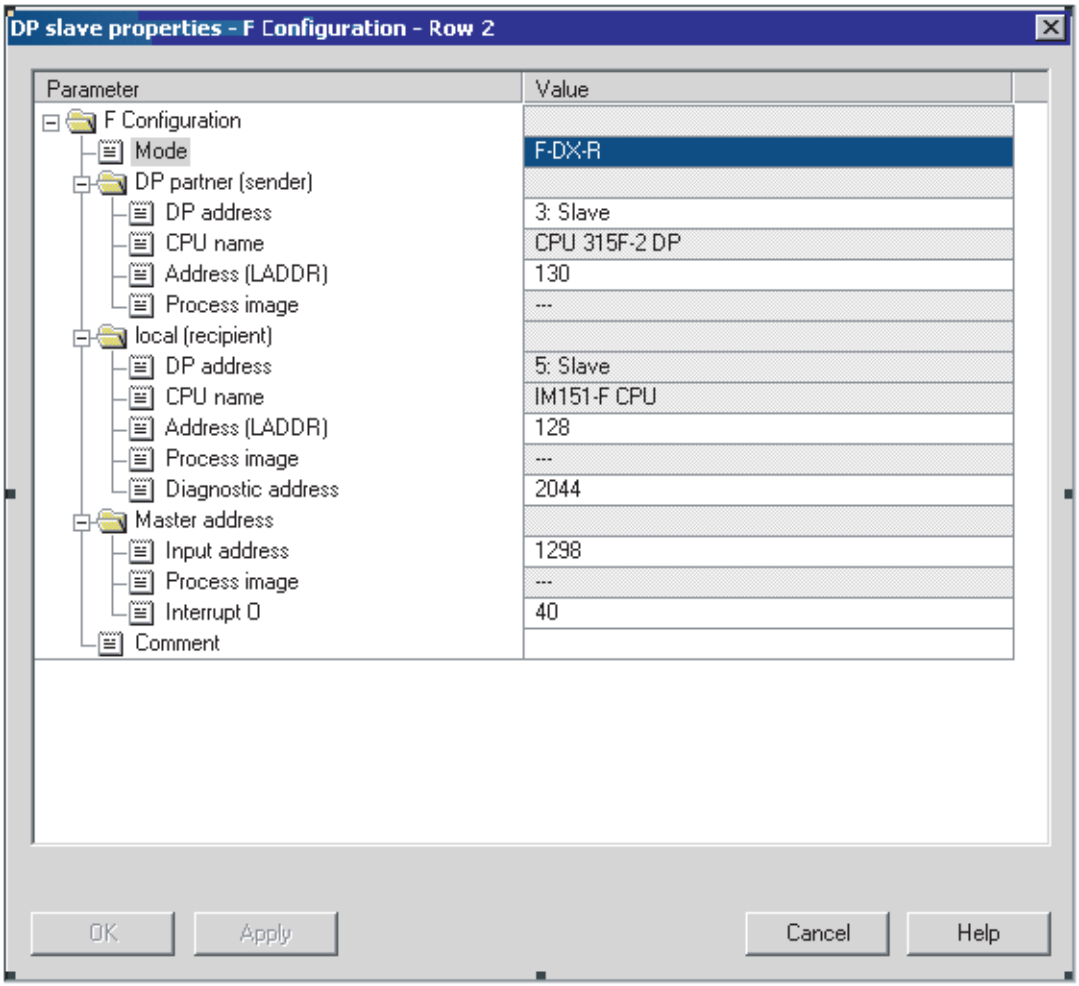
1. Create a station in your project (in *SIMATIC Manager*, for example, an S7-300 station).
2. Assign an F-CPU to this station (from the hardware catalog in *HW Config*).
3. Configure this CPU as a DP slave (in *HW Config*, in the "Operating Mode" tab of the object properties for the DP interface of the CPU).
4. Follow steps 1 to 3 to configure another DP slave (I-slave).
5. Create another station, and assign an F-CPU (see steps 1 and 2).
6. Configure this CPU as a DP master (in *HW Config*, in the "Operating Mode" tab of the object properties for the DP interface of the CPU).

   Note: The CPU of the DP master can be an F-CPU or a standard CPU.
7. In the hardware catalog, under "Configured stations," select the station type of one I-slave (for example, the "CPU 31x") and place it on the DP master system.
8. Link the I-slave to the DP master in the Connection dialog, which opens automatically.
9. After steps 7 and 8, link the second I-slave to the DP master.

   Now you can define the address areas for safety-related I-slave-I-slave communication:
10. In the "F-Configuration" tab of the object properties for I-slave 1, select "New".

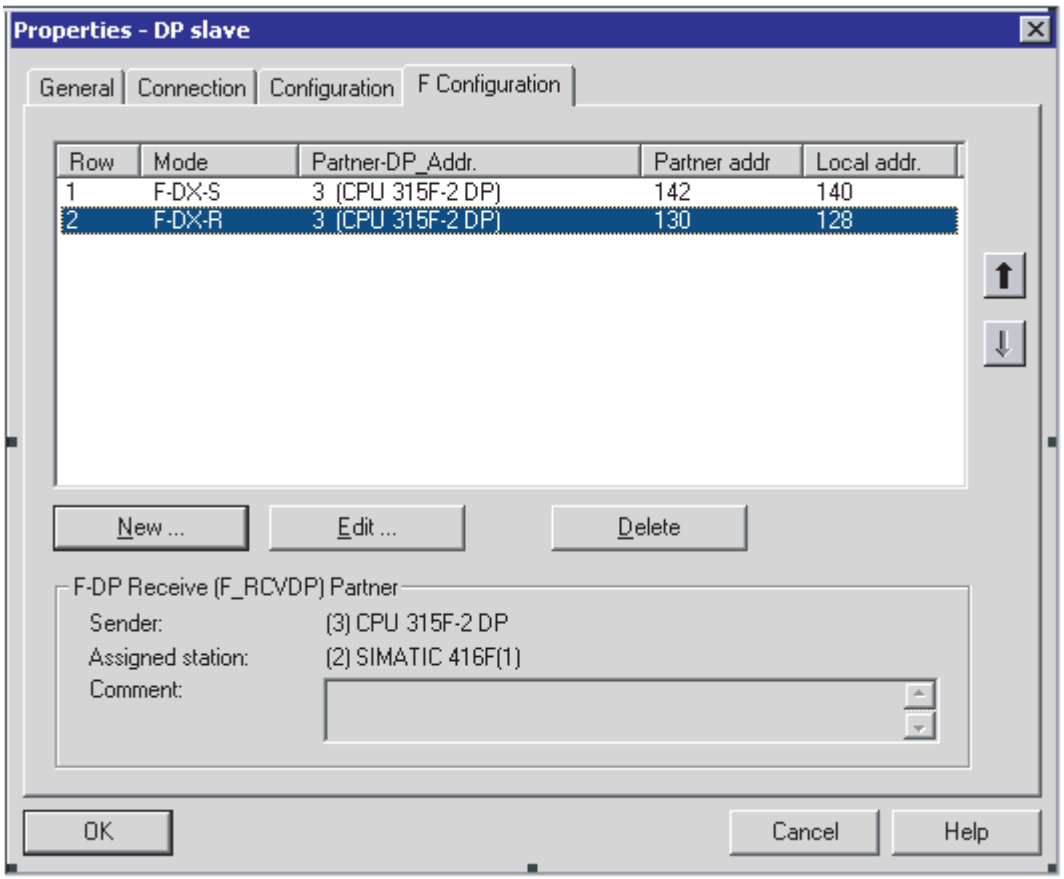
11. In the next dialog, make the following entries for the receive connection from I-slave 2 in our example:
   – For "Mode: F-DX-R" (receive via fail-safe I-slave-I-slave communication)
   – For "DP partner (sender): DP address: 5: Slave (PROFIBUS address); address (LADDR): 130"
   – For "Local (receiver): Address (LADDR): 128"
   – Accept the defaults for the other parameters in the dialog box.

   The dialog box has the following appearance:



12. Confirm your entries with "OK".
13. In the "F-Configuration" tab of the object properties for I-slave 1, select "New".
14. In the next dialog, make the following entries for the send connection to I-slave 2 for our example:
   – For "Mode: F-DX-S" (send via fail-safe I-slave-I-slave communication)
   – For "DP partner (receiver): DP address: 5: Slave; address (LADDR): 142"
   – For "Local (sender): Address (LADDR): 140"
   – Accept the defaults for the other parameters in the dialog box.

15. Confirm your entries with "OK".
    This results in two configuration lines for this example:



Note

In the object properties for the respective I-slave, entries are automatically made in the "Configuration" tab based on the configuration in the "F-Configuration" tab. These entries must not be modified. Otherwise, safety-related I-slave-I-slave communication is not possible.

You can obtain the assigned address areas in the DP master and the I-slaves in the "Configuration" tab.

Disabling Active Coupling of an I-Slave

Before you can disable "active coupling" of an I-slave, you must delete all safety-related communication connections to other F-CPUs or F-modules in the "F-Configuration" tab.

Additional Information

You will find a description of the parameters in the *context-sensitive online help for the "F-Configuration" tab*.

For information on address areas, partial process images, and supported interrupt OBs, refer to the *technical specifications for the CPU you are using*.

8.4.3 Communication by Means of F_SENDDP and F_RCVDP (Safety-Related I-Slave-I-Slave Communication)

Reference

For a description, refer to Chapter "Communication by Means of F_SENDDP and F_RCVDP (Safety-Related Master-I-Slave/I-Slave-I-Slave Communication)".

8.4.4 Programming Safety-Related I-Slave-I-Slave Communication

Reference

For a description, refer to Chapter "Programming Safety-Related Master-I-Slave/I-Slave-I-Slave Communication".

8.4.5 Limits for Data Transfer (Safety-Related I-Slave-I-Slave Communication)

Limits for Data Transfer

For a description, refer to Chapter "Limits for Data Transfer (Safety-Related Master-I-Slave/I-Slave-I-Slave Communication)".

8.5 Safety-Related I-Slave-Slave Communication

8.5.1 Configuring Address Areas (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 – same as in standard programs. The channels of the F-I/O in the safety program of the F-CPU of the I-slave are accessed via the process image (PII and PAA) as described in Chapter "F-I/O Access".

For F-I/O access via safety-related I-slave-I-slave communication, an F-I/O DB is generated automatically in the safety program of the F-CPU when the program is compiled in *HW Config*.

You do not need any additional hardware for I-slave-slave communication.

Restrictions

Note

Safety-related I-slave-slave communication is possible with F-I/O in a DP slave that supports safety-related I-slave-slave communication, e.g., with all ET 200S F-modules and with all S7-300 fail-safe signal modules with IM 153-2, order no. 6ES7 153-2BA01-0XB0 or higher, firmware version > V4.0.0.

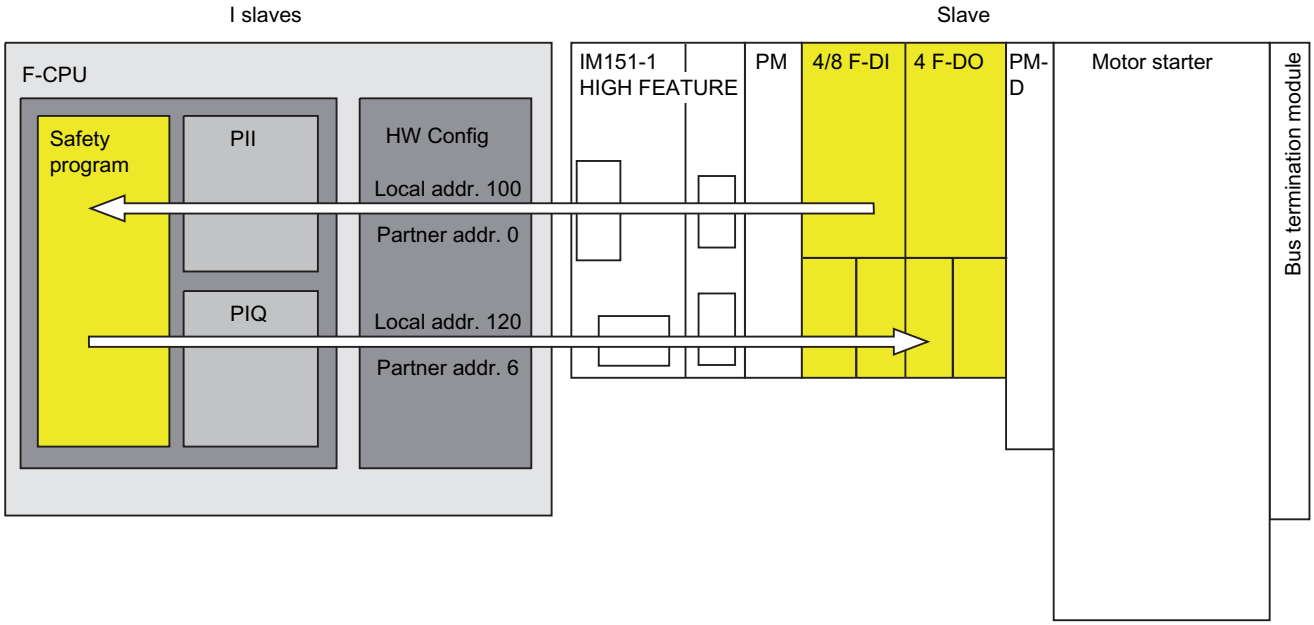
Note

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 the safety-related communication (communication error) between the F-CPU and the F-I/O assigned to the I-slave. That is, after startup of the F-system, the F-I/O are not reintegrated automatically. Rather, they are only reintegrated after a user acknowledgment with a positive edge on the ACK_REI variable of the F-I/O DB (see also Chapters "Passivation and Reintegration of F-I/O after Communication Errors" and "Passivation and Reintegration of F-I/O after F-System Startup").

Configuring Address Areas

For every communication connection from an F-CPU of an I-slave to an F-I/O in a slave, you must configure address areas in *HW Config*. The figure below shows an example for an ET 200S with F-DI and F-DO modules.



You can configure the following in the object properties for the I-slave for each I-slave-slave communication with an F-IO:

- A local address (safety program) that you can use to access the F-I/O in the safety program of the I-slave
- A partner address (F-I/O) of the F-I/O in the DP master

No configuration of communication is necessary in the object properties for the F-I/O of the DP slave and DP master.

Assigned Address Areas

Each of the local and partner addresses represents a start address of an address area of input and output data. Once the local and partner addresses are configured, the address areas are automatically assigned. An example of assigned address areas for I-slave-I-slave communication with F-I/O is shown in the table below for a 4/8F-DI and a 4 F-DO of ET 200S:

Communication Connection	Assigned Address Areas in ... *
I-slave-slave communication with 4/8 F-DI	Of F-CPU of I-slave: 6 bytes of input data and 4 bytes of output data
	Of F-CPU** of DP master: 6 + 4 bytes of input data
I-slave-slave communication with 4 F-DO	Of F-CPU of I-slave: 5 bytes of output data and 5 bytes of input data
	Of F-CPU** of DP master: 5 +5 bytes of input data
* Example for 4/8 F-DI and 4 F-DO of ET 200S (for specific address relationship, see F-I/O manuals) ** The CPU of the DP master can be an F-CPU or a standard CPU. Whether or not the PROFIBUS DP interface of the standard CPU supports direct data exchange can be found in the info text for the relevant CPU in the hardware catalog in *HW Config*.	

Note

You must use addresses within the process image for the local addresses because communication is taking place with real F-I/O.

8.5.2 Configuring Safety-Related I-Slave-Slave Communication

Requirements

You have created a project in *STEP 7*.

Procedure for Configuring I-Slave-Slave Communication

In this section we demonstrate how to configure the address areas of the above figure.

1. Create a station in your project (in *SIMATIC Manager*, for example, an S7-300 station).
2. Assign an F-CPU to this station (from the hardware catalog in *HW Config*).
3. Configure this CPU as a DP slave (in *HW Config*, in the "Operating Mode" tab of the object properties for the DP interface of the CPU).
4. Create another station, and assign a standard CPU or F-CPU (see steps 1 and 2).
5. Configure this CPU as a DP master (in *HW Config*, in the "Operating Mode" tab of the object properties for the DP interface of the CPU).
6. In the hardware catalog, select an IM 151 HIGH FEATURE, order no. 6ES7 151-1BA01-0AB0 or higher, and place it on the DP master system.
7. Assign a power module, a 4/8 F-DI module, and a 4 F-DO module to the IM using a drag-and-drop operation.
8. In the hardware catalog under "Configured stations," select the station type of the I-slave (for example, "CPU 31x") and place it on the DP master system.
9. Link the I-slave to the DP master in the Connection dialog, which opens automatically.

   Now you can define the F-I/O for safety-related I-slave-slave communication:
10. In the "F-Configuration" tab of the object properties for the I-slave, select "New".
11. In the next dialog, make the following entries for the connection to the 4/8 F-DI module in our example:
    - For "Mode: F-DX-Module" (fail-safe I-slave-slave communication)
    - For "DP partner (F-I/O)":

      "DP address: 1: Slave" (PROFIBUS address of slave with F-I/O);
      "Address (LADDR): 0: 4/8 F-DI" (starting address of F-I/O)
    - For "Local (safety program): Address (LADDR): 100" (starting address of F-I/O via which access is made in the safety program of the F-CPU of the I-slave)
    - Accept the defaults for the other parameters in the dialog box.

Note

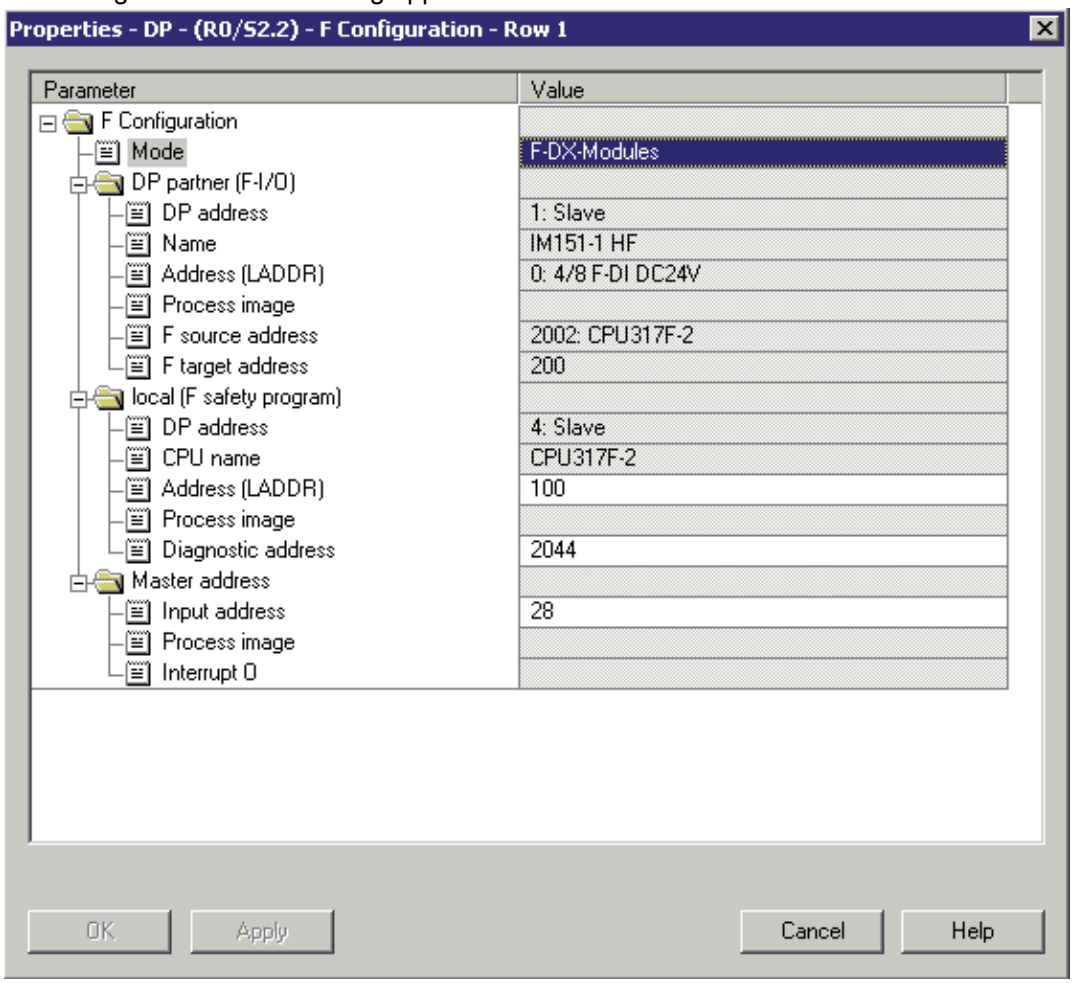
"DP partner (F-I/O)"

For the "DP address", a list field displays the PROFIBUS addresses of possible DP slaves that support safety-related I-slave-slave communication. Select the desired DP slave from this list.

Note, however, that the list may include DP slaves that are not assigned to the DP master system containing the I-slave. You must not select these.

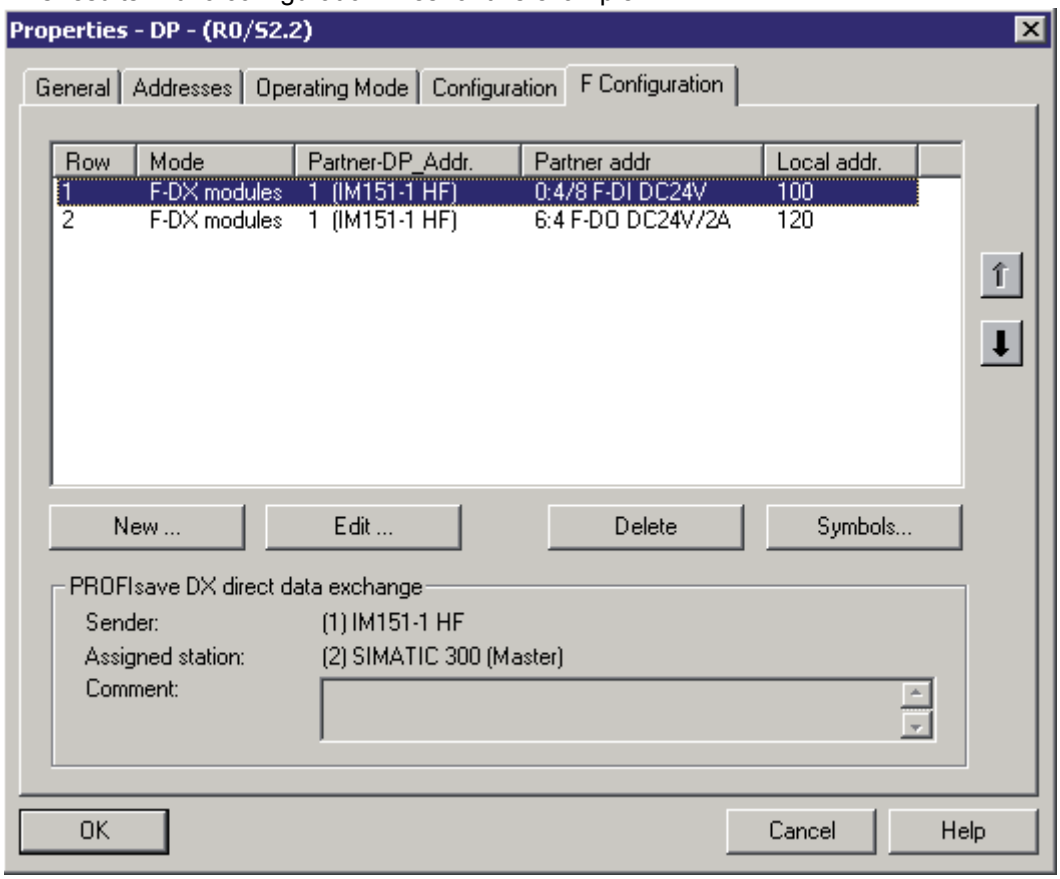
For the "Address (LADDR)", a list field displays the start addresses of the F-I/O of the selected DP slave. Select the desired F-I/O from this list.

The dialog box has the following appearance:



12. Confirm your entry with "OK".
13. In the "F-Configuration" tab of the object properties for the I-slave, select "New".

14. In the next dialog, make the following entries for the connection to the 4 F-DO module for our example:
    - For "Mode: F-DX-Module" (fail-safe I-slave-slave communication)
    - For "DP partner (F-I/O)":
      "DP address: 1: Slave" (PROFIBUS address of slave with F-I/O);
      "Address (LADDR): 6: 4 F-DO" (starting address of F-I/O)
    - For "Local (safety program): Address (LADDR): 120" (starting address of F-I/O via which access is made in the safety program of the F-CPU of the I-slave)
    - Accept the defaults for the other parameters in the dialog box.
15. Confirm your entries with "OK".
    This results in two configuration lines for this example:



Note

Entries are automatically made in the "Configuration" tab in the object properties for the I-slave based on the configuration in the "F-Configuration" tab. These entries must not be modified. Otherwise, safety-related I-slave-slave communication is not possible.

You can obtain the assigned address areas in the DP master and I-slave in the "Configuration" tab.

Change in Configuration of I-Slave-Slave Communication

⚠ WARNING

If you have configured a new I-slave-slave communication for an F-I/O or have deleted an existing I-slave-slave communication, you must save and compile the hardware configuration of the station of the DP master as well as the hardware configuration of the station of the I-slave and download them to the station of the DP master or I-slave.

The collective signature of the safety program of the F-CPU of the I-slave and the collective signature of the safety program of 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).

Disabling Active Coupling of an I-Slave

Before you can disable "active coupling" of an I-slave, you must delete all safety-related communication connections to other F-CPUs or F-modules in the "F-Configuration" tab.

Additional Information

You will find a description of the parameters in the *context-sensitive online help for the "F-Configuration" tab*.

For information on address areas, process images, and supported interrupt OBs, refer to the *technical specifications for the CPU you are using*.

8.5.3 F-I/O Access for Safety-Related I-Slave-Slave Communication

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 the I-slave. In the I-slave, you reference the F-I/O using the starting address that you configured as "Address (LADDR)" under "Local (safety program)" in the "F-Configuration" tab. Direct I/O access is not permitted. The channels of an F-I/O can only be accessed from one F-runtime group.

⚠ WARNING

Due to the special safety protocol, the F-I/O occupy a larger area of the process image than is required for the channels that are actually present on the F-I/O. To find out the area of the process image where the channels (user data) 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 are permitted to be accessed.

Note that for certain F-I/O (such as S7-300 F-SMs and ET 200S 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

F-I/O Access (Page 95)

8.5.4 Limits for Data Transfer (Safety-Related I-Slave-Slave Communication)

Limits for Data Transfer

Note the maximum limit of 244 bytes of input data and 244 bytes of output data for transfer between an I-slave and a DP master.

An example of the amount of output and input data that are assigned for safety-related communications is shown in the table below for a 4/8 F-DI and a 4 F-DO of ET 200S:

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-I-slave communication with 4/8 F-DI	4 bytes	6 bytes
	I-slave-I-slave communication with 4 F-DO	5 bytes	5 bytes
* Example for 4/8 F-DI and 4 F-DO of ET 200S			

If necessary, you should also take into account fail-safe master-I-slave communication (F-MS-R, F-MS-S) and master-slave connections (MS) or
direct data exchange connections (DX) used to exchange data within your standard user program as part of the maximum limit of 244 bytes of input data and 244 bytes of output data for transmission between an I-slave and a DP master.

You can check whether you are within the maximum limit of 244 bytes of input data and 244 bytes of output data for all configured safety-related and standard communication connections in the "Configuration" tab in the object properties for the I-slave. Include all lines with MODE "MS" in the "Configuration" tab. The lines with MODE "DX" are not included.

8.6 Safety-Related IO Controller-IO Controller Communication

Requirements

Safety-related communication between safety programs of the F-CPUs of IO controllers takes place over a PN/PN coupler (order number 6ES7158-3AD00-0XA0) that you set up between the F-CPUs.

For this communication you will need HSP 101 for STEP 7 V5.4 SP1 or the GSD file for the PN/PN coupler.

In the case of a CPU 416F without an integrated PROFINET interface, use CP 443-1 Advanced.

Note

Disable the "Data validity indicator DIA" (same as default setting) in the object properties for the PN/PN coupler in *HW Config*. Otherwise, safety-related IO-controller-IO-controller communication is not possible.

Reference

In addition, the information on safety-related master-master communication in Chapter "Safety-Related Master-Master Communication" also applies analogously.

8.7 Safety-Related Communication via S7 Connections

8.7.1 Configuring safety-related communication using S7 connections

Introduction

Safety-related communication between the safety programs of F-CPUs via S7 connections takes place by means of connection tables in *NetPro* - same as in standard programs.

Restrictions

Note

In S7 Distributed Safety, S7 connections are generally permitted over Industrial Ethernet only!

Safety-related communication via S7 connections is possible from and to the following CPUs:

- CPU 315F-2 PN/DP (only via the CPU PN interface)
- CPU 317F-2 PN/DP (only via the CPU PN interface)
- CPU 416F-3 PN/DP (only via the CPU PN interface)
- CPU 416F-2 **firmware version V4.0 and higher**

Creating an S7 Connection in the Connection Table

For each connection between two F-CPUs, you must create an S7 connection in the connection table in *NetPro*.

STEP 7 assigns a local ID and a partner ID for each connection end-point. If necessary, you can change the local ID in *NetPro*. You assign the local ID to the ID parameter of the appropriate F-application blocks in the safety programs.

Note

Safety-related communication via S7 connections to unspecified partners is not possible.

Procedure for Configuring S7 Connections

You configure the S7 connections for safety-related CPU-CPU communication the same was as for standard systems.

Note

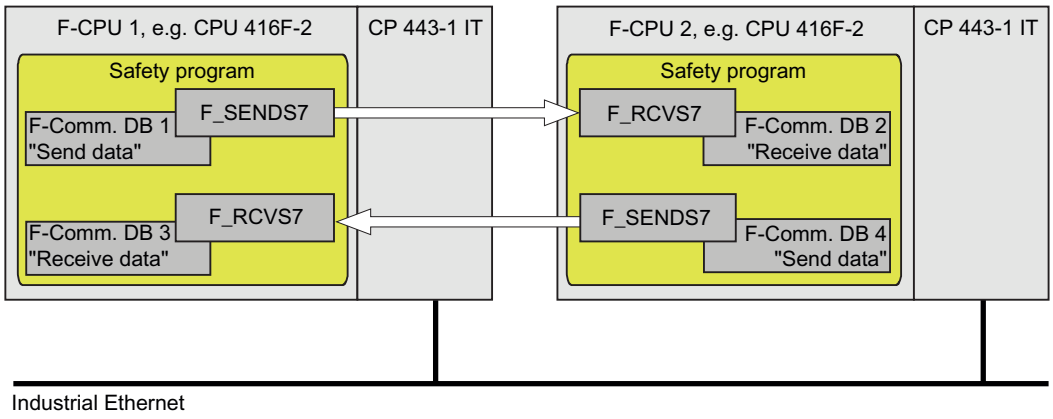
If you change the configuration of the S7 connections for safety-related communication, the collective signature of the safety program is set to "0". You must then recompile the safety program.

Additional Information

For a description of configuring S7 connections, refer to the *Configuring Hardware and Communication Connections with STEP 7 V5.x* and the *STEP 7 Online Help*.

8.7.2 Communication via F_SENDS7, F_RCVS7, and F-Communication DB

Communication by Means of F_SENDS7 and F_RCVS7



You use the **F_SENDS7 and F_RCVS7** F-application blocks for fail-safe sending and receiving data via S7 connections.

These F-application blocks can be used to transmit a specified amount of fail-safe data of data types BOOL, INT, WORD, and TIME in a fail-safe manner. The fail-safe data are stored in F-DBs that you have created.

You can find these F-application blocks in the *F-application blocks* block container in the *Distributed Safety* F-library (V1). The F_RCVS7 **must** be called at the start of the F-PB. The F_SENDS7 **must** be called at the end of the F-PB.

Note that the send signals are sent only after the F_SENDS7 call at the end of the relevant F-runtime group execution.

For a detailed description of the F-application blocks, refer to Chapter "FB 225 "F_SENDS7", FB 226 "F_RCVS7": Communication via S7 Connections".

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).

The F-communication DB numbers are made available to the F_SENDS7 or F_RCVS7 as parameters.

See also

FB 225 "F_SENDS7" and FB 226 "F_RCVS7": Communication via S7 Connections (Page 235)

8.7.3 Programming Safety-Related CPU-CPU Communication via S7 Connections

Introduction

This section describes how to program safety-related communication between safety programs of the F-CPUs via S7 connections. You must do the following in the safety programs of the relevant F-CPUs:

- Create F-DBs in which send data or receive data for communication are stored
- Call and assign parameters for F-application blocks for communication from the *Distributed Safety* F-library (V1) in the safety program

Requirements for Programming

The following requirements must be met prior to programming:

- The S7 connections between the relevant F-CPUs must be configured in *NetPro*
- Both CPUs must be configured as F-CPUs:
  - "CPU contains safety program" option must be selected
  - The password for the F-CPU must be entered

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 *SIMATIC Manager*.

Note the following when creating F-communication DBs:

When creating the F-DB, assign the "COM_DBS7" identifier in the "Family" field in the "General - Part 2" tab of the object properties for the F-DB. This identifier designates the F-DB as an F-communication DB. Only F-DBs with this identifier can be transferred as F-communication DBs to F_SENDS7 or F_RCVS7. Assign a symbolic name for the F-communication DB.

Note

The length and structure of the F-communication DB on the receiver side must match the length and structure of the associated F-communication DB on the sender side.

If the F-communication DBs do not match, the F-CPU can go to STOP mode. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety Program: internal CPU fault; internal error information: 404"

For this reason, we recommend that you use the following procedure:

1. Create an F-communication DB in the block container of the offline safety program on the sender side in *SIMATIC Manager*.
2. Specify the appropriate structure of the F-communication DB, taking into account the data to be transferred.
3. Copy this F-communication DB in the block container of the offline safety program on the receiver side, and change the DB number, 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.
- Only data types BOOL, INT, WORD, and TIME are permitted to be declared in the F-communication DBs.
- Data types must be arranged block-by-block in the following order: BOOL, INT, WORD, and TIME. Only one block per data type is permitted in an F-communication DB.
- 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, *S7 Distributed Safety* outputs an error message.

Assigning Fail-Safe Values

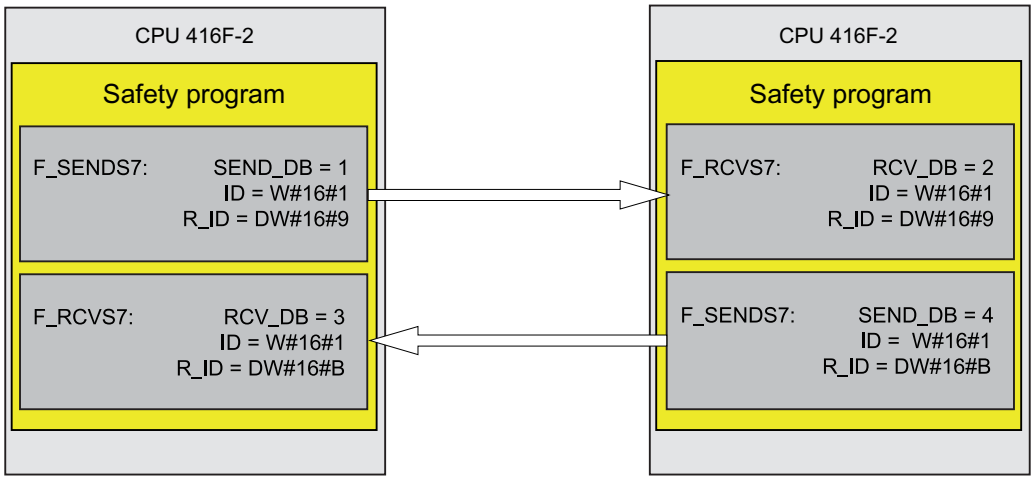
Fail-safe values are made available from the receiver side:

- 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 in the F-communication DB on the receiver side are made available as fail-safe values (default of F-communication DB).

Programming Procedure

1. Supply the variables in the F-communication DB of the sender side with send signals using symbolic, fully qualified access (e.g., "Name of F-communication DB"."variable name").
2. Read the variables in the F-communication DB of the receiver side (receive signals) that you want to process further in other sections of the program using symbolic, fully qualified access (e.g., "Name of F-communication DB"."variable name").
3. In the safety program from which data are to be sent, call the F_SENDS7 F-application block for sending at the end of the F-PB.
4. In the safety program from which data are to be received, call the F_RCVS7 F-application block for receiving at the start of the F-PB.
5. Assign the applicable F-communication DB numbers to the SEND_DB input of F_SENDS7 and the RCV_DB input of F_RCVS7.
6. Assign the local ID of the S7 connection (data type: WORD) configured in *NetPro* to the input ID of F_SENDS7.
7. Assign the local ID of the S7 connection (data type: WORD) configured in *HW Config* to the input ID of F_RCVS7.
8. Assign an odd number (data type: DWORD) to the R_ID inputs of F_SENDS7 and F_RCVS7. This specifies that an F_SENDS7 and an F_RCVS7 belong together. The related F-blocks are given the same R_ID.



WARNING

The value for each address association (input parameter R_ID; data type: DWORD) is user-defined; however, it must be unique from all other safety-related communication connections in the network. The value R_ID + 1 is assigned internally and must not be used.

Note

A separate instance DP must be used for each call of an F_SENDS7 and F_RCVS7.

The input and output parameters of the F_RCVS7 must not be supplied with local data of the F-program block.

You must not use an actual parameter for an output parameter of an F_RCVS7 if it is already being used for an input parameter of the same F_RCVS7 or another F_RCVS7 or F_RCVDP call. The F-CPU can go to STOP if this is not observed. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety program: internal CPU fault; internal error information: 404"

9. Configure the TIMEOUT inputs of the F_SENDS7 and F_RCVS7 with the required monitoring time.

⚠ WARNING

It can be ensured (from a fail-safe standpoint) that a signal level to be transferred will be captured on the sender side and transferred to the receiver only if the signal is pending for at least as long as the assigned monitoring time (TIMEOUT).
For information on calculating the monitoring times, refer to the *Safety Engineering in SIMATIC S7* system manual.

10. To reduce the bus load, you can temporarily shut down communication between the F-CPUs. To do so, supply input EN_SEND of F_SENDS7 with "0" (default = "1"). Then, send data are no longer sent to the F-communication DB of the associated F_RCVS7 and the receiver F_RCVS7 provides fail-safe values for this period (default F-communication DB). If communication was already established between the partners, a communication error is detected.
11. Optional: Evaluate the ACK_REQ output of the F_RCVS7, for example, in the standard user program or on the operator control and monitoring system in order to query or to indicate whether user acknowledgment is required.
12. Provide the ACK_REI input of the F_RCVS7 with the signal for the acknowledgment for reintegration.
13. Optional: Evaluate output SUBS_ON of F_RCVS7 or F_SENDS7 to query whether the F_RCVS7 is outputting the fail-safe values you specified as defaults in the F-communication DB.
14. Optional: Evaluate the ERROR output of the F_RCVS7 or the F_SENDS7, for example, in the standard user program or on the operator control and monitoring system in order to query or to indicate whether a communication error has occurred.
15. Optional: Evaluate the SENDMODE output of the F_RCVS7 in order to query whether the F-CPU with the associated F_SENDS7 is in deactivated safety mode.

⚠WARNING

If the F-CPU with the associated F_SENDS7 is in deactivated safety mode, you can no longer assume that the data received from this F-CPU were 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 received data. Alternatively, you must output fail-safe values instead of the received data in the F-CPU with the F_RCVS7 by evaluating SENDMODE.

See also

Creating and Editing F-DB (Page 79)

8.7.4 Limits for Data Transfer (Safety-Related Communication via S7 Connections)

Limits for Data Transfer

Note

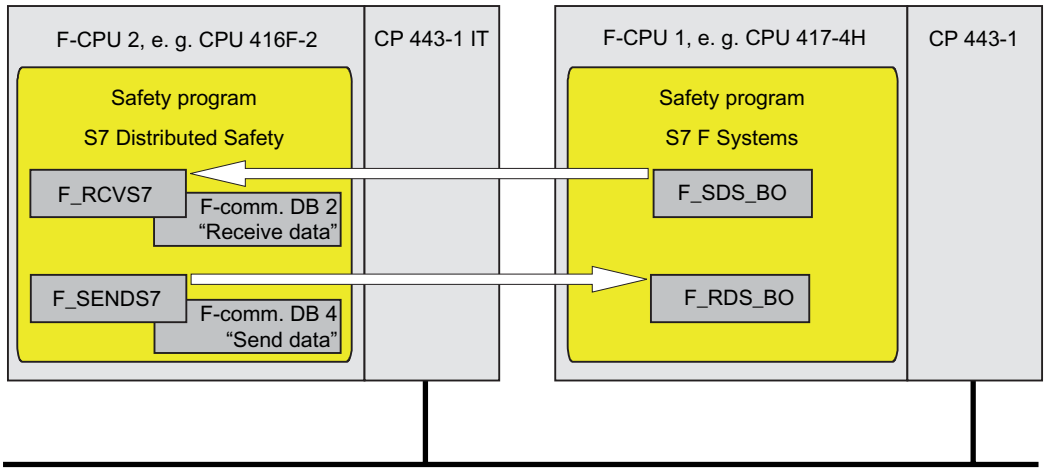
If the amount of data to be transmitted exceeds the permissible length for the F-communication DB (100 bytes), you can create another F-communication DB that you transfer to an additional F_SENDS7/F_RCVS7 call with modified R_ID.

Note that SFB 8 and SFB 9 are called internally at each F_SENDS7 call or F_RCVS7 call and use connection resources in the F-CPU. This affects the maximum number of communication connections available. Information about the connection resources of an F-CPU is obtained in the same way as for standard systems in the "Module Information" dialog of the "Communication" tab.

8.8 Safety-Related Communication between S7 Distributed Safety and S7 F System

Introduction

Safety-related communication via S7 connections for F-CPUs in S7 F Systems is also possible. A maximum of 32 data elements of data type BOOL can be exchanged.



Procedure on the *S7 Distributed Safety* side

On the *S7 Distributed Safety* side, proceed as described in Chapter "Safety-Related Communication via S7 Communications".

Particularity:

For communication between *S7 F Systems* and *S7 Distributed Safety*, you must create the F-communication DB with exactly 32 data elements of data type BOOL on the *S7 Distributed Safety* side.

Procedure on the *S7 F Systems* side

On the *S7 F Systems* side, proceed as described in Chapter "Safety-Related Communication between F-CPUs" in the "S7 F/FH Systems - Configuring and Programming" manual (http://support.automation.siemens.com/WW/view/de/16537972).

Particularity:

Communication between *S7 F Systems* and *S7 Distributed Safety* is only possible on the *S7 F Systems* side with the F-Blocks F_SDS_BO/F_RDS_BO.

F-Libraries

9

9.1 Distributed Safety F-library (V1)

9.1.1 Overview of Distributed Safety F-Library (V1)

Overview

The *Distributed Safety* F-library (V1) contains:

- F-application blocks in the *F-Application Blocks*\|*Blocks* block container
- F-system blocks and the F-shared DB in the *F-System Blocks*\|*Blocks* block container

---

Note

You must not change the F-library name.

The *Distributed Safety* F-library (V1) can contain only those F-blocks that were installed with the *S7 Distributed Safety* version.

9.1.2 F-Application Blocks

9.1.2.1 Overview of F-application blocks

Overview of F-Application Blocks

Block number	Block Name	Function
FB179 (Page 179)	F_SCA_I	Scale Values of Data Type INT
FB181 (Page 180)	F_CTU	Count up
FB182 (Page 181)	F_CTD	Count down
FB183 (Page 182)	F_CTUD	Count up and down
FB184 (Page 183)	F_TP	Create pulse
FB185 (Page 185)	F_TON	Create ON-delay
FB186 (Page 187)	F_TOF	Create OFF-delay
FB187 (Page 189)	F_ACK_OP	Fail-safe acknowledgment
FB188 (Page 191)	F_2HAND	Two-hand monitoring
FB189 (Page 193)	F_MUTING	Muting
FB 190 (Page 201)	F_1oo2DI	1oo2 evaluation with discrepancy analysis
FB 211 (Page 205)	F_2H_EN	Two-hand monitoring with enable
FB 212 (Page 207)	F_MUT_P	Parallel Muting
FB 215 (Page 217)	F_ESTOP1	Emergency STOP up to Stop Category 1
FB 216 (Page 220)	F_FDBACK	Feedback monitoring
FB 217 (Page 224)	F_SFDOOR	Safety door monitoring
FB 219 (Page 228)	F_ACK_GL	Global acknowledgment of all F-I/Os in an F-Runtime group
FB223 (Page 229)	F_SENDDP	Send data (16 BOOL, 2 INT) via PROFIBUS DP
FB224 (Page 229)	F_RCVDP	Receive data (16 BOOL, 2 INT) via PROFIBUS DP
FB 225 (Page 235)	F_SENDS7	For CPUs 4xxF: Send data (from F-DB) via S7 connections
FB 226 (Page 235)	F_RCVS7	For CPUs 4xxF: Receive data (from F-DB) via S7 connections
FC 174 (Page 243)	F_SHL_W	Shift Left 16 Bits
FC 175 (Page 244)	F_SHR_W	Shift Right 16 Bits
FC 176 (Page 245)	F_BO_W	Convert 16 Data Elements of Data Type BOOL to a Data Element of Data Type WORD
FC 177 (Page 245)	F_W_BO	Convert a Data Element of Data Type WORD to 16 Data Elements of Data Type BOOL
FC 178 (Page 246)	F_INT_WR	Write value of data type INT indirectly to an F-DB
FC 179 (Page 248)	F_INT_RD	Read value of data type INT indirectly from an F-DB

Note

You may change the numbers of the F-application blocks. Exception: When using the F_ESTOP1 and F_FDBACK F-application blocks, the F_TOF F-application block must have number FB 186 and must not be renumbered.

If you change the numbers for an F-application block, note that the symbolic name in the symbol table must continue to match the name in the object properties for the block (header).

You cannot use symbolic names of F-application blocks of the *Distributed Safety F-library* (V1) for user-created F-FBs, F-FCs, and blocks.

Note

You must ensure that the F-blocks in the F-CPU are consistent.

To do so, you must use F-application blocks **of a single** *S7 Distribution Safety* version only and compile the safety program using the *S7 Distributed Safety* setup.

Note

If you call a block, the enable input EN and the enable output ENO appear automatically. You must not interconnect these connections, supply them with "0", or evaluate them.

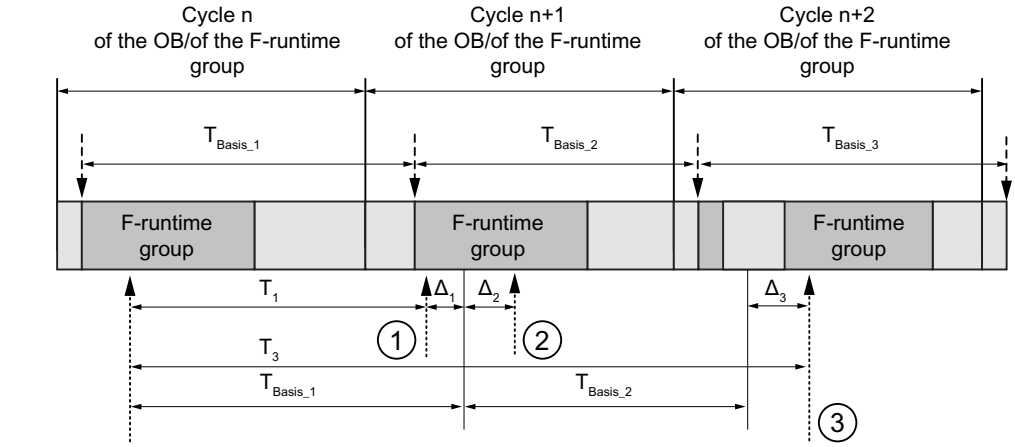
Timing Imprecision for F-Application Blocks with Time Processing

⚠WARNING

When using an F-application block 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 F-application block (see figure below)
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value
- You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

Timing Imprecision Resulting from the Update Time of the Time Base Used in the F-Application Block



⇢ = Update of the time base

⇢ = Call times of an F-application block with time processing

Description

(1) For the first call in cycle n+1, the call time of the F-application block relative to the start of the F-runtime group is earlier than that in cycle n by the amount of Δ_1, e.g., because portions of the safety program of the F-runtime group before the call time of the F-application in cycle n+1 are skipped. For the time update, the F-application block takes into account time T_{Base_1} instead of the time T_1 that has actually elapsed in cycle n since the call.

(2) The F-application block is called a second time in cycle n+1. This does not involve another time update (by Δ_2).

(3) For the call in cycle n+2, the call time of the F-application block relative to the start of the F-runtime group is later than that in cycle n by the amount of Δ_3, e.g., because the F-runtime group was interrupted by a higher priority interrupt prior to the time of the F-application block call in cycle n+2. The F-application block 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.

9.1.2.2 FB 179 "F_SCA_I": Scale Values of Data Type INT

Inputs/Outputs

	Parameter	Data Type	Description	Default
Inputs:	IN	INT	Input value to be scaled in physical units	0
	HI_LIM	INT	Upper limit value in physical units	0
	LO_LIM	INT	Lower limit value in physical units	0
Outputs:	OUT	INT	Result of scaling	0
	OUT_HI	BOOL	1 = Input value > 27648: OUT = HI_LIM	0
	OUT_LO	BOOL	1 = Input value < 0: OUT = LO_LIM	0

Principle of operation

This F-application block scales the value at input IN in physical units between the lower limit value at input LO_LIM and the upper limit value at input HI_LIM. It is assumed that the value at input IN is between 0 and 27,648. The scaling result is provided at output OUT.

The F-application block acts according to the following equation:

OUT = [IN * (HI_LIM - LO_LIM)] / 27648 + LO_LIM

So long as the value at input IN is greater than 27,648, output OUT is linked to HI_LIM, and OUT_HI is set to 1.

So 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 reverse scaling, you must assign LO_LIM > HI_LIM. With reverse scaling, the output value at output OUT decreases while the input value at input IN increases.

Performance 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 13 bit are used as input values, you must bear in mind 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 variable 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 greater than 27648 or less than 0, you can likewise branch to the output of an individual fail-safe value by evaluating outputs OUT_HI and OUT_LO, respectively.

9.1.2.3 FB 181 "F_CTU": Count Up

Connections

	Parameter	Data Type	Description	Default
Inputs:	CU	BOOL	Counter input	0
	R	BOOL	Reset input (R prevails over CU)	0
	PV	INT	Default value, see parameter Q for effect of PV	0
Outputs:	Q	BOOL	Counter status: Q = 1, if CV >= PV Q = 0, if CV < PV	0
	CV	INT	Current counter value (possible values: 0 to 32767)	0

Principle of operation

This F-application block forms an edge-controlled up-counter (with functionality based on IEC counter SFB 0 "CTU").

The counter counts up 1 on a rising edge (relative to the last F-application block call) at input CU.

When the counter value reaches the upper limit of 32,767, it no longer counts up. For every additional rising edge at input CU, no counter action takes place.

Signal state 1 at input R causes the counter to be reset to 0, irrespective of the value at input CU. Output Q displays whether the current counter value is greater than or equal to the default value PV.

The functionality of this F-application block is in accordance with IEC 61131-3.

Startup Characteristics

Following an F-system startup, the instances of the F_CTU are reset, resulting in:

- CV = 0
- Q = 0

9.1.2.4 FB 182 "F_CTD": Count Down

Inputs/Outputs

	Parameter	Data Type	Description	Default
Inputs:	CD	BOOL	Counter input	0
	LOAD	BOOL	Load input, LOAD prevails over CD	0
	PV	INT	Default value; the counter is preset to PV, if the signal state 1 is present at input LOAD.	0
Outputs:	Q	BOOL	Counter status: Q = 1, if CV <= 0 Q = 0, if CV > 0	0
	CV	INT	Current counter value (possible values: -32768 to 32767)	0

Principle of operation

This F-application block forms an edge-controlled down-counter (with functionality based on IEC counter SFB 1 "CTD").

The counter counts down 1 at a rising edge (relative to the last F-application block call) at input CD.

When the counter value reaches the lower limit of -32,768, it no longer counts down. For every additional rising edge at input CD, no counter action takes place.

Signal state 1 at input LOAD causes the counter to be preset to preset value PV. This occurs irrespective of the value at input CD. Output Q displays whether the current counter value is less than or equal to zero.

The functionality of this F-application block is in accordance with IEC 61131-3.

Startup Characteristics

The instances of F_CTD are reset in the first cycle following startup of the F-system, resulting in:

- CV = 0
- Q = 0

9.1.2.5 FB 183 "F_CTUD": Count Up and Down

Connections

	Parameter	Data Type	Description	Default
Inputs:	CU	BOOL	Count up input	0
	CD	BOOL	Count down input	0
	R	BOOL	Reset input, R prevails over LOAD	0
	LOAD	BOOL	Load input, LOAD prevails over CU and CD	0
	PV	INT	Default value; the counter is preset to PV, if signal state 1 is present at input LOAD.	0
Outputs:	QU	BOOL	Status of up-counter: QU = 1, if CV >= PV QU = 0, if CV < PV	0
	QD	BOOL	Status of down-counter: QD = 1, if CV <= 0 QD = 0, if CV > 0	0
	CV	INT	Current counter value (possible values: -32768 to 32767)	0

Principle of operation

This F-application block forms an edge-controlled up/down-counter (with functionality based on IEC counter SFB 2 "CTUD").

At a rising edge (relative to the last F-application block call), the counter behaves as follows:

- Counter counts up 1 at input CU

  When the counter value reaches the upper limit (32,767), it no longer counts up.

- Counter counts down 1 at input CD

  When the counter value reaches the lower limit (-32,768), it no longer counts down.

If there is a rising edge at both input CU and input CD during one cycle, the counter remains at its current value.

⚠WARNING

When the CU signal and the CD signal are present simultaneously, performance deviates from that prescribed in IEC 61131-3. According to the standard, the CU input prevails when the CU signal and the CD signal are present simultaneously.

Load = 1: CV is preset with the value of the PV input. The values at inputs CU and CD are ignored.

R = 1: CV is reset to 0. The values at inputs CU, CD, and LOAD are ignored.

Output QU displays whether the current counter value is greater than or equal to the preset value PV. Output QD displays whether the current counter value is less than or equal to zero.

Startup Characteristics

The instances of the F_CTUD are reset in the first cycle following a startup of the F-system, resulting in:

- CV = 0
- QU = 0
- QD = 0

9.1.2.6 FB 184 "F_TP": Create Pulse

Connections

	Parameter	Data Type	Description	Default
Inputs:	IN	BOOL	Start input	0
	PT	TIME	Pulse duration, with PT >= 0	T# 0 ms
Outputs:	Q	BOOL	Time status	0
	ET	TIME	Elapsed time	T# 0 ms

Principle of operation

This F-application block generates a pulse of length PT at output Q (this functionality is based on IEC TIMER SFB 3 "TP").

The pulse is initiated on a rising edge at input IN. Output Q remains set for duration PT, irrespective of any further variation of the input signal (that is, even if input IN switches from 0 back to 1 before time PT has elapsed).

Output ET displays how long output Q has already been set. It can have a maximum value equal to the value of input PT. It is reset when input IN changes to 0, however, time PT must elapse before it can be reset.

⚠WARNING

When using an F-application block 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 F-application block (see figure in "F-Application Blocks")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

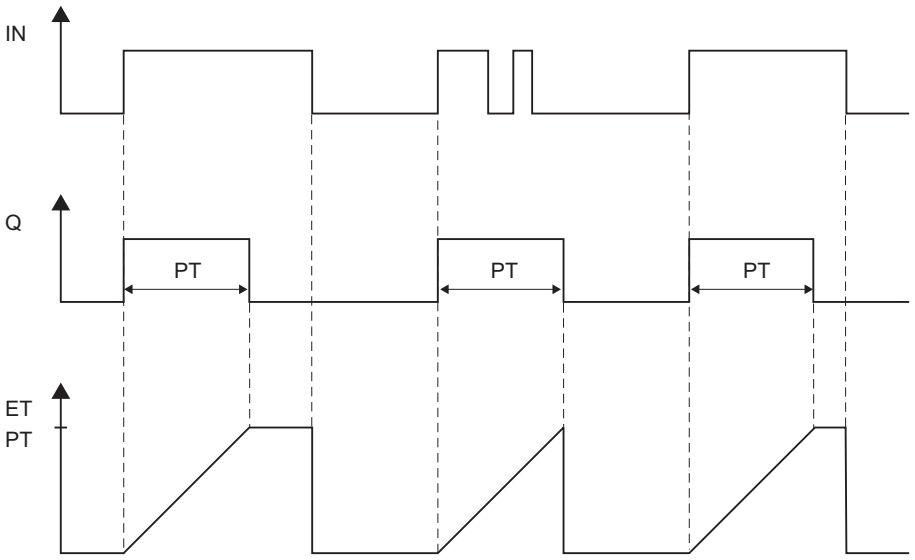
You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

⚠WARNING

The functionality of this F-application block complies with IEC 61131-3, however, it deviates from IEC TIMER SFB 3 "TP" as follows:

- When it is called with PT = 0 ms, the F_TP instance is not reset completely (initialized). The block behaves in accordance with the timing diagrams: only outputs Q and ET are reset. Another rising edge at input IN is required to restart the pulse, once PT is greater than 0 again.
- A call with PT < 0 ms resets outputs Q and ET. Another rising edge at input IN is required to restart the pulse, once PT is greater than 0 again.

F_TP Timing Diagrams



Startup Characteristics

The instances of F_TP are reset in the first cycle following a startup of the F-system, resulting in:

- ET = 0
- Q = 0

See also

Overview of F-application blocks (Page 176)

9.1.2.7 FB 185 "F_TON": Create ON Delay

Connections

	Parameter	Data Type	Description	Default
Inputs:	IN	BOOL	Start input	0
	PT	TIME	Time by which the rising edge at input IN is delayed, with PT >= 0	T# 0 ms
Outputs:	Q	BOOL	Time status	0
	ET	TIME	Elapsed time	T# 0 ms

Principle of operation

This F-application block delays a rising edge by time PT (this functionality is based on IEC TIMER SFB 4 "TON").

A rising edge at input IN results in a rising edge at output Q once time PT has elapsed. Q remains set until input IN changes to 0.

If input IN changes to 0 before time PT has elapsed, then output Q remains at 0.

Output ET supplies the time that has passed since the last rising edge at input IN, not to exceed the value at input PT. ET is reset if input IN changes to 0.

⚠ WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in the "F-Application Blocks" section)
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

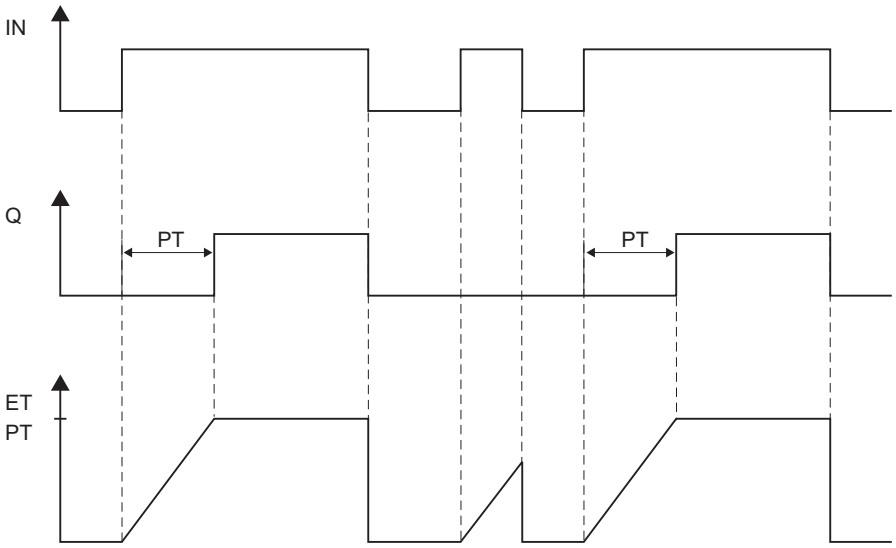
You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

⚠ WARNING

The functionality of this F-application block complies with IEC 61131-3, however, it deviates from IEC TIMER SFB 4 "TON" as follows:

- When it is called with PT = 0 ms, the F_TON instance is not reset completely (initialized). The block behaves in accordance with the timing diagrams: Only output ET is reset. Another rising edge at input IN is required to restart the ON delay, once PT is greater than 0 again.
- A call with PT < 0 ms resets outputs Q and ET. Another rising edge at input IN is required to restart the ON delay, once PT is greater than 0 again.

F_TON Timing Diagrams



Startup Characteristics

The instances of F_TON are reset in the first cycle following a startup of the F-system, resulting in:

- ET = 0
- Q = 0

See also

Overview of F-application blocks (Page 176)

9.1.2.8 FB 186 "F_TOF": Create OFF Delay

Connections

	Parameter	Data Type	Description	Default
Inputs:	IN	BOOL	Start input	0
	PT	TIME	Time by which the falling edge at input IN is delayed, with PT >= 0	T# 0 ms
Outputs:	Q	BOOL	Time status	0
	ET	TIME	Elapsed time	T# 0 ms

Principle of operation

This F-application block delays a falling edge by time PT (this functionality is based on IEC TIMER SFB 5 "TOF").

A rising edge at input IN causes a rising edge at output Q. A falling edge at input IN results in a falling edge at output Q once time PT has elapsed.

If input IN changes back to 1 before time PT has elapsed, then output Q remains at 1.

Output ET supplies the time that has passed since the last falling edge at input IN, not to exceed the value at input PT. ET is reset if input IN changes to 1.

⚠WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in the "F-Application Blocks" section)
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

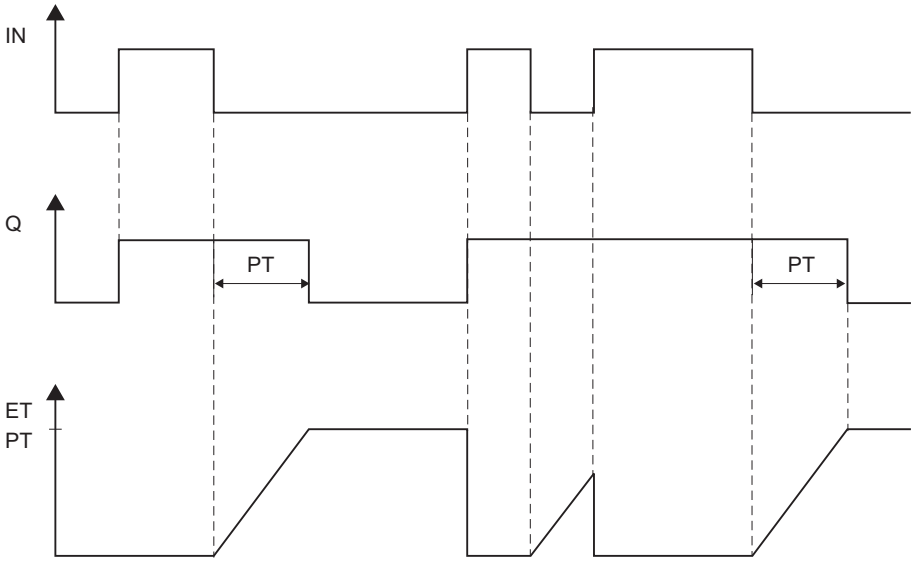
You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

⚠ WARNING

The functionality of this F-application block complies with IEC 61131-3, however, it deviates from IEC TIMER SFB 5 "TOF" as follows:

- When it is called with PT = 0 ms, the F_TOF instance is not reset completely (initialized). The block behaves in accordance with the timing diagrams: only outputs Q and ET are reset. Another falling edge at input IN is required to restart the OFF delay, once PT is greater than 0 again.
- A call with PT < 0 ms resets outputs Q and ET. Another falling edge at input IN is required to restart the OFF delay, once PT is greater than 0 again.

F_TOF Timing Diagrams



Startup Characteristics

The instances of F_TOF are reset in the first cycle following a startup of the F-system, resulting in:

- ET = 0
- Q = 0

See also

Overview of F-application blocks (Page 176)

9.1.2.9 FB 187 "F_ACK_OP": Fail-Safe Acknowledgment

Connections

	Parameter	Data Type	Description	Default
In/Out Parameters:	IN	INT	Input variable from operator control and monitoring system	0
Outputs:	OUT	BOOL	Output for acknowledgment	0
	Q	BOOL	Time status	0

Principle of operation

This F-application block enables fail-safe acknowledgment from an operator control and monitoring system. It allows, for example, reintegration of F-I/O to be controlled from the operator control and monitoring system. Acknowledgment takes place in two steps:

1. In/out parameter IN changes to a value of 6.
2. In/out parameter IN changes to a value of 9 within 1 min.

Once the in/out parameter IN has changed to a value of 6, the F-application block evaluates whether this parameter has changed to a value of 9 after 1 s, at the earliest, or 1 min, 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 9 within 1 min or the change occurred before 1 s 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 9, output Q is set to 1. Otherwise, Q has a value of 0.

⚠ WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in the "F-Application Blocks" section)
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

Note

You can evaluate output Q only in your standard user program. Access to output Q in the safety program is not permissible.

You can supply in/out parameter IN with just a memory word or nothing at all. In the safety program, read and write access to in/out parameter IN in the associated instance DB is not permitted!

Note

A separate instance DB must be used for each call of F_ACK_OP. Each call can be processed only once in an F-run-time group cycle.

The F-CPU can go to STOP mode if the information above is not observed. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety program: internal CPU fault; internal error information: 404"

Additional Information

You will find additional information about fail-safe acknowledgment with the F_ACK_OP F-application block in the references provided under "See also."

See also

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Implementing User Acknowledgment in the Safety Program of a I-Slave F-CPU (Page 120)

Overview of F-application blocks (Page 176)

9.1.2.10 FB 188 "F_2HAND": Two-Hand Monitoring

Connections

	Parameter	Data Type	Description	Default
Inputs:	IN1	BOOL	Momentary-contact switch 1	0
	IN2	BOOL	Momentary-contact switch 2	0
	DISCTIME	TIME	Discrepancy time (0 to 500 ms)	T# 0 ms
Outputs:	Q	BOOL	1=Enable	0

Principle of operation

This F-application block implements two-hand monitoring. If momentary-contact switches IN1 and IN2 are activated within the permissible discrepancy time DISCTIME ≤ 500 ms (IN1/IN2 = 1) (synchronous activation), output signal Q is set to 1. If the time difference between activation of momentary-contact switch IN1 and momentary-contact switch IN2 is greater than DISCTIME, then the momentary-contact switches must be released and reactivated.

Q is reset to 0 as soon as one of the momentary-contact switches is released (IN1/IN2 = 0). Enable signal Q can be reset to 1 only if the other momentary-contact switch has been released, and if both switches 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.

The F-application block supports requirements in accordance with EN 574.

Note: Only one signal per momentary-contact switch can be evaluated in the F-application block. With suitable configuration (type of sensor interconnection: 2-channel nonequivalent), discrepancy monitoring of the NC and NO contacts of the IN1 and IN2 momentary-contact switches is performed directly by the F-I/O with inputs. The NO 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 "0 - provide value" for the behavior of discrepancy during configuration. If a discrepancy is detected, a fail-safe value of 0 is entered in the process input image (PII) for the momentary-contact switch and QBAD or QBAD_I_xx = 1 is set in the relevant F-I/O DB.

⚠ WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in the "F-Application Blocks" section)
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

Additional Information

You will find additional information about the configuration and the F-I/O DB in the references provided under "See also."

See also

Overview of Configuration (Page 23)

F-I/O DB (Page 98)

Overview of F-application blocks (Page 176)

9.1.2.11 FB 189 "F_MUTING": Muting

Connections

	Parameter	Data Type	Description	Default
Inputs:	MS_11	BOOL	Muting sensor 1 of sensor pair 1	0
	MS_12	BOOL	Muting sensor 2 of sensor pair 1	0
	MS_21	BOOL	Muting sensor 1 of sensor pair 2	0
	MS_22	BOOL	Muting sensor 2 of sensor pair 2	0
	STOP	BOOL	1=Conveyor system stopped	0
	FREE	BOOL	1=Light curtain uninterrupted	0
	QBAD_MUT	BOOL	QBAD or QBAD_O_xx signal of F-I/O/channel of muting lamp (F-I/O DB)	0
	DISCTIM1	TIME	Discrepancy time of sensor pair 1 (0 to 3 s)	T# 0 ms
	DISCTIM2	TIME	Discrepancy time of sensor pair 2 (0 to 3 s)	T# 0 ms
	TIME_MAX	TIME	Maximum muting time (0 to 10 min)	T# 0 M
	ACK	BOOL	Acknowledgment of restart inhibit	0
Outputs:	Q	BOOL	1= Enable, not off	0
	MUTING	BOOL	Display of muting is active	0
	ACK_REQ	BOOL	Acknowledgment necessary	0
	FAULT	BOOL	Group error	0
	DIAG	BYTE	Service information	0

Principle of Operation

This F-application block 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.

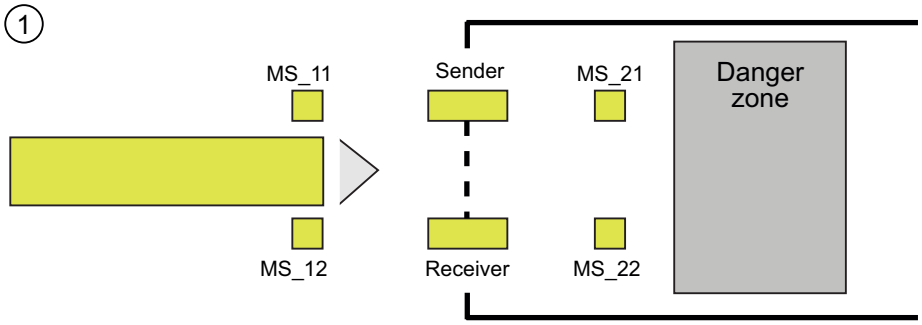
WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in Chapter "F-Application Blocks")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

Schematic Sequence of Error-Free Muting Procedure with Four Muting Sensors (MS_11, MS_12, MS_21, MS_22)



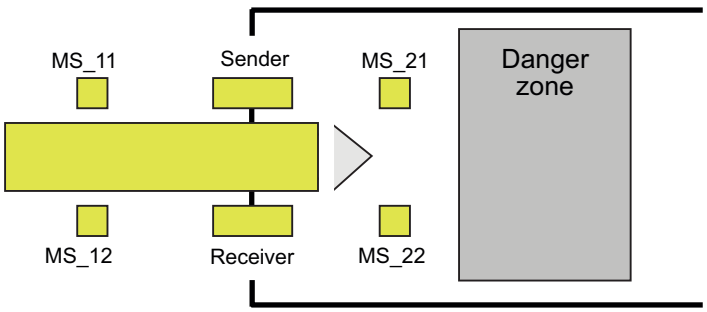
- If both muting sensors MS_11 and MS_12 are activated by the product within DISCTIM1 (apply signal state = 1), the F-application block starts the MUTING function. Enable signal Q remains 1, even when input FREE = 0 (light curtain interrupted by product). The MUTING output for setting the muting lamp 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 or QBAD_O_xx signal of the associated F-I/O or channel. If QBAD_MUT = 1, muting is terminated by the F-application block. If monitoring of the muting lamp is not necessary, you do not have to supply input QBAD_MUT.

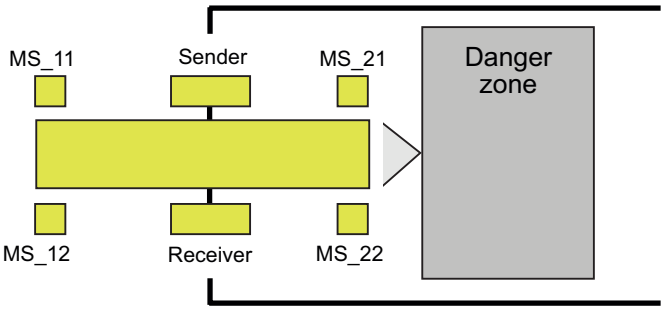
F-I/O that can promptly detect a wire break after activation of the muting operation must be used (*see manual for specific F-I/O*).

②



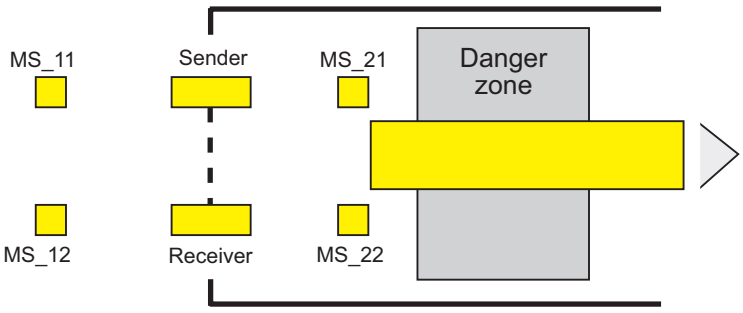
- As long as both muting sensors MS_11 and MS_12 continue to be activated, the MUTING function of the F-application block causes Q to remain 1 and MUTING to remain 1 (so that the product can pass through the light curtain without causing the machine to stop).

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 F-application block retains the MUTING function. (Q = 1, MUTING = 1).

④

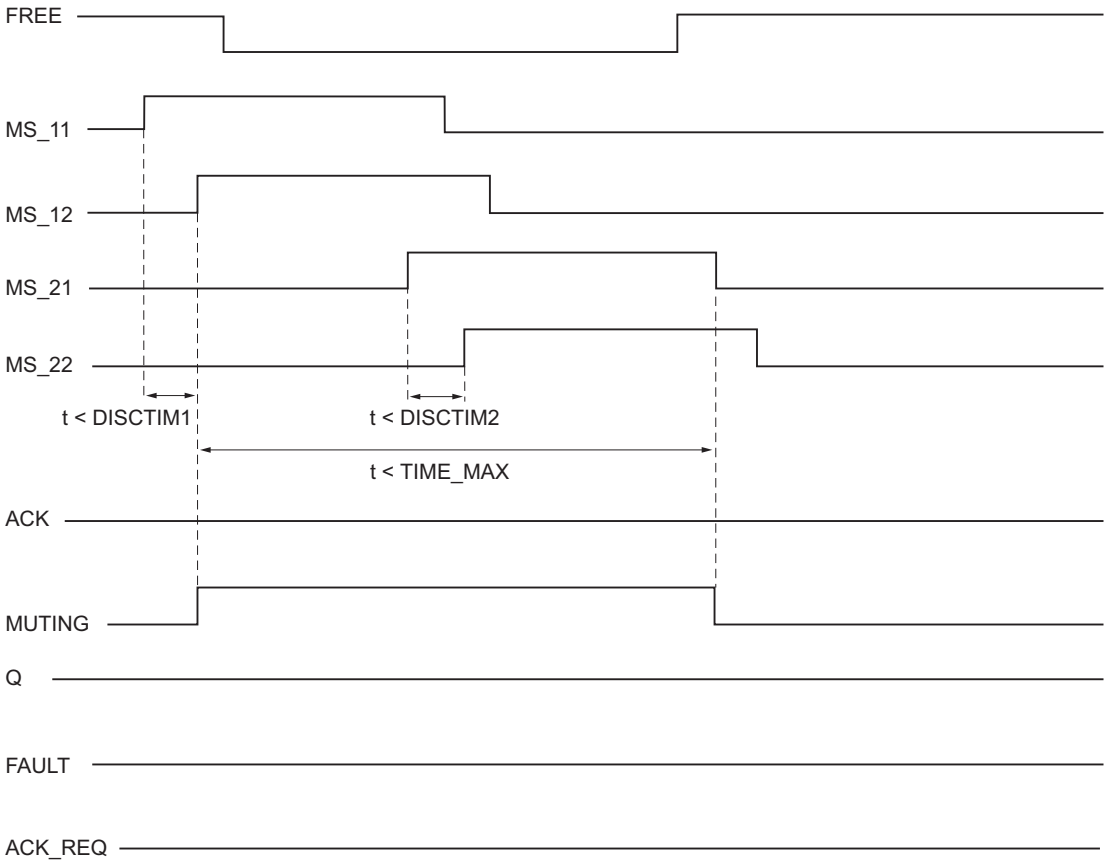


- Only if one of the two muting sensors MS_21 and MS_22 is switched to inactive (product enables sensors) is the MUTING function terminated (Q = 1, MUTING = 0). The maximum activation time for the MUTING function is the time set at input TIME_MAX.

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 Four Muting Sensors

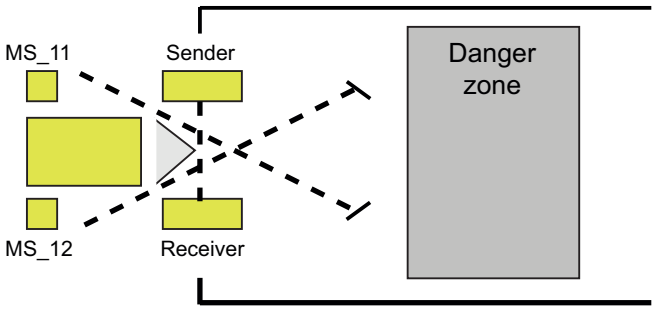


Schematic Sequence of Muting Procedure with Reflection Light Barriers

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 four multiple 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!).

Acknowledgment of Restart Inhibit

Enable signal Q becomes 1 again, if:

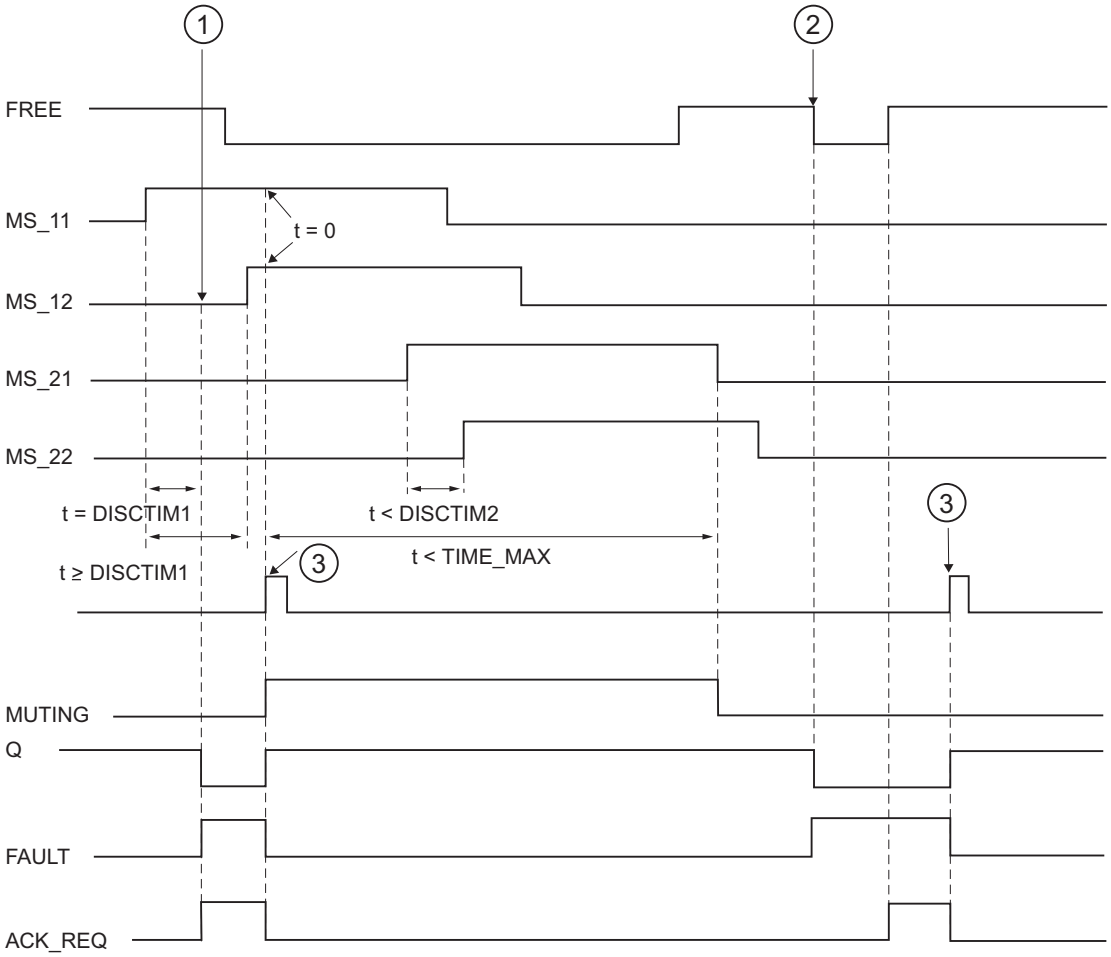
- The light curtain is no longer interrupted
- Errors, if present, are eliminated (see output DIAG)
  and
- A user acknowledgement with a positive edge is issued at input ACK (see also Chapter "Implementing User Acknowledgment").

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 block sets ACK-REQ = 1 as soon as the light curtain is no longer interrupted or errors have been eliminated. Once acknowledgment has occurred, the block 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 monitoring is deactivated while the conveyor equipment has stopped for one of the following reasons:

- 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 assume a signal state of 1 due to an unknown error, e.g., because both muting sensors fail to 1, the error is not detected and the MUTING function can be started unintentionally.

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 Causes of Problems	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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
		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 No.	Assignment	Possible Causes of Problems	Remedies
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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
		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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
Bit 5	Reserved	-	-
Bit 6	Reserved	-	-
Bit 7	Reserved	-	-

Note

Access to the DIAG output is not permitted in the safety program!

See also

F-I/O DB (Page 98)

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Implementing User Acknowledgment in the Safety Program of a I-Slave F-CPU (Page 120)

Overview of F-application blocks (Page 176)

9.1.2.12 FB 190 "F_1oo2DI": 1oo2 Evaluation with Discrepancy Analysis

Connections

	Parameter	Data Type	Description	Default
Inputs:	IN1	BOOL	Sensor 1	0
	IN2	BOOL	Sensor 2	0
	DISCTIME	TIME	Discrepancy time (0 to 60 s)	T# 0 ms
	ACK_NEC	BOOL	1 = acknowledgment necessary for discrepancy error	1
	ACK	BOOL	Acknowledgment of discrepancy error	0
Outputs:	Q	BOOL	Output	0
	ACK_REQ	BOOL	1 = acknowledgement required	0
	DISC_FLT	BOOL	1 = discrepancy error	0
	DIAG	BYTE	Service information	0

Principle of Operation

This F-application block 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.

ACK_REQ = 1 signals that a user acknowledgment at input ACK is necessary to acknowledge the discrepancy error (cancel the restart inhibit). The F-application block 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 F-application block 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.

⚠WARNING

Variable ACK_NEC must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded.

⚠WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in Chapter "F-Application Blocks")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

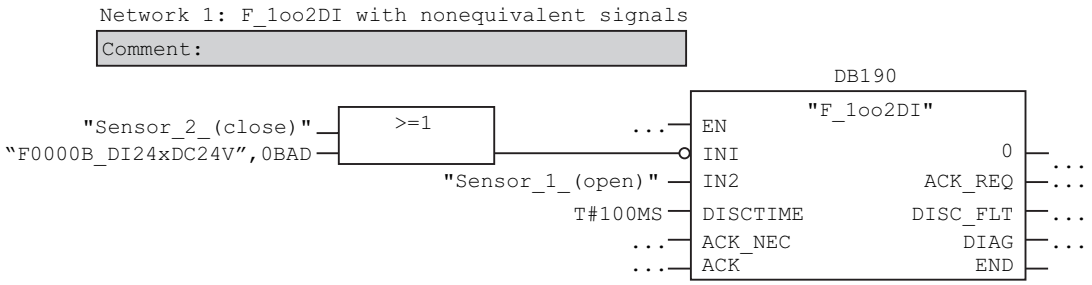
You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

Activating Inputs IN1 and IN2

Inputs IN1 and IN2 must both be activated in such a way that their safe state is 0.

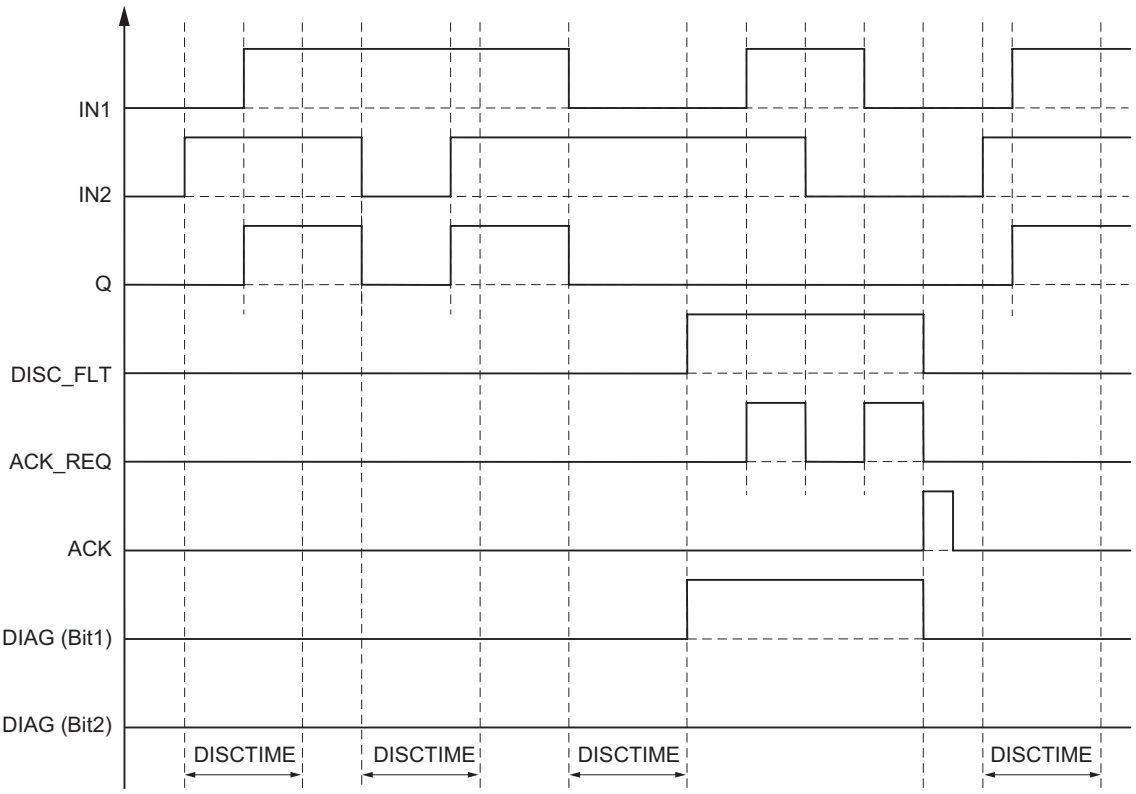
Example

For nonequivalent signals, you have to invert the input (IN1 or IN2) to which you have assigned the sensor signal with a safe state of 1. You must also OR the sensor signal with the QBAD or QBAD_I_xx variable of the associated F-I/O DB or channel, so that a signal state of 0 is present at input IN1 or IN2 (after inversion) if fail-safe values are output.



Timing Diagrams for F_1oo2DI

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 Causes of Problems	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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
		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 from 0 s to 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	-	-

Note

Access to the DIAG output is not permitted in the safety program!

See also

F-I/O DB (Page 98)

Overview of F-application blocks (Page 176)

9.1.2.13 FB 211 "F_2H_EN": Two-Hand Monitoring with Enable

Connections

	Parameter	Data Type	Description	Default
Inputs:	IN1	BOOL	Momentary-contact switch 1	FALSE
	IN2	BOOL	Momentary-contact switch 2	FALSE
	ENABLE	BOOL	Enable input	FALSE
	DISCTIME	TIME	Discrepancy time (0 to 500 ms)	T# 0 ms
Outputs:	Q	BOOL	1=Enable	FALSE
	DIAG	BYTE	Service information	B#16#0

Principle of Operation

This F-application block implements two-hand monitoring. If momentary-contact switches IN1 and IN2 are activated within the permissible discrepancy time DISCTIME ≤ 500 ms (IN1/IN2 = 1) (synchronous activation), output signal Q is set to 1 when existing enable ENABLE = 1. If the time difference between activation of momentary-contact switch IN1 and momentary-contact switch IN2 is greater than DISCTIME, then the momentary-contact switches must be released and reactivated.

Q is reset to 0 as soon as one of the momentary-contact switches is released (IN1/IN 2 = 0) or ENABLE = 0. Enable signal Q can be reset to 1 only if the other momentary-contact switch has been released, and if both switches are then reactivated within the discrepancy time when existing enable ENABLE = 1.

The F-application block supports requirements in accordance with EN 574.

Note: Only one signal per momentary-contact switch can be evaluated in the F-application block. With suitable configuration (type of sensor interconnection: 2-channel nonequivalent), discrepancy monitoring of the normally closed and normally open contacts of the IN1 and IN2 momentary contact switches is performed directly by the F-I/O with inputs. The NO 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 "Provide 0-value" for the behavior of discrepancy during configuration.

If a discrepancy is detected, a fail-safe value of 0 is entered in the process input image (PII) for the momentary-contact switch and QBAD or QBAD_I_xx = 1 is set in the relevant F-I/O DB.

⚠WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in Chapter "F-Application Blocks")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

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 Causes of Problems	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
		Momentary-contact switches were not activated within the discrepancy time	Release momentary-contact switches and activate them within the discrepancy time
		Wiring fault	Check wiring of momentary-contact switches
		Momentary-contact switches defective	Check momentary-contact switches
		Momentary-contact switches 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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
Bit 2	Reserved	-	-
Bit 3	Reserved	-	-

Bit No.	Assignment	Possible Causes of Problems	Remedies
Bit 4	Incorrect activation sequence	One momentary-contact switch was not released	Release momentary-contact switches and activate them within the discrepancy time
		Momentary-contact switches defective	Check momentary-contact switches
Bit 5	Enable ENABLE does not exist	Enable ENABLE = 0	Set ENABLE = 1, release momentary-contact switch and activate it within the discrepancy time
Bit 6	Reserved	-	-
Bit 7	State of output Q	-	-

Note

Access to the DIAG output is not permitted in the safety program!

See also

F-I/O DB (Page 98)

Overview of F-application blocks (Page 176)

9.1.2.14 FB 212 "F_MUT_P": Parallel Muting

Connections

	Parameter	Data Type	Description	Default
Inputs:	MS_11	BOOL	Muting sensor 11	0
	MS_12	BOOL	Muting sensor 12	0
	MS_21	BOOL	Muting sensor 21	0
	MS_22	BOOL	Muting sensor 22	0
	STOP	BOOL	1=Conveyor system stopped	0
	FREE	BOOL	1=Light curtain uninterrupted	0
	ENABLE	BOOL	1=Enable MUTING	0
	QBAD_MUT	BOOL	QBAD or QBAD_O_xx signal of F-I/O/channel of muting lamp (F-I/O DB)	0
	ACK	BOOL	Acknowledgment of restart inhibit	0
	DISCTIM1	TIME	Discrepancy time of sensor pair 1 (0 to 3 s)	T# 0 ms
	DISCTIM2	TIME	Discrepancy time of sensor pair 2 (0 to 3 s)	T# 0 ms
	TIME_MAX	TIME	Maximum muting time (0 to 10 min)	T# 0 ms

	Parameter	Data Type	Description	Default
Outputs:	Q	BOOL	1: Enable, not off	0
	MUTING	BOOL	Display of muting is active	0
	ACK_REQ	BOOL	Acknowledgment necessary	0
	FAULT	BOOL	Group error	0
	DIAG	WORD	Service information	W#16#0

Principle of Operation

This F-application block 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.

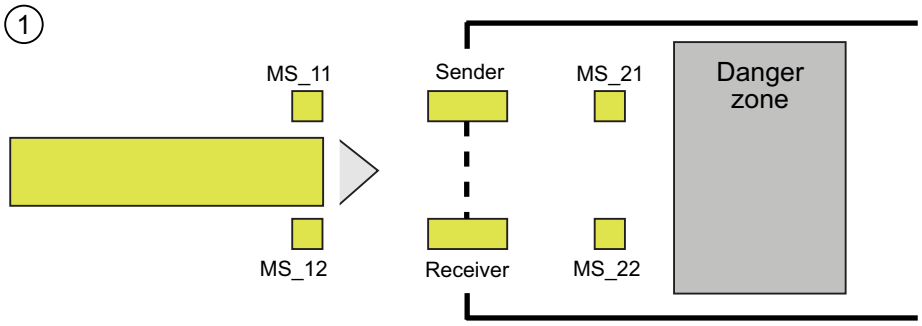
⚠WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in Chapter "F-Application Blocks")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

Schematic Sequence of Error-Free Muting Procedure with Four Muting Sensors (MS_11, MS_12, MS_21, MS_22)

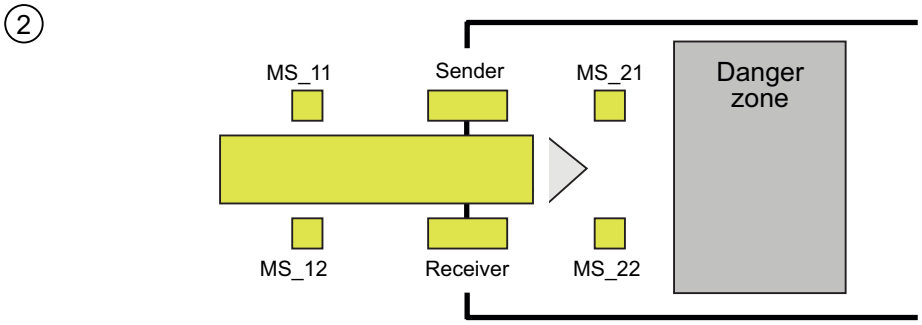


- If muting sensors MS_11 and MS_12 are both activated by the product within DISCTIM1 (apply signal state = 1) and MUTING is enabled by setting the ENABLE input to 1, the F-application block starts the MUTING function. Enable signal Q remains 1, even when input FREE = 0 (light curtain interrupted by product). The MUTING output for setting the muting lamp switches to 1.

Note

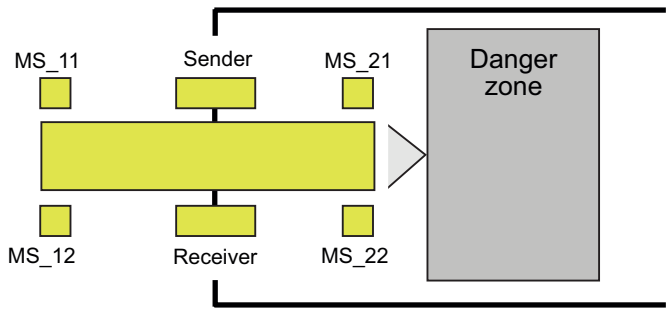
The muting lamp can be monitored using the QBAD_MUT input. To do this, you must wire the muting lamp to an output with wire break monitoring of an F-I/O and supply the QBAD_MUT input with the QBAD or QBAD_O_xx signal of the associated F-I/O or channel. If QBAD_MUT = 1, muting is terminated by the F-application block. 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*).



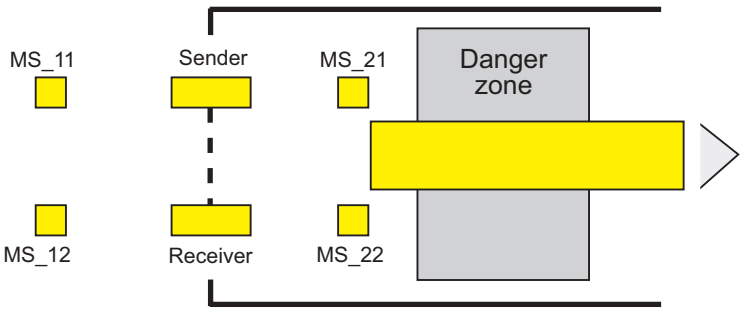
- As long as both muting sensors MS_11 and MS_12 continue to be activated, the MUTING function of the F-application block causes Q to remain 1 and MUTING to remain 1 (so that the product can pass through the light curtain without causing the machine to stop). Each of the two muting sensors MS_11 and 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 F-application block retains the MUTING function. (Q = 1, MUTING = 1).

④

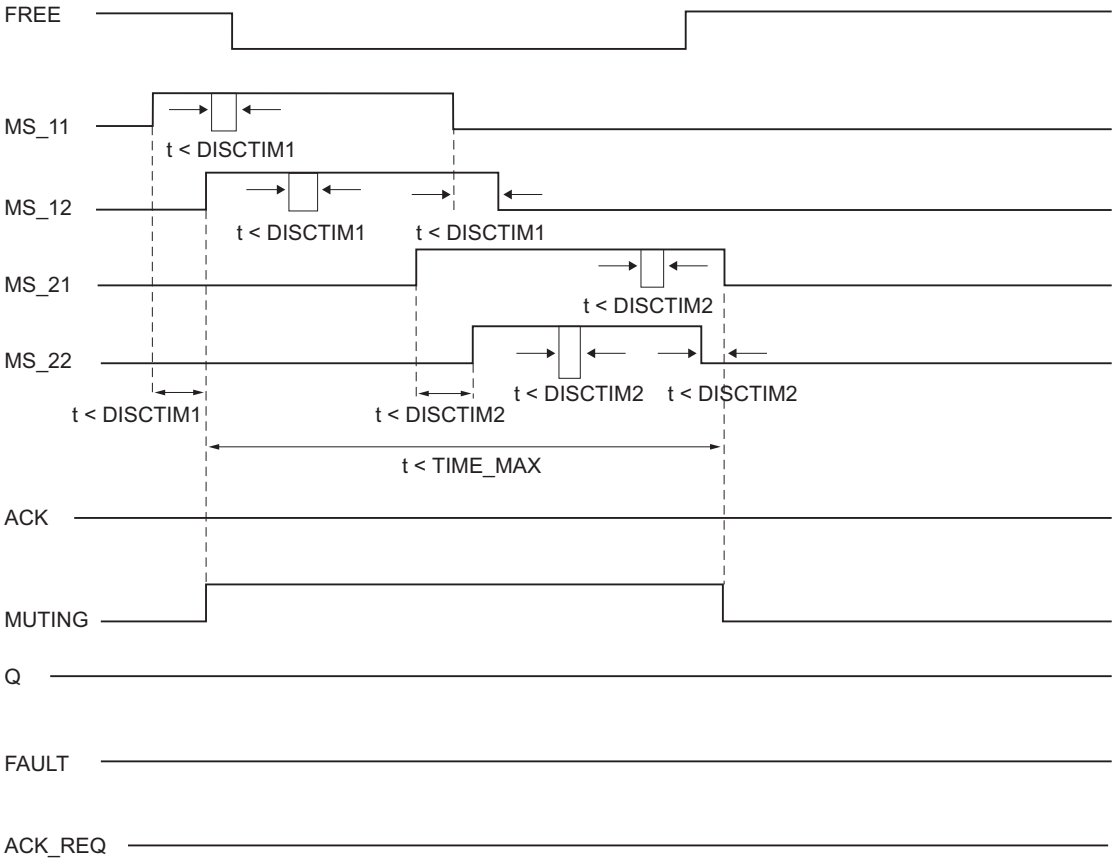


Only if muting sensors MS_21 and MS_22 are both switched to inactive (product enables sensors) is the MUTING function terminated (Q = 1, MUTING = 0). The maximum activation time for the MUTING function is the time set at input TIME_MAX.

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 Four Muting Sensors

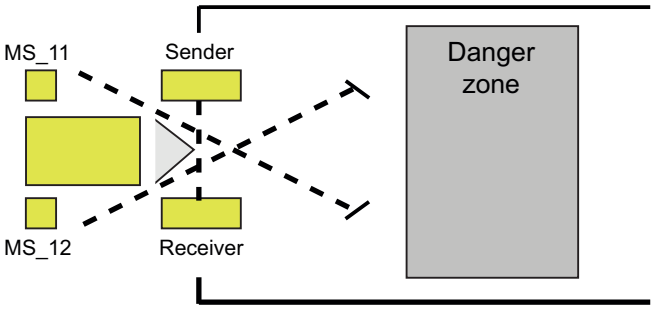


Schematic Sequence of Muting Procedure with Reflection Light Barriers

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 four multiple 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 responds at input QBAD_MUT
- 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 (irrespective 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, if:

- The light curtain is no longer interrupted
- Errors, if present, are eliminated (see output DIAG)
  and
- A user acknowledgement with a positive edge is issued at input ACK (see also Chapter "Implementing User Acknowledgment").

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 block 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 block 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, if:

- 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 signals that FREE is necessary for error elimination and for removal of the restart inhibit. Following a successful FREE, ACK_REQ is reset to 0 by the block.

Note

Once the maximum muting time is exceeded, TIME_MAX is rewound as soon as the MUTING function is restarted.

FREE function

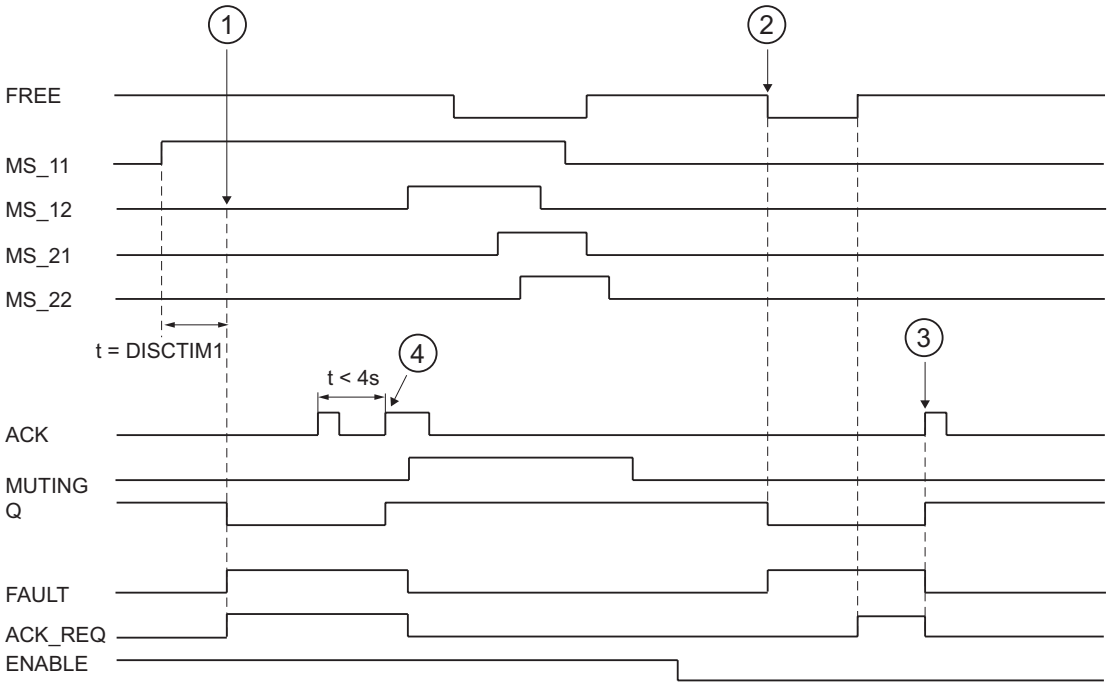
If an error cannot be corrected immediately, the FREE function can be used to free the muting range. Enable signal Q and output MUTING =1 temporarily. The FREE function can be used if:

- 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 managed.

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 monitoring is deactivated while the conveyor equipment has stopped for one of the following reasons:

- 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 assume a signal state of 1 due to an unknown 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).

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 Causes of Problems	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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
		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 No.	Assignment	Possible Causes of Problems	Remedies
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)
		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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
		Startup of F-system	For FREE, see DIAG variable, 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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
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		

Note

Access to the DIAG output is not permitted in the safety program!

See also

F-I/O DB (Page 98)

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Implementing User Acknowledgment in the Safety Program of a I-Slave F-CPU (Page 120)

Overview of F-application blocks (Page 176)

9.1.2.15 FB 215 "F_ESTOP1": Emergency STOP up to Stop Category 1

Connections

	Parameter	Data Type	Description	Default
Inputs:	E_STOP	BOOL	Emergency STOP	0
	ACK_NEC	BOOL	1=Acknowledgment necessary	1
	ACK	BOOL	1=Acknowledgment	0
	TIME_DEL	TIME	Time delay	T# 0 ms
Outputs:	Q	BOOL	1=Enable	0
	Q_DELAY	BOOL	Enable is OFF delayed	0
	ACK_REQ	BOOL	1= Acknowledgment request	0
	DIAG	BYTE	Service information	B#16#0

Principle of Operation

This F-application block implements an emergency STOP shutdown with acknowledgment for Stop Categories 0 and 1.

Enable signal Q is reset to 0, as soon as the E_STOP input assumes 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 only if input E_STOP assumes 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 F-application block sets output ACK_REQ to 1, as soon as input E_STOP = 1.

Following an acknowledgment, the F-application block resets ACK_REQ to 0.

⚠ WARNING

Variable ACK_NEC must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded.

Note

Prior to inserting F-application block F_ESTOP, you must copy F-application block F_TOF from the F-Application Blocks\Blocks block container of the *Distributed Safety* F-library (V1) to the block container of your S7 program, if it is not already present.

⚠WARNING

When using F-application block F_ESTOP1, F-application block F_TOF must have number FB 186 and must not be renumbered!

⚠WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in Chapter "F-Application Blocks")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

The F-application block supports the requirements of EN 418, EN 292-2, and EN 60204-1.

Note: Only one emergency STOP signal (E_STOP) can be evaluated on the F-application block. With suitable configuration (type of sensor interconnection: 2-channel equivalent), discrepancy monitoring of the two NC contacts (when two channels are involved) in accordance with Categories 3 and 4 as defined in EN 954-1 is performed directly by the F-I/O with inputs. In order to keep the discrepancy time from influencing the response time, you must assign "Provide 0-value" for the behavior of discrepancy during configuration.

Startup Characteristics

After an F-system startup, when ACK_NEC = 1, you must acknowledge the F-application block 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 Causes of Problems	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 interlocked	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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
		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	Acknowledgement required (= state of ACK_REQ)	-	-
Bit 7	State of output Q	-	-

Note

Access to the DIAG output is not permitted in the safety program!

See also

F-I/O DB (Page 98)

9.1.2.16 FB 216 "F_FDBACK": Feedback Monitoring

Connections

	Parameter	Data Type	Description	Default
Inputs:	ON	BOOL	1= Enable output	0
	FEEDBACK	BOOL	Feedback input	0
	QBAD_FIO	BOOL	QBAD or QBAD_O_xx signal of F-I/O/channel of output Q (F-I/O DB)	0
	ACK_NEC	BOOL	1=Acknowledgment necessary	1
	ACK	BOOL	Acknowledgment	0
	FDB_TIME	TIME	Feedback time	T# 0 ms
Outputs:	Q	BOOL	Output	0
	ERROR	BOOL	Feedback error	0
	ACK_REQ	BOOL	Acknowledgment request	0
	DIAG	BYTE	Service information	B#16#0

Principle of Operation

This F-application block implements feedback monitoring.

To do this, the signal state of the output Q is checked for equality with 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 after a feedback error between the feedback input FEEDBACK and the output Q, 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 F-application block resets ACK_REQ to 0.

To avoid a feedback error from being detected and an acknowledgment from being required when the F-I/O controlled by output Q are passivated, you must supply input QBAD_FIO with the QBAD or QBAD_O_xx variable of the associated F-I/O.

⚠WARNING

Variable ACK_NEC must not be assigned a value of 0 unless an automatic restart of the affected process following a feedback error is otherwise excluded.

Note

Prior to inserting the F_FDBACK F-application block, you must copy the F_TOF F-application block from the F-Application Blocks\Blocks block container of the Distributed Safety F-library (V1) to the block container of your S7 program, if it is not already present.

⚠WARNING

When using the F_FDBACK F-application block, the F_TOF F-application block must have number FB 186 and must not be renumbered!

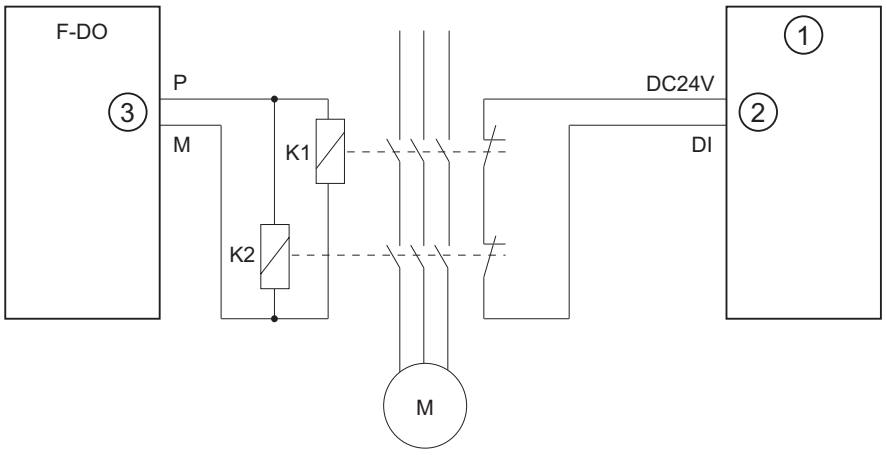
⚠WARNING

When using an F-application block 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 timing of the time base used in the F-application block (see figure in Chapter "F-Application Blocks")
- Tolerance of internal time monitoring in the F-CPU
  - For time values up to 100 ms, a maximum of 20% of the (configured) time value
  - For time values starting at 100 ms, a maximum of 2% of the (configured) time value

You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

Interconnection Example



(1) Standard DI
(2) Input FEEDBACK
(3) Output Q

The feedback contact is wired to a standard I/O module.

Startup Characteristics

After an F-system startup, the F-application block 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, it can be evaluated 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 Causes of Problems	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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
Bit 2	After feedback error: feedback input has permanent signal state of 0	II/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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
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	Acknowledgement required (= state of ACK_REQ)	-	-
Bit 7	State of output Q	-	-

Note

Access to the DIAG output is not permitted in the safety program!

See also

F-I/O DB (Page 98)

Overview of F-application blocks (Page 176)

9.1.2.17 FB 217 "F_SFDOOR": Safety Door Monitoring

Connections

	Parameter	Data Type	Description	Default
Inputs:	IN1	BOOL	Input 1	0
	IN2	BOOL	Input 2	0
	QBAD_IN1	BOOL	QBAD or QBAD_I_xx signal of F-I/O/channel of input IN1 (F-I/O)	0
	QBAD_IN2	BOOL	QBAD or QBAD_I_xx signal of F-I/O/channel of input IN2 (F-I/O)	0
	OPEN_NEC	BOOL	1= Open necessary at startup	1
	ACK_NEC	BOOL	1=Acknowledgment necessary	1
	ACK	BOOL	Acknowledgment	0
Outputs:	Q	BOOL	1= Enable, safety door closed	0
	ACK_REQ	BOOL	Acknowledgment request	0
	DIAG	BYTE	Service information	B#16#0

Principle of Operation

This F-application block implements safety door monitoring.

Enable signal Q is reset to 0 as soon as one of the inputs IN1 or IN2 assumes a signal state of 0 (safety door is opened). The enable signal can be reset to 1, only if:

- Inputs IN1 and IN2 both assume a signal state of 0 prior to opening the door (safety door has been completely opened)
- Inputs IN1 and IN2 then both assume 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 F-application block sets ACK_REQ = 1 as soon as the door is closed. Following an acknowledgment, the F-application block resets ACK_REQ to 0.

In order for the F-application block to recognize whether inputs IN1 and IN2 are 0 merely due to passivation of the associated F-I/O, you must supply inputs QBAD_IN1 or QBAD_IN2 with the QBAD or QBAD_I_xx variable of the associated F-I/O or channel. 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.

⚠WARNING

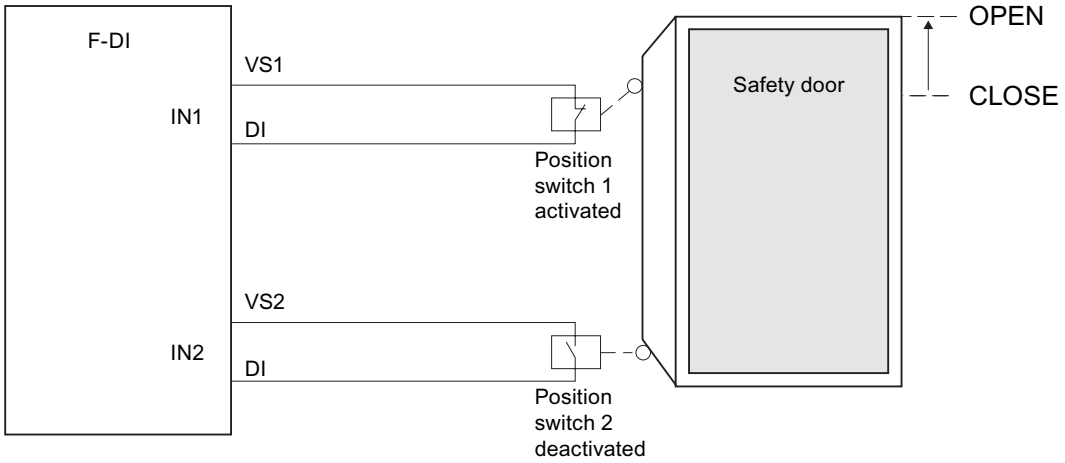
Variable ACK_NEC must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded.

The F-application block supports the requirements of EN 954-1 and EN 1088.

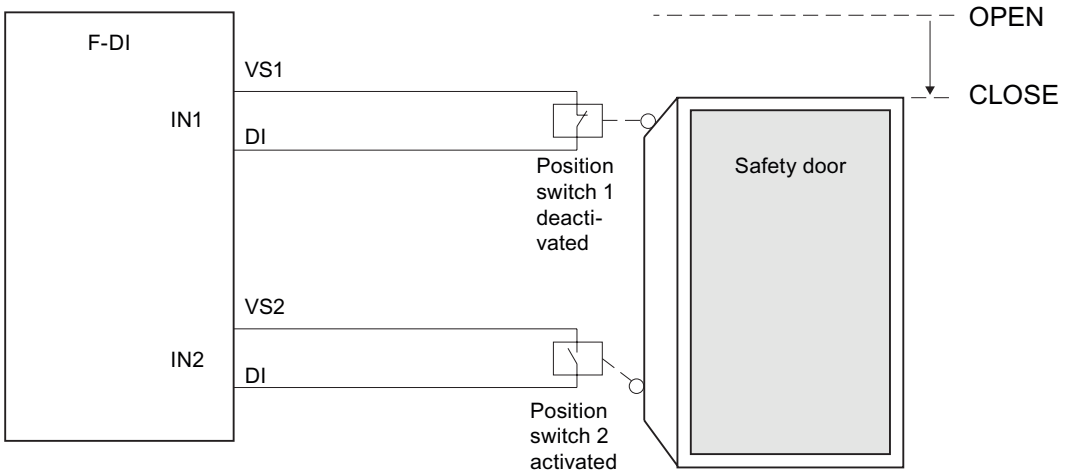
Interconnection Example

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 acknowledgement occurs **independently** of ACK_NEC, as soon as the two inputs IN1 and IN2 assume 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 assume a signal state of 0 (safety door has been completely opened) followed by a signal state of 1 (safety door is closed).

⚠ WARNING
Variable OPEN_NEC must not be assigned a value of 0 unless an automatic restart of the affected process is otherwise excluded.

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 Causes of Problems	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 DIAG variable, bits 0 to 6 in Chapter "F-I/O DB"
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	Acknowledgement required (= state of ACK_REQ)	-	-
Bit 7	State of output Q	-	-

Note

Access to the DIAG output is not permitted in the safety program!

See also

F-I/O DB (Page 98)

Passivation and Reintegration of F-I/O after F-System Startup (Page 106)

9.1.2.18 FB 219 "F_ACK_GL": Global acknowledgment of all F-I/Os in an F-Runtime group

Connections

	Parameter	Data Type	Description	Default
Input:	ACK_REI_GLOB	BOOL	1=acknowledgment for reintegration	0

Principle of operation

This F-application block creates an acknowledgment for the simultaneous reintegration of all F I/Os/channels of the F I/O of an F-runtime group after communication errors or F I/O/channel errors.

For the reintegration an acknowledgment with a positive edge at the input ACK_REI_GLOB is required. The acknowledgement is analogous to the user acknowledgment via the variable ACK_REI of the F-I/O DB, however has a simultaneous effect on all F-I/Os of the F-runtime group in which the F-application block is called.

If you use the F-application block F_ACK_GL, you do not have to provide for a user acknowledgment for each F-I/O of the F-runtime group by means of the variable ACK_REI of the F-I/O DB.

Note

Use of the F_ACK_GL F-application block is only possible if your safety program was created with *S7 Distributed Safety* V5.4 or higher, you have configured channel-level passivation for at least one F-I/O, or at least one F-I/O is connected to PROFINET IO. The F-system block F_IO_CGP is then in the block container of the "S7-Program"

An acknowledgment via F_ACK_GL is only possible if the variable ACK_REI of the F-I/O DB = 0. Accordingly, an acknowledgment via the variable ACK_REI of the F-I/O DB is only possible if the input ACK_REI_GLOB of the F-application block = 0.

The F-application block is only allowed to be called once per F-runtime group.

See also

F-I/O DB (Page 98)

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Implementing User Acknowledgment in the Safety Program of a I-Slave F-CPU (Page 120)

9.1.2.19 FB 223 "F_SENDDP" and FB 224 "F_RCVDP": Send and Receive Data via PROFIBUS DP

Introduction

You use F-application blocks F_SENDDP and F_RCVDP for fail-safe sending and receiving of data by means of:

- Safety-related master-master communication
- Safety-related master-I-slave communication
- Safety-related I-slave-I-slave communication

Connections of F-Application Block F_SENDDP

	Parameter	Data Type	Description	Default
Inputs:	SD_BO_00	BOOL	Send data BOOL 00	0
	...			
	SD_BO_15	BOOL	Send data BOOL 15	0
	SD_I_00	INT	Send data INT 00	0
	SD_I_01	INT	Send data INT 01	0
	DP_DP_ID	INT	Network-wide unique value for address association between F_SENDDP and F_RCVDP	0
	TIMEOUT	TIME	Monitoring time in ms for safety-related communication (see also *Safety Engineering in SIMATIC S7* system manual)	0 ms
	LADDR	INT	Start address of address area: • Of DP/DP coupler for safety-related master-master communication • For safety-related master-I-slave communication • For safety-related I-slave-I-slave communication	0
Outputs:	ERROR	BOOL	1=Communication error	0
	SUBS_ON	BOOL	1=Receiver outputs fail-safe values	1
	RETVAL14	WORD	Error code of SFC 14 (You can find a description of error codes in the online Help for SFC 14.)	0
	RETVAL15	WORD	Error code of SFC 15 (You can find a description of error codes in the online Help for SFC 15.)	0
	DIAG	BYTE	Service information	0

Connections of F-Application Block F_RCVDP

	Parameter	Data Type	Description	Default
Inputs:	ACK_REI	BOOL	1=Acknowledgment for reintegration of send data following communication error	0
	SUBBO_00	BOOL	Fail-safe value for receive data BOOL 00	0
	...			
	SUBBO_15	BOOL	Fail-safe value for receive data BOOL 15	0
	SUBI_00	INT	Fail-safe value for receive data INT 00	0
	SUBI_01	INT	Fail-safe value for receive data INT 01	0
	DP_DP_ID	INT	Network-wide unique value for address association between F_SENDDP and F_RCVDP	0
	TIMEOUT	TIME	Monitoring time in ms for safety-related communication (see also *Safety Engineering in SIMATIC S7* system manual)	0 ms
	LADDR	INT	Start address of address area: • Of DP/DP coupler for safety-related master-master communication • For safety-related master-I-slave communication • For safety-related I-slave-I-slave communication	0
Outputs:	ERROR	BOOL	1=Communication error	0
	SUBS_ON	BOOL	1=Fail-safe values are output	1
	ACK_REQ	BOOL	1=Acknowledgment for reintegration of send data required	0
	SENDMODE	BOOL	1=F_CPU with F_SENDDP in deactivated safety mode	0
	RD_BO_00	BOOL	Receive data BOOL 00	0
	...			
	RD_BO_15	BOOL	Receive data BOOL 15	0
	RD_I_00	INT	Receive data INT 00	0
	RD_I_01	INT	Receive data INT 01	0
	RETVAL14	WORD	Error code of SFC 14 (You can find a description of error codes in the online Help for SFC 14.)	0
	RETVAL15	WORD	Error code of SFC 15 (You can find a description of error codes in the online Help for SFC 15.)	0
	DIAG	BYTE	Service information	0

Principle of Operation

F-application block F_SENDDP sends 16 data elements of data type BOOL and 2 data elements of data type INT in a fail-safe manner to another F-CPU via PROFIBUS DP. There, they can be received by the associated F_RCVDP F-application block.

In F_SENDDP, the data to be sent (for example, outputs of other F-blocks) are applied at inputs SD_BO_xx and SD_I_xx.

In F_RCVDP, the data received are available at outputs RD_BO_xx and RD_I_xx for additional processing by other F-blocks.

The operating mode of the F-CPU with the F_SENDDP is provided at output SENDMODE. If the F-CPU with the F_SENDDP is in deactivated safety mode, output SENDMODE = 1.

Communication between F-CPUs takes place hidden in the background by means of a special safety protocol. You must define an association between an F_SENDDP in one F-CPU and an F_RCVDP in the other F-CPU by assigning a unique address association at the DP_DP_ID inputs of the F_SENDDP and F_RCVDP. Associated F_SENDDPs and F_RCVDPs receive the same value for DP_DP_ID.

⚠ WARNING

The value for each address association (input parameter DP_DP_ID; data type: INT) is user-defined; however, it must be unique from all other safety-related communication connections in the network.

You must supply inputs DP_DP_ID and LADDR with constant values when calling the F-application block. Direct read or write access in the associated instance DB is not permitted in the safety program!

Note

Within a safety program, you must assign a different start address at the LADDR input for each F_SENDDP and F_RCVDB call. You must use a separate instance DB for each F_SENDDP and F_RCVDP call.

The input and output parameters of the F_RCVDP must not be supplied with local data of the F-program block.

You must not use an actual parameter for an output parameter of an F_RCVDP, if it is already being used for an input parameter of the same F_RCVDP call or another F_RCVDP or F_RCVS7 call. The F-CPU can go to STOP if this is not observed. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety Program: internal CPU fault; internal error information: 404"

Startup Characteristics

After the sending and receiving F-systems are started up, communication must be established initially between communication peers F_SENDDP and F_RCVDP. During this time, receiver F_RCVDP outputs the fail-safe values present at its inputs SUBBO_xx and SUBBI_xx.
F_SENDDP and F_RCVDP 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 Event of Communication Errors

If a communication error occurs, for example, due to a test-value error (CRC) or when monitoring time TIMEOUT expires, outputs ERROR and SUBS_ON are set to 1 at both F-application blocks. Receiver F_RCVDP then outputs the fail-safe values assigned at its SUBBO_xx inputs. Output SENDMODE is not updated while output SUBS_ON = 1.
The send data of F_SENDDP present at inputs SD_BO_xx and SUBI_xx are only output again when the communication error is no longer detected (ACK_REQ = 1) and you acknowledge with a positive edge at input ACK_REI.

⚠WARNING

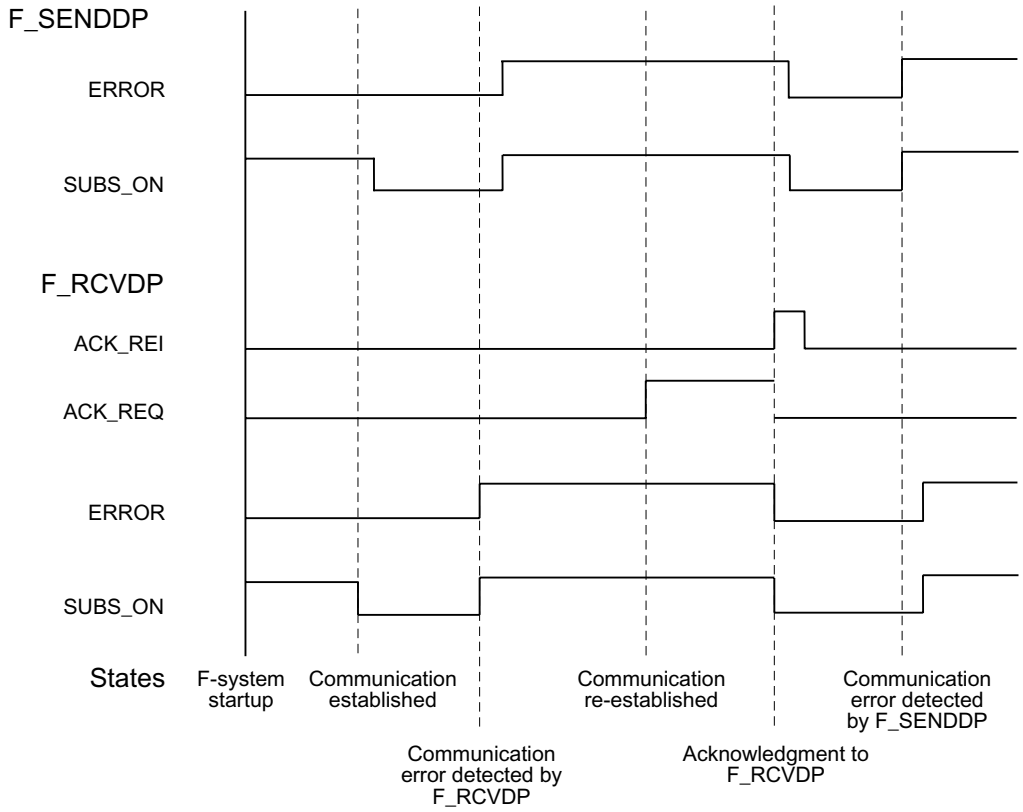
For the user acknowledgment, you must interconnect the ACK_REI input with a signal generated by the operator input.

Interconnections with automatically generated signal is not allowed.

Note that when a communication error occurs, the ERROR output (1=communication error) is set for the first time if communication has already been established between communication peers F-SENDDP and F_RCVDP. If communication cannot be established after startup of the sending and receiving F-systems, check the configuration of the safety-related CPU-CPU communication, F-SENDDP and F_RCVDP parameter assignment, and the bus connection. You can also find possible causes of error by evaluating the RETVAL14 and RETVAL15 outputs.

In general, always evaluate RETVAL14 and RETVAL15, since only one of the two outputs may be able to receive error information.

Timing Diagrams for F_SENDDP/F_RCVDP



Output DIAG

In addition, non-fail-safe information about the type of error that has occurred is provided for service purposes at output DIAG of both F-application blocks F_SENDDP and F_RCVDP.

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_REI.

Structure of DIAG in F-Application Block F_SENDDP/F_RCVDP

Bit No.	Assignment of F_SENDDP and F_RCVDP	Possible Causes of Problems	Remedies
Bit 0	Reserved	-	-
Bit 1	Reserved	-	-
Bit 2	Reserved	-	-
Bit 3	Reserved	-	-
Bit 4	Timeout, detected by F_SENDDP/F_RCVDP	Interference in bus connection to partner F-CPU.	Check bus connection and ensure that no external interference sources are present.
		Monitoring time setting for F-CPU and partner F-CPU is too low.	Check assigned monitoring time parameter TIMEOUT at F_SENDDP and F_RCVDP of both F-CPUs. If necessary, set a higher value. Recompile safety program.
		DP/DP coupler configuration is invalid.	Check DP/DP coupler configuration.
		Internal error of DP/DP coupler	Replace DP/DP 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, detected by F_SENDDP/F_RCVDP	See description for Bit 4	See description for Bit 4
Bit 6	CRC-error, recognized by F_SENDDP/F_RCVDP	See description for Bit 4	See description for Bit 4
Bit 7	Reserved	-	-

Note

Outputs DIAG, RETVAL14, and RETVAL15 cannot be accessed in the safety program.

Additional Information

You will find more information about configuring and programming safety-related communication between safety programs on different F-CPUs in the references provided under "See also".

See also

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Overview of safety-related communication (Page 127)

Configuring Address Areas (Safety-Related Master-Master Communication) (Page 130)

Configuring Address Areas (Safety-Related Master-I-Slave Communication) (Page 140)

Configuring Address Areas (Safety-Related I-Slave-I-Slave Communication) (Page 150)

9.1.2.20 FB 225 "F_SENDS7" and FB 226 "F_RCVS7": Communication via S7 Connections

Introduction

You use the F_SENDS7 and F_RCVS7 F-application blocks for fail-safe sending and receiving data via S7 connections.

Note

In S7 Distributed Safety, S7 connections are generally permitted over Industrial Ethernet only!

Safety-related communication via S7 connections is possible from and to the following CPUs:

- CPU 315F-2 PN/DP (only via PN interface of the CPU)
- CPU 317F-2 PN/DP (only via PN interface of the CPU)
- CPU 416F-3 PN/DP (only via PN interface of the CPU)
- CPU 416F-2 **firmware version V4.0 and higher**

Connections of F-Application Block F_SENDS7

	Parameter	Data Type	Description	Default
Inputs:	SEND_DB	BLOCK_DB	Number of F-communication DB	0
	TIMEOUT	TIME	Monitoring time in ms for safety-related communication (see also *Safety Engineering in SIMATIC S7* system manual)	0 ms
	EN_SEND	BOOL	1= Send enable	1
	ID	WORD	Local ID of the S7 connection (from *NetPro*)	0
	R_ID	DWORD	Unambiguous network address correlation between F_SENDS7 and F_RCVS7	0

	Parameter	Data Type	Description	Default
Outputs:	ERROR	BOOL	1=Communication error	0
	SUBS_ON	BOOL	1=Receiver outputs fail-safe values	1
	STAT_RCV	WORD	Error code of SFB/FB URCV (SFB9/FB9) (for a description of error codes, refer to the Online Help for SFB9)	0
	STAT_SND	WORD	Error code of SFB/FB USEND (SFB 8/FB 8) (For a description of error codes, refer to online Help for SFB 8)	0
	DIAG	BYTE	Service information	0

Connections of F-Application Block F_RCVS7

	Parameter	Data Type	Description	Default
Inputs:	ACK_REI	BOOL	Acknowledgment for reintegration of send data following communication error	0
	RCV_DB	BLOCK_DB	Number of F-communication DB	0
	TIMEOUT	TIME	Monitoring time in ms for safety-related communication (see also *Safety Engineering in SIMATIC S7* system manual)	0 ms
	ID	WORD	Local ID of the S7 connection (from *NetPro*)	0
	R_ID	DWORD	Unambiguous network address correlation between F_SENDS7 and F_RCVS7	0
Outputs:	ERROR	BOOL	1=Communication error	0
	SUBS_ON	BOOL	1=Fail-safe values are output	1
	ACK_REQ	BOOL	1=Acknowledgment for reintegration of send data required	0
	SENDMODE	BOOL	1=F-CPU with F_SENDS7 in deactivated safety mode	0
	STAT_RCV	WORD	Error code of SFB/FB URCV (SFB9/FB9) (for a description of error codes, refer to the Online Help for SFB9)	0
	STAT_SND	WORD	Error code of SFB/FB USEND USEND (SFB 8/FB 8) (For a description of error codes, refer to online Help for SFB 8)	0
	DIAG	BYTE	Service information	0

Principle of Operation

F_SENDS7 sends the send data contained in an F-communication DB to the F-communication DB of the associated F_RCVS7 in a fail-safe manner via an S7 connection.

An F-communication DB is an F-DB for safety-related CPU-CPU communication with special properties. The properties, creation, and editing of F-communication DBs are described in Chapter "Programming Safety-Related CPU-CPU Communication via S7 Connections".

You must specify the numbers of the F-communication DBs at inputs SEND_DB and RCV_DB of F-application blocks F_SENDS7 and F_RCVS7.

The operating mode of the F-CPU with the F_SENDS7 is provided at output SENDMODE of F_F_RCVS7. If the F-CPU with the F_SENDS7 is in deactivated safety mode, output SENDMODE = 1.

To reduce the bus load, you can temporarily shut down communication between the F-CPUs. To do so, supply input EN_SEND of F_SENDS7 with "0" (default = "1"). Then, send data are no longer sent to the F-communication DB of the associated F_RCVS7 and the receiver F_RCVS7 provides fail-safe values for this period (default F-communication DB). If communication was already established between the partners, a communication error is detected.

For F-CPU purposes, the local ID of the S7 connection (from connection table in *NetPro*) must be specified at input ID of F_SENDS7 or F_RCVS7.

Communication between F-CPUs takes place hidden in the background by means of a special safety protocol. You must define a communication association between an F_SENDS7 in one F-CPU and an F_RCVS7 in the other F-CPU by assigning an odd number at the R_ID inputs of the F_SENDS7 and F_RCVS7. Associated F_SENDS7s and F_RCVS7s receive the same value for R_ID.

⚠WARNING

The value for each address association (input parameter R_ID; data type: DWORD) is user-defined; however, it must be unique from all other safety-related communication connections in the network. 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 F-application block. Direct read or write access in the associated instance DB is not permitted in the safety program!

Note

A separate instance DP must be used for each call of an F SENDS7 or F_RCVS7 block. You must not call these F-application blocks as multiple instances.

The input and output parameters of F_RCVS7 must not be supplied with local data of the F-program block.

You must not use an actual parameter for an output parameter of an F_RCVS7, if it is already being used for an input parameter of the same or another F_RCVS7 or F_RCVDP call. The F-CPU can go to STOP if this is not observed. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety Program: internal CPU fault; internal error information: 404"

Startup Characteristics

After the sending and receiving F-systems are started up, communication must be established initially between communication peers F_SENDS7 and F_RCVS7. Receiver F_RCVS7 provides fail-safe values for this time period (default in its F-communication DB). F_SENDS7 and F_RCVS7 signal this at output SUBS_ON with 1. Output SENDMODE of the F_RCVS7 has a default of 0 and is not updated, as long as output SUBS_ON = 1.

Behavior in Event of Communication Errors

If a communication error occurs, for example, due to a test-value error (CRC) or when monitoring time TIMEOUT expires, outputs ERROR and SUBS_ON are set to 1 at F_SENDS7 and F_RCVS7 . Receiver F_RCVS7 then provides the fail-safe values (default in its F-communication DB). Output SENDMODE is not updated while output SUBS_ON = 1. The send data present in the F-communication DB of F_SENDS7 are only output again when the communication error is no longer detected (ACK_REQ = 1) and you acknowledge with a positive edge at input ACK_REI of F_RCVS7.

⚠ WARNING

For the user acknowledgment, you must interconnect the ACK_REI input with a signal generated by the operator input.

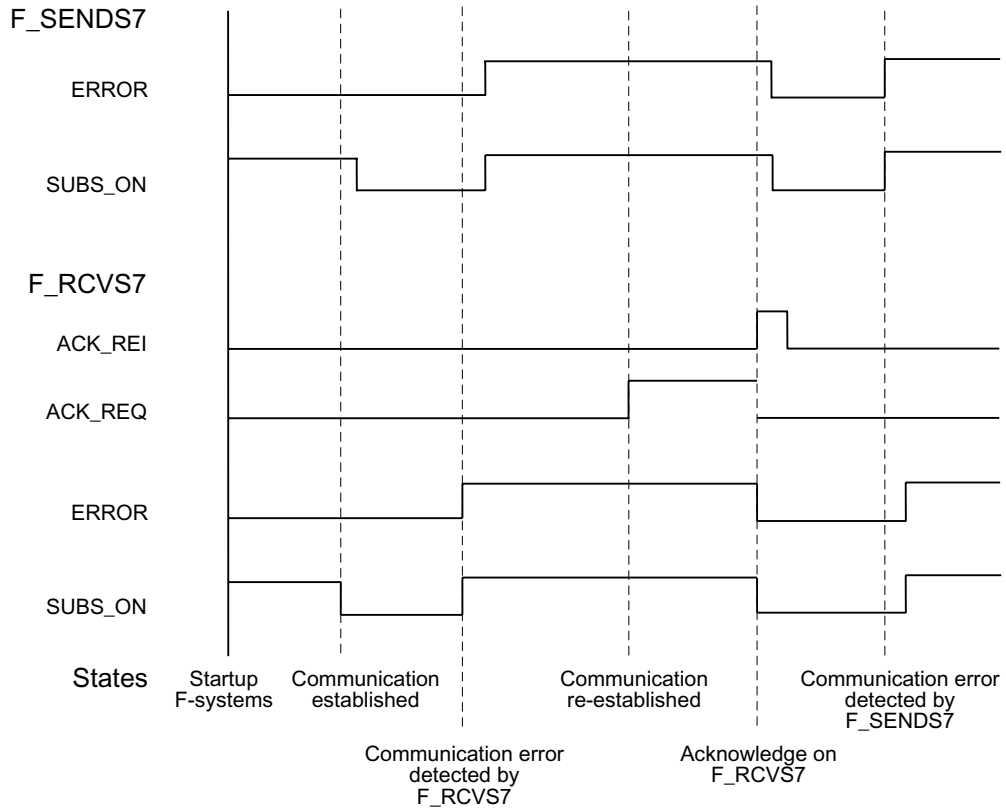
An interconnection with an automatically generated signal is not allowed.

Note that when a communication error occurs, the ERROR output (1=communication error) is set for the first time if communication has already been established between communication peers F-SENDS7 and F_RCVS7. If communication cannot be established after startup of the sending and receiving F-systems, check the configuration of the safety-related CPU-CPU communication, F-SENDS7 and F_RCVS7 parameter assignment, and the bus connection. You can also find possible causes of error by evaluating the STAT_RCV and STAT_SND outputs.

In general, always evaluate STAT_RCV and STAT_SND, since only one of the two outputs may be able to receive 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 the sender side match.

Time Diagram F_SENDS7 and F_RCVS7



Output DIAG

The DIAG output provides non-fail-safe information on the type of communication errors that 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. The DIAG bits are saved until acknowledgment at input ACK_REI of the associated F_RCVS7.

Structure of DIAG

Bit No.	Assignment F_SENDS7 and F_RCVS7	Possible Causes of Problems	Remedies
Bit 0	Reserved	-	-
Bit 1	Reserved	-	-
Bit 2	Reserved	-	-
Bit 3	Reserved	-	-
Bit 4	Timeout detected by F_SENDS7 and F_RCVS7	Interference in bus connection to partner F-CPU	Check bus connection and ensure that no external interference sources are present.
		Monitoring time setting for F-CPU and partner F-CPU is too low	Check assigned monitoring time parameter TIMEOUT at F_SENDS7 and F_RCVS7 of both F-CPUs. If necessary, set a higher value. Recompile safety program.
		CPs in STOP mode, or internal fault in CPs	Switch CPs to RUN mode Check diagnostic buffer of CPs 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 Check diagnostic buffer of F-CPUs Replace F-CPUs, if necessary
		Communication was shut down with EN_SEND = 0.	Enable communication again at associated F_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 numbers detected by F_SENDS7 and F_RCVS7	See description of bit 4	See description of bit 4
Bit 6	CRC-error detected by F_SENDS7 and F_RCVS7	See description of bit 4	See description of bit 4
Bit 7	Reserved	-	-

Note

Access to outputs DIAG, STAT_RCV, and STAT_SND is not permitted in the safety program!

Additional Information

You will find more information about configuring and programming safety-related communication via S7 connections in the references provided under "See also".

See also

Implementing User Acknowledgment in the Safety Program of a DP Master F-CPU or IO Controller (Page 117)

Implementing User Acknowledgment in the Safety Program of a I-Slave F-CPU (Page 120)

Overview of safety-related communication (Page 127)

Configuring safety-related communication using S7 connections (Page 166)

9.1.2.21 FC 174 "F_SHL_W": Shift Left 16 Bits

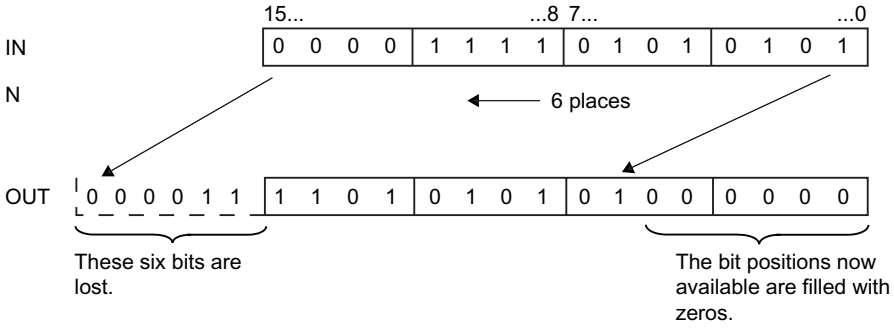
Inputs/Outputs

	Parameter	Data Type	Description	Default
Inputs:	IN	WORD	Value that is shifted	-
	N	INT	Shift number	-
Outputs	OUT	WORD	Result of shift operation	-

Principle of Operation

This F-application block shifts the content of the bits of the value transferred at input IN to the left bit-by-bit. The bit locations that are freed up during the shift operation are filled with zeros. Shift number N indicates by how many bits the content is to be shifted. The result of the shift instruction is provided at output OUT. Output OUT is always 0 when 15 < N ≤ 255.

Note that when N < 0 or N > 255 is specified, only the low byte of the value transferred at input N is evaluated as a shift number.



9.1.2.22 FC 175 "F_SHR_W": Shift Right 16 Bits

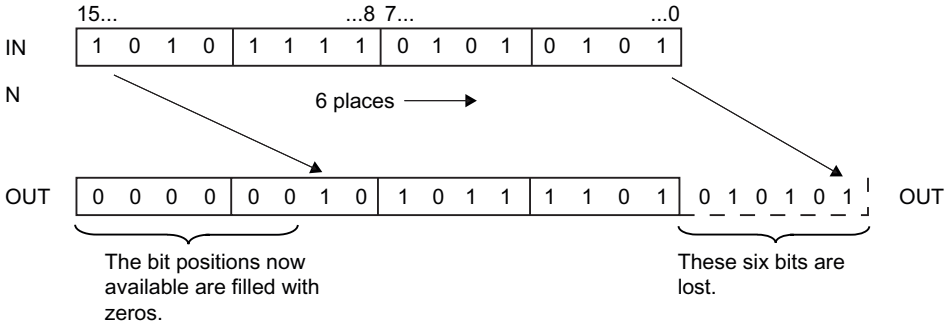
Inputs/Outputs

	Parameter	Data Type	Description	Default
Inputs:	IN	WORD	Value that is shifted	-
	N	INT	Shift number	-
Outputs	OUT	WORD	Result of shift operation	-

Principle of Operation

This F-application block shifts the content of the bits of the value transferred at input IN to the right bit-by-bit. The bit locations that are freed up during the shift operation are filled with zeros. Shift number N indicates by how many bits the content is to be shifted. The result of the shift instruction is provided at output OUT. Output OUT is always 0 when 15 < N ≤ 255.

Note that when N < 0 or N > 255 is specified, only the low byte of the value transferred at input N is evaluated as a shift number.



9.1.2.23 FC 176 "F_BO_W": Convert 16 Data Elements of Data Type BOOL to a Data Element of Data Type WORD

Connections

	Parameter	Data Type	Description	Default
Inputs	IN0	BOOL	Bit 0 of WORD value	0
	IN1	BOOL	Bit 1 of WORD value	0
	...			
	IN15	BOOL	Bit 15 of WORD value	0
Outputs	OUT	WORD	WORD value consisting of IN0 to IN15	0

Principle of Operation

This F-application block 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 ith bit of the WORD value is set to 0 (or 1), if the value at input INi is 0 (or 1).

Note: To supply inputs IN0 to IN15 with Boolean constants "0" and "1", you can access variables "RLO0" and "RLO1" in the F-shared DB using a fully-qualified DB access ("F_GLOBDB".RLO0 or "F_GLOBDB".RLO1).

9.1.2.24 FC 177 "F_W_BO": Convert a Data Element of Data Type WORD to 16 Data Elements of Data Type BOOL

Connections

	Parameter	Data Type	Description	Default
Inputs	IN	WORD	WORD value	0
Outputs	OUT0	BOOL	Bit 0 of WORD value	0
	OUT1	BOOL	Bit 1 of WORD value	0
	...			
	OUT15	BOOL	Bit 15 of WORD value	0

Principle of Operation

This F-application block 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 ith bit of the WORD value is 0 (or 1).

9.1.2.25 FC 178 "F_INT_WR": Write Value of Data Type INT Indirectly to an F-DB

Connections

	Parameter	Data Type	Description
Inputs	IN	INT	Value to be written to the F-DB
	ADDR_INT	POINTER	Start address of the INT area in an F-DB
	END_INT	POINTER	End address of the INT area in an F-DB
	OFFS_INT	INT	Address offset in the INT area

Principle of Operation

This F-application block writes the value of data type INT indicated at input IN to the variable in an F-DB addressed by means of ADDR_INT and OFFS_INT.

The address of the variable addressed by means of ADDR_INT and OFFS_INT must be within the address area defined by addresses ADDR_INT and END_INT.

If the F-CPU has gone to STOP mode with diagnostic event ID 75E2, verify that this condition is satisfied.

The start address of the area with variables of data type INT in an F-DB in which the value at input IN is to be written is transferred using the ADDR_INT input. The associated address offset in this area is transferred using the OFFS_INT input.

The addresses transferred at the ADDR_INT or END_INT inputs must point to a variable of data type INT in an F-DB. Only variables of data type INT are permitted between the ADDR_INT and END_INT addresses. The ADDR_INT address must be smaller than the END_INT address. As shown in the following example, the ADDR_INT and END_INT 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 ADDR_INT, END_INT, and OFFS_INT

Address	Declaration	Name	Type	Initial Value	Comments
0.0	stat		STRUCT		
+0.0	stat	VAR_BOOL10	BOOL	FALSE	
+0.1	stat	VAR_BOOL11	BOOL	FALSE	
+0.2	stat	VAR_BOOL12	BOOL	FALSE	
+0.3	stat	VAR_BOOL13	BOOL	FALSE	
+2.0	stat	VAR_TIME10	TIME	T#0MS	
+6.0	stat	VAR_TIME11	TIME	T#0MS	
+10.0	stat	VAR_INT10	INT	0	<- ADDR_INT = "F-DB",VAR_INT10 Example 1
+12.0	stat	VAR_INT11	INT	0	
+14.0	stat	VAR_INT12	INT	0	
+16.0	stat	VAR_INT13	INT	0	<-OFFS_INT = 3
+18.0	stat	VAR_INT14	INT	0	
+20.0	stat	VAR_INT15	INT	0	<- END_INT = "F-DB",VAR_INT15
+22.0	stat	VAR_BOOL20	BOOL	FALSE	
+22.1	stat	VAR_BOOL21	BOOL	FALSE	
+22.2	stat	VAR_BOOL22	BOOL	FALSE	
+22.3	stat	VAR_BOOL23	BOOL	FALSE	
+24.0	stat	VAR_INT20	INT	0	<- ADDR_INT = "F-DB",VAR_INT20 <-OFFS_INT = 0 Example 2
+26.0	stat	VAR_INT21	INT	0	
+28.0	stat	VAR_INT22	INT	0	
+30.0	stat	VAR_INT23	INT	0	<- END_INT = "F-DB",VAR_INT23
+32.0	stat	VAR_INT30	INT	0	<- ADDR_INT = "F-DB",VAR_INT30 Example 3
+34.0	stat	VAR_INT31	INT	0	<-OFFS_INT = 1
+36.0	stat	VAR_INT32	INT	0	
+38.0	stat	VAR_INT33	INT	0	
+40.0	stat	VAR_INT34	INT	0	<- END_INT = "F-DB",VAR_INT34
+42.0	stat	VAR_TIME20	TIME	T#0MS	
-46.0	stat		END_STRUCT		

9.1.2.26 FC 179 "F_INT_RD": Read Value of Data Type INT Indirectly from an F-DB

Connections

	Parameter	Data Type	Description
Inputs	ADDR_INT	POINTER	Start address of the INT area in an F-DB
	END_INT	POINTER	End address of the INT area in an F-DB
	OFFS_INT	INT	Address offset in the INT area
Outputs	OUT	INT	Value to be read from the F-DB

Principle of Operation

This F-application block reads the variable of data type INT in an F-DB addressed using ADDR_INT and OFFS_INT and makes it available at output OUT.

The address of the variable addressed by means of ADDR_INT and OFFS_INT must be within the address area defined by addresses ADDR_INT and END_INT.

If the F-CPU has gone to STOP mode with diagnostic event ID 75E2, verify that this condition is satisfied.

The start address of the area with variables of data type INT in an F-DB from which the variable is to be read is transferred using the ADDR_INT input. The associated address offset in this area is transferred using the OFFS_INT input.

The addresses transferred at the ADDR_INT or END_INT inputs must point to a variable of data type INT in an F-DB. Only variables of data type INT are permitted between the ADDR_INT and END_INT addresses. The ADDR_INT address must be smaller than the END_INT address.

The ADDR_INT and END_INT addresses must be transferred fully-qualified as "DBx.DBWy" or in the corresponding symbolic representation. Transfers in other forms are not permitted. You will find examples for the parameter assignment of ADDR_INT, END_INT, and OFFS_INT in the references provided under "See also."

See also

FC 178 "F_INT_WR": Write Value of Data Type INT Indirectly to an F-DB (Page 246)

9.1.3 F-System Blocks

Function

F-system blocks are automatically added when the safety program is compiled to create an executable safety program from the safety program you create.

With F-system blocks, fault control measures are automatically added to your safety program, and additional safety-related tests are performed.

Overview of F-System Blocks

The following F-system blocks are available:

- F_CTRL_1
- F_CTRL_2
- F_IO_BOI
- FSIO_BOI
- F_RTGCO2
- F_IO_CGP
- FSIO_CGP
- F_DIAG_N
- FISCA_I
- FICTU
- FICTD
- FICTUD
- FITP
- FITON
- FITOF
- FIACK_OP
- FI2HAND
- FIMUTING
- FI1oo2DI
- FI2H_EN
- FIMUT_P
- FIACK_GL
- FISHL_W
- FISHR_W
- FIBO_W
- FIW_BO
- FIINT_WR
- FIINT_RD

When the safety program is compiled, F-system blocks are automatically added and stored in the number range you have reserved for the "F-function blocks" in order to create an executable safety program from the safety program you have programmed.

Note

You must not insert F-system blocks from the *F-System Blocks* block container in an F-PB/F-FB/F-FC. Likewise, you must not modify (rename) or delete F-system blocks in the *Distributed Safety* F-library (V1) or the block container of your user project.

See also

Overview of Configuration (Page 23)

9.1.4 F-Shared DB

Function

The F-shared data block is a fail-safe block that contains all of the shared data of the safety program and additional information needed by the F-system. When the hardware configuration is saved and compiled in *HW Config*, the F-shared DB is automatically inserted and expanded.

Using the symbolic name of the F-shared DB (i.e., F_GLOBDB), you can evaluate certain data of the safety program in the standard user program.

⚠ WARNING
Do not copy the F-shared DB from a safety program to another safety program (exception: copying the entire S7 program).

See also

Data Transfer from the Safety Program to the Standard User Program (Page 123)

Data Transfer from the Standard User Program to the Safety Program (Page 125)

9.1.5 Custom F-Libraries

Introduction

You have the option of creating your own F-libraries for *S7 Distributed Safety*.

How to Create an F-Library

You create your own F-library as follows:

1. In *SIMATIC Manager,* select **File >New**.
2. In the "Libraries" tab, select "F-library" from the "Type" list.
3. Assign a name to the F-library.
4. Specify the "file path."
5. Close the dialog with "OK." The F-library is created.

Working with User-Created F-Libraries

To use F-FBs/F-FCs/application templates from user-created F-libraries, you must have the same *S7 Distributed Safety* version installed on your PC or programming device that was used to create the F-FBs, F-FCs, or application templates.

You must check yourself whether an existing user-created F-library is still current. If necessary, you must replace a user-created F-library with a newer, available version. *S7 Distributed Safety* does not check the versions of the F-FBs/F-FCs in a user-created F-library. When you compile a safety program, there is also no automatic replacement of F-FBs/F-FCs from a user-created F-library with corresponding F-FBs/F-FCs from a newer version of this F-library. If necessary, copy F-FBs/F-FCs with a newer version from the user-created F-library into the block container of your safety program.

You cannot use symbolic names of F-application blocks of the *Distributed Safety* F-library (V1) for user-created F-FBs, F-FCs, and blocks.

The F-FBs/F-FCs from user-created F-libraries are handled the same as those from the *Distributed Safety* F-library (V1).

Removing S7 Distributed Safety

When you remove *S7 Distributed Safety*, the user-created F-libraries are retained.

Compiling and commissioning a safety program 10

10.1 "Safety Program" Dialog

Introduction

The "Safety Program" dialog provides information about the safety program and contains important functions you can use to edit your safety program.

Note

F-blocks are highlighted in yellow in *SIMATIC Manager* and in the "Safety Program" dialog.

- In *SIMATIC Manager,* know-how protected blocks are also represented with a lock symbol.

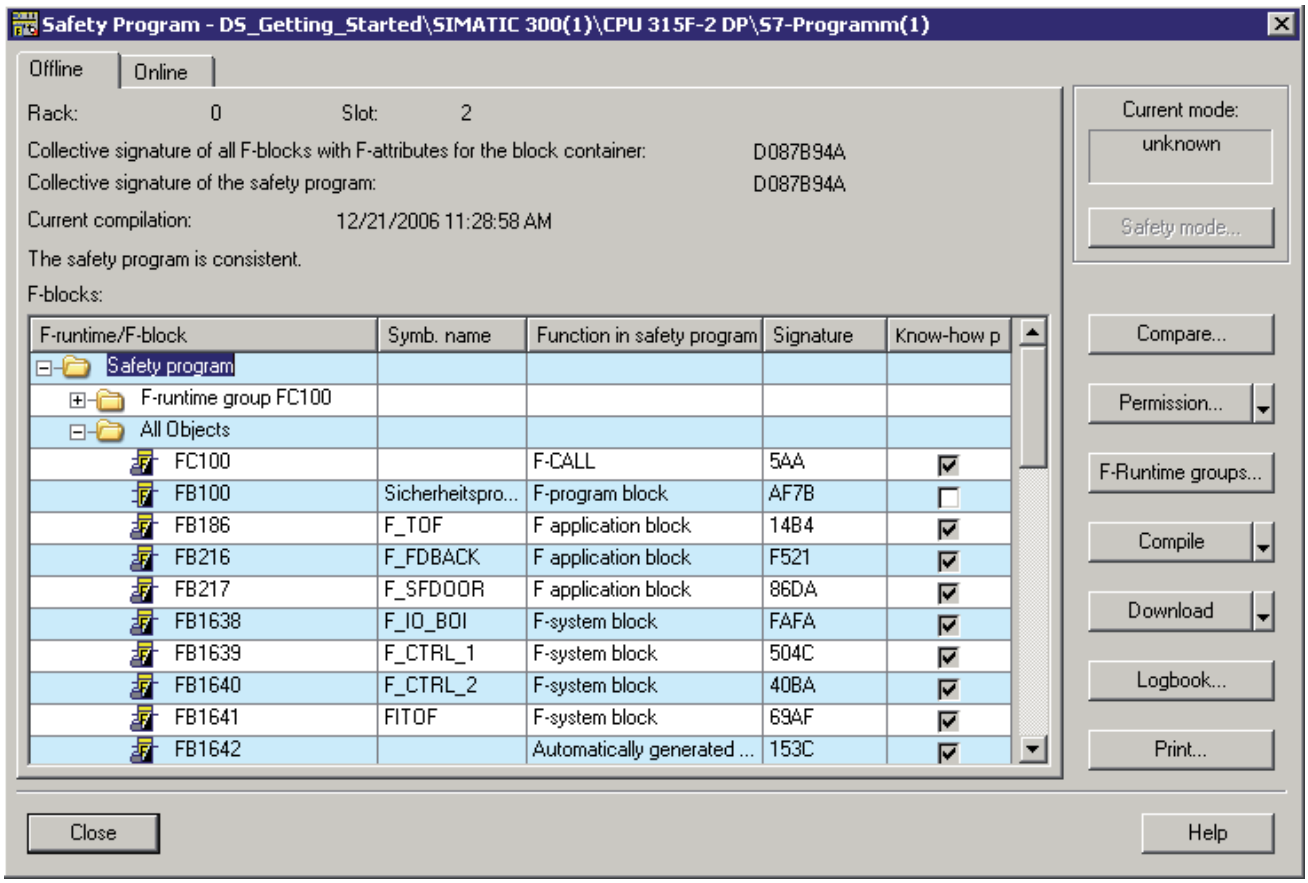
  Once the safety program has been successfully compiled, all blocks of the safety program are know-how protected. The exception to this are any F-blocks you created (F-PB, F-FBs, F-FCs, F-DBs) and did not assign know-how protection to.
- In the "Safety Program" dialog , F-blocks with F-attribute are also represented with an "F" in the block symbol.

Once the safety program has been successfully compiled, only the blocks of the safety program have the F-attribute.

Procedure for Calling the "Safety Program" Dialog

1. Select the correct F-CPU or S7 program assigned to it.
2. In *SIMATIC Manager*, select the **Options > Edit safety program** menu command, or in *STEP 7* V5.4 and higher, select the corresponding icon in the toolbar.

   The "Safety Program" dialog will appear.



Information Regarding F-Blocks of Safety Program

All of the F-blocks of the block container are displayed in this dialog. Use the "Offline"/"Online" tab to choose whether the F-blocks of the offline or online block container are to be listed.

- The "F-Runtime Group..." folder contains the F-runtime group structure of the safety program. The F-Runtime Groups view is displayed only for the offline safety program containing an existing F-shared DB and at least one defined F-runtime group. The names of the F-runtime group folders are formed as follows: "F-runtime group" + name of F-CALL of F-runtime group.

  The following is displayed in the "F-Runtime Group ..." folder: all F-FBs, F-FCs, F-application blocks, instance DBs, F-DBs, the F-CALL, and, if applicable, the DB for F-runtime communication for each respective F-runtime group.

  The "F-Runtime Group" folder also contains an "F-I/O DBs" folder. This folder contains all F-I/O DBs that are addressed from the F-runtime group.

  Note

  If a consistent safety program does not exist, the contents of the "F-runtime group ..." and "F-I/O DBs" folders are not complete.

- The "Complete" folder contains all F-blocks of the offline block container.

  The following properties are displayed for each F-block:

  – Block designation (type/number) with/without F-attribute with/without know-how protection in the block symbol
  – Symbolic block name
  – Function in the safety program
  – Signature of the F-block
  – Know-how protection is/has been selected (for offline safety program)

  Note

  The symbolic names of F-blocks from the Distributed Safety F-library (V1) and automatically generated F-blocks must not be changed. The symbolic name of these F-blocks must always match the header name; otherwise, the safety program compile operation will be aborted.

Information Regarding Safety Program

The following information regarding the safety program is displayed:

- Date of the last compile operation and the collective signatures calculated during compilation:
  - "Collective signature of all F-blocks with F-attribute in the block container"
  - "Collective signature of the safety program": value across all F-blocks called in the F-runtime group of the safety program
- Information regarding the state of the safety program. There are three possible states:
  - Consistent
  - Inconsistent
  - Modified
- "Current Mode:" contains information on whether:
  - The safety mode is "activated" or
  - The safety mode is "deactivated"
  - "CPU is in STOP mode"
  - The status of safety mode is "unknown," that is, it cannot be determined or
  - F-runtime group was not called: The associated F-CALL was not called for at least one F-runtime group (e.g., because no F-CALL call was programmed in an OB (OB35), FB, or FC).

Note

If the text below "Current Mode" is enclosed in square brackets **[**abc**]** , this indicates that the collective signatures of the safety program and/or the passwords for the safety program do not match online and offline. This means one of the following:

- The offline safety program was modified after downloading.
- The wrong F-CPU was addressed. You can verify the latter based on the online collective signature of all F-blocks with F-attribute in the block container.

Click on the title row of the block list to sort the list.

Note that the current safety mode display may not be up to date if the programming device or PC is not directly connected to the F-CPU/intelligent DP slave and the safety program dialog for a safety program located on this F-CPU is opened. In this case, "unknown" is output for the mode.

Solution: Connect the programming device or PC directly to the F-CPU for which the safety program dialog should be opened.

To log the safety program, see Chapter "Printing Project Data of Safety Program".

See also

Safety Program States (Page 257)

Printing out project data (Page 279)

10.2 Safety Program States

Possible states

The safety program can have the following states:

- Consistent

  The collective signature of all F-blocks with F-attribute in the block container is identical to the collective signature of the safety program.

  F-blocks that are not called in the F-runtime group of the safety program are displayed in the "Safety Program" dialog without the F-attribute in the block symbol and are not included in the calculation of the collective signatures. When the safety program is compiled, you are notified about unused F-blocks in the block container.

  For greater clarity, it is recommended that you delete unused F-blocks. On the other hand, it is possible to configure F-I/O that have not (yet) been addressed in the safety program and still compile a consistent safety program. A consistent safety program is required for the safety program acceptance test.

- Inconsistent

  The collective signature of all F-blocks with F-attribute in the block container and the collective signature of the safety program are different, because, for example, an F-block with F-attribute has been copied, but the copied F-block with F-attribute is not called in the F-runtime group of the safety program.

  If the F-CPU contains a safety program with "inconsistent" status, the F-CPU cannot be started up if the F-CPU supports this detection function (see Product Information for the particular F-CPU). To obtain a consistent safety program, you must recompile the safety program.

- Modified

  The collective signature of the safety program is set to "0" because the safety program or the safety-related parameters of the F-CPU and F-I/O were changed.

  The collective signature of all F-blocks with F-attribute in the block container is different from the collective signature of the safety program.

  If the F-CPU contains a safety program with "changed" status, the F-CPU cannot be started up if the F-CPU supports this detection function (see Product Information for the particular F-CPU). If the F-CPU does not support this detection function, the F-CPU can go to STOP mode if a safety program with "changed" status is executed when safety mode is enabled.

  To obtain a consistent safety program, you must recompile the safety program.

See also

Overview of System Acceptance Test (Page 293)

10.3 Compiling Safety Program

Note

Before you compile the safety program, close the *LAD/FBD Editor*, *Display S7 Reference Data*, and *Check Block Consistency* applications, as well as the symbol table.

Procedure for Compiling the Safety Program

1. Select the correct F-CPU or S7 program assigned to it.
2. In *SIMATIC Manager,* select the **Options > Edit Safety Program** menu command.

   The "Safety Program" dialog will appear.
3. Activate the "Compile" button.

   The safety program will now be compiled.

Alternatively, you can compile the safety program using the "Check block consistency" function in *SIMATIC Manager* (see "Check Block Consistency" function in Chapter "Creating and Editing F-FB/F-FC").

Compiling the Safety Program

Compilation is only possible for valid runtime groups. That is, none of the F-blocks you defined in the "F-runtime groups" dialog can be missing in the F-runtime group.

When the safety program is compiled, a consistency check is performed. That is, the safety program is checked for errors and for F-blocks that you created in the block container but did not use in the F-runtime group. Any error messages are output in an error window.

Only F-blocks that are part of the safety program receive an F-attribute. Following a successful compile operation, the block container always contains a consistent safety program composed entirely of F-blocks with F-attribute.

The offline block container can contain F-blocks without F-attribute.

Following a successful consistency check, the additional F-system blocks that are required and the automatically generated F-blocks are added.

Error messages and warnings identified during the compile operation are collected and output in a dialog box when compilation is finished. Warnings are specially labeled.

Using the drop-down arrow on the "Compile" button, you can:

- View and save the log of the most recent compile process
- Enable "Check for accesses from standard"
- Enable or disable “Update reference data”

⚠WARNING

You must not insert F-system blocks from the *F-System Blocks* block container of the *Distributed Safety* library (V1) in an F-PB/F-FB/F-FC. Likewise:

- In the *Distributed Safety* F-library (V1), you must not:
- insert, delete, or rename F-system blocks in the Distributed Safety F-library (V1) or the block container of your user project (offline). This could cause errors during the next compile operation.
- Insert, delete, or rename F-system blocks in the Distributed Safety F-library (V1) or the block container of your user project (online). This could cause the F-CPU to go to STOP mode.

Depending on the extent of the intervention, the compiled safety program may not be executable.

In this case, you must delete all automatically added F-blocks (that is, all F-blocks in *SIMATIC Manager* indicated by a yellow symbol with F-STL programming language or author FALGxxxx, and the F-shared DB); you must then perform the following actions:

- Copy all blocks from the F-Application Blocks block container of the Distributed Safety library (V1) to your user project.
- Save and compile in *HW Config.*
- Define the F-runtime groups.
- Compile the complete safety program.

"Check for Accesses from the Standard Program"

The following is checked:

- Whether OBs, FBs, and FCs from the standard user program are writing to F-DBs of the safety program using **fully qualified DB accesses**.
- Whether OBs, FBs, and FCs from the standard user program are writing to address areas of F-I/O using process image accesses or direct I/O accesses.
- Whether F-blocks are called in OBs, FBs, and FCs of the standard user program.
- Whether the clock memory in the F-block is read-accessed.
  (You have defined the clock memory during configuration of the F-CPU in *HW Config* in the object properties dialog for the F-CPU.)

The result is displayed in a message window.

Note

Note that the checks described above are not exhaustive, e.g., the check to determine whether F-DBs are write-accessed from the standard user program is unsuccessful in the event of indirect addressing or partially qualified access to F-DBs in the standard user program.

"Updating Reference Data"

You can disable the reference data update at the end of the compilation operation. This shortens the time required to compile the complete safety program.

Note: If updating of reference data is disabled, the program structure may be displayed incorrectly in the reference data.

Updating of reference data is enabled by default.

The setting applies to the current Windows user.

See also

Creating and editing F-FB/F-FC (Page 77)

10.4 Downloading the Safety Program

Introduction

Once you have compiled your safety program, you can download it to the F-CPU. You have the following options:

- Downloading the entire safety program in the "Safety Program" dialog in STOP mode. This is the recommended method for downloading a consistent safety program.
- Downloading the changes in the safety program in the"Safety Program" dialog in STOP mode
- Downloading individual F-blocks in *SIMATIC Manager* or *FBD/LAD Editor*

Procedure for Downloading the Entire Safety Program to the F-CPU in the "Safety Program" Dialog

1. Select the correct F-CPU or S7 program assigned to it.
2. In *SIMATIC Manager*, select the **Options > Edit Safety Program** menu command.

   The "Safety Program" dialog will appear.
3. Activate the "Download" button.

   All F-blocks with F-attribute belonging to the safety program are identified and downloaded to the F-CPU.

   A note is displayed offering you the option of downloading the standard user program in addition to the safety program (provided this prompt is enabled).

   If the safety program has been modified or is not consistent, you are notified of the option to generate (compile) a consistent safety program.

4. Confirm the prompt indicating that the F-CPU will be stopped.

Note

To download the entire safety program, the F-CPU must be in STOP mode.

If you are downloading F-blocks only, the blocks in which the F-CALL blocks are called (e.g., cyclic interrupt OB35) are not downloaded. You must then download these OBs separately the same way as for a standard program.

Note

When you download the safety program in the "Safety Program" dialog, an online/offline comparison is automatically performed for all F-blocks with F-attribute in the safety program. All F-blocks without F-attribute are deleted in the F-CPU. The F-CPU now contains exactly the same F-blocks with F-attribute as the offline block container.

5. In the "Safety Program" dialog, select the "Offline" and "Online" tabs in turn to check whether the collective signatures of all F-blocks with F-attribute in the block container match offline and online. If they match, downloading was successful. If not, repeat the download operation.
6. To activate safety mode, switch the F-CPU from STOP to RUN mode.

Note

If the download operation is aborted, you must repeat the download step (step 3) and the recheck the collective signatures of all F-blocks with F-attribute in the block container online and offline (step 5).

Procedure for Downloading Changes to the Safety Program in the "Safety Program" Dialog

1. Select the correct F-CPU or S7 program assigned to it.
2. In *SIMATIC Manager,* select the **Options > Edit Safety Program** menu command.

   The "Safety Program" dialog will appear.
3. Click the down-arrow "Download Changes" on the "Download" button.

   All new and changed F-blocks with F-attribute in the safety program are identified and downloaded to the F-CPU.

The rest of the procedure is the same as for downloading the entire safety program in the "Safety Program" dialog (see above).

Note

Note that downloading changes in the safety program is intended for the commissioning phase only. Prior to the acceptance test of the safety program, you must download the complete safety program to the F-CPU. Failure to do so could result in different online and offline time stamps for the F-blocks in the block container.

Downloading the Safety Program to a Programming Device or PC

Note

In principle, it is possible to download 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.

After you upload a safety program to a programming device or PC, you can download it to the F-CPU again without repeating acceptance testing as long as the safety program was not modified. The safety program you downloaded to the F-CPU again can only be executed if:

- The F-CPU did not execute the safety program prior to uploading it to the programming device or PC.
- The hardware configuration of the safety-related communication (see Chapter "Configuring and Programming Communication") has not been changed.

Note

If the safety program has been changed or has already been executed in the F-CPU, you must do the following before downloading the **complete** safety program to the F-CPU again:

1. Delete all instance DBs of F-blocks from the block container
2. Reinsert all F-blocks used in the safety program from the "Distributed Safety" library (V1) or from a custom F-library in the offline block container, thereby overwriting existing F-blocks
3. Reassign constants for parameters of F-blocks from the "Pointer" data type (required for F-blocks F_INT_WR, F_INT_RD only)
4. Recompile the safety program. This recreates the deleted instance DBs.

The F-CPU can go to STOP mode if this is disregarded. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety program: internal CPU fault; internal error information: 404"

⚠ WARNING

A modification to the safety program causes a change in the collective signature, and, consequently, a new acceptance test may be required.

Downloading to an S7-PLCSIM

You can test the safety program with the S7-PLCSIM function (hardware simulation) of *STEP 7*.

Requirements for Downloading to an S7-PLCSIM

- The S7-PLCSIM V5.3 (or higher) optional package is installed on your programming device or PC.
- You have write authorization for the directory where the Distributed Safety F-library (V1) is installed.
- S7-PLCSIM is active. To activate S7-PLCSIM, select **Options > Simulate Modules** in *SIMATIC Manager*.

  The S7-PLCSIM application is started and the "CPU" subwindow is displayed.
- A hardware configuration with F-CPU is downloaded. To download this hardware configuration, open *HW Config* and download the desired configuration the same way as you would download it to a real CPU.
- The safety program is consistent.

Procedure for Downloading to an S7-PLCSIM

1. Select the correct F-CPU or S7 program assigned to it.
2. In *SIMATIC Manager*, select the **Options > Edit Safety Program** menu command.

   The "Safety Program" dialog will appear.
3. In the "Safety Program" dialog, press the "Download" button.
   All F-blocks with F-attribute belonging to the safety program are identified and downloaded to S7-PLCSIM.
4. Confirm the prompt indicating that the F-CPU will be stopped.

   Note

   S7 Distributed Safety automatically determines whether the target device is a "real" F-CPU or S7-PLCSIM. If the target device is S7-PLCSIM, special simulation blocks (F-system blocks) are downloaded automatically from the *S7 Distributed Safety* F-library (V1) to S7-PLCSIM.

   Your offline safety program is unchanged and consistent following the download operation to the S7-PLCSIM. The collective signature of all F-blocks with F-attribute no longer matches the collective signature in S7-PLCSIM.

   Because the safety program is not changed offline for support of S7-PLCSIM, it can also be downloaded to an F-CPU after being downloaded to S7-PLCSIM. To download the safety program to an F-CPU, simply deactivate S7-PLCSIM.
5. You must re-download the safety program to the S7-PLCSIM following each S7-PLCSIM STOP.

It is also possible to download changes in the safety program to an S7-PLCSIM (see above).

Downloading in *SIMATIC Manager* or *FDB/LAD Editor*

F-blocks and standard blocks can be simultaneously downloaded to the F-CPU using standard *STEP 7* tools. 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 deactivated safety mode. If not, you have the option of switching to deactivated safety mode or placing the F-CPU in STOP mode.

Be aware that the consistency of the safety program in the F-CPU cannot be guaranteed when individual F-blocks are downloaded. Therefore, use the download from the "Safety Program" dialog with the F-CPU in STOP to ensure a consistent safety program.

Note

If *S7 Distributed Safety* detects an inconsistent safety program during startup of the F-CPU, the F-CPU cannot be started up if the F-CPU supports this detection function (see Product Information for the particular F-CPU). The following diagnostic event is then entered in the diagnostic buffer of the F-CPU:

- “Inconsistent safety program”

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 when safety mode is enabled. One of the following diagnostic events is then entered in the diagnostic buffer of the F-CPU:

- "Data corruption in the safety program prior to output to F-I/O"
- "Data corruption in the safety program prior to output to partner F-CPU"
- "Safety program: internal CPU fault; internal error information: 404"

While it is possible to download a safety program to an S7-PLCSIM in *SIMATIC Manager* or *FBD/LAD Editor,* no simulation blocks are automatically downloaded and the therefore safety program cannot run. Downloading individual F-blocks in *SIMATIC Manager* or *FBD/LAD Editor* to an S7 PLCSIM in deactivated safety mode is only practical for test purposes.

⚠ WARNING

If F-blocks are downloaded in *SIMATIC Manager* or *FBD/LAD Editor*, you must ensure that there is not an unused F-CALL in the block container. If you always download the safety program in the "Safety Program" dialog, all uncalled F-blocks - including an unused F-CALL block - are automatically deleted.

Rules for Downloading F-Blocks in *SIMATIC Manager* or *FBD/LAD Editor*

The following rules apply to downloading of F-blocks:

- You can only download in deactivated 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 with the "Safety Program" dialog.
- The offline password and online password of the safety program must match.
- Changes to the password for the safety program ("Permission" button in the "Safety Program" dialog) can only be activated in the F-CPU by downloading the safety program using the "Safety Program" dialog.
- Only an offline safety program is permitted to be used as a source program.

Consequently, the "Safety Program" dialog must be used to download the safety program for the first time and after any change to the password for the safety program.

If F-blocks cannot be downloaded (because the F-CPU is in safety mode or because no password or the wrong password was entered for the safety program), you are notified of the option to continue downloading the remaining standard blocks.

See also

Testing the Safety Program (Page 289)

10.5 Work Memory Requirement for Safety Program

Estimation

You can estimate the work memory requirement for the safety program as follows:

Work Memory Requirement for Safety Program

31 Kbytes for F-system blocks F_CTRL_1, F_CTRL_2, F_IO_CGP/F_IO_BOI, and F_DIAG_N

\+ 4.3 Kbytes for F-system block F_RTGCO2 (for F-run-time group communication only)

\+ 4.5 x work memory requirement for all F-FB/F-FC/F-PB

\+ 4.5 x work memory requirement for all F-blocks used (except F_SENDDP, F_RCVDP, F_SENDS7, and F_RCVS7)

\+ Work memory requirement for F_SENDDP and F_RCVDP F-application blocks used (4.4 Kbytes each)

\+ Work memory requirement for F_SENDS7 and F_RCVS7 F-application blocks used (9.5 Kbytes each)

Work Requirement for Data

5 x work memory requirement for all F-DBs (including F-communication DB, but excluding DB for F-run-time group communication) and I-DBs for F-PB/F-FB

\+ 24 x work memory requirement for all DBs for F-run-time group communication

\+ 2.3 x work memory requirement for all I-DBs of F-application blocks (except F_SENDDP, F_RCVDP, F_SENDS7, and F_RCVS7)

\+ Work memory requirement for all I-DBs of the F-application blocks F_SENDDP (0.2 Kbyte), F_RCVDP (0.3 Kbyte), F_SENDS7 (0.6 Kbyte), and F_RCVS7 (1.0 Kbyte).

\+ 0.7 Kbyte per F-FC (including F-application block of type FC)

\+ 0.7 Kbyte per F-I/O (for F-I/O DBs, etc.)

\+ 4.5 Kbytes

Block Size of Automatically Generated F-Blocks

To ensure that the automatically generated F-blocks do not exceed the maximum possible size in the particular F-CPU, observe the following:

- An F-FB/F-FC/F-PB should be not exceed 25% of the maximum size of the FBs or FCs (see *Technical Specifications in the manual for the F-CPU you are using*).
- F-FBs/F-FCs/F-PBs must comply with the following:

2 x number of all parameters or static data of data type BOOL

\+ 4 x number of all parameters or static data of data type INT/WORD

\+ 6 x number of all parameters or static data of the data type TIME

\+ 36

< Maximum size of data blocks in bytes (see *Technical Specifications in the manual for the F-CPU you are using*)

- F-DBs must comply with the following:

2 x number of all variables of the F-DB of data type BOOL

\+ 4 x number of all variables of the F-DB of data type INT/WORD

\+ 6 x number of all variables of the F-DB of data type TIME

\+ 36

< Maximum size of data blocks in bytes
(see *Technical Specifications in the manual for the F-CPU you are using*)

If you receive the message "Block x could not be copied" when you download your safety program to the F-CPU, check whether these conditions are met. Reduce the following, as necessary:

- Size of F-FB/F-FC/F-PB
- Number of parameters and static data of F-FBs/F-FCs/F-PBs
- Number of variables of F-DBs
- Number of blocks. You must not exceed the maximum block limit of the F-CPU (*see Technical Specifications in the manual for the F-CPU you are using*).

10.6 Function Test of Safety Program and Protection through Program Identification

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.

Transferring the Safety Program to the F-CPU with a Programming Device or PC

F-CPUs with Inserted Memory Card (Flash Card or MMC)

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 MMC
  (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 target F-CPU, you must comply with the following procedure when transferring the safety program to the F-CPU with a **programming device or PC** to ensure that the F-CPU does not contain an "old" safety program:

- For F-CPUs with MMC: Download the safety program to the F-CPU in the "Safety Program" dialog.
- For F-CPUs with inserted Flash Card: Download the safety program to the F-CPU in the "Download User Program to Memory Card" dialog.
- Perform a program identification (that is, check to determine whether the collective signatures of all F-blocks with F-attribute in the block container 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 MMC for F-CPUs 3xxF and IM 151-7 F-CPU or Flash Card for F-CPUs 4xxF).

⚠WARNING

If **multiple F-CPUs** can be reached over a network (such as MPI) by **one programming device or PC**, you must take the following actions to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, such as a uniform password for the F-CPUs having the respective MPI address as an extension: "Password_8".

Note the following:

- A point-to-point connection must be used when assigning a password to an F-CPU for the first time (analogous to assigning an MPI address to an F-CPU for the first time).
- Before downloading a safety program to an F-CPU for which access authorization by means of an F-CPU password does not yet exist, you must first revoke existing access authorization for any other F-CPU.

F-CPUs without Inserted Flash Card

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 target F-CPU, you must comply with the following procedure when transferring the safety program to the F-CPU with a **programming device or 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 to the F-CPU in HW Config.
- Download the safety program to the F-CPU in the "Safety Program" dialog.
- Perform a program identification (that is, check to determine whether the collective signatures of all F-blocks with F-attribute in the block container match online and offline).

⚠WARNING

If **multiple F-CPUs** can be reached over a network (such as MPI) by **one programming device or PC**, you must take the following actions to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, such as a uniform password for the F-CPUs having the respective MPI address as an extension: "Password_8".

Note the following:

- A point-to-point connection must be used when assigning a password to an F-CPU for the first time (analogous to assigning an MPI address to an F-CPU for the first time).

Before downloading a safety program to an F-CPU for which access authorization by means of an F-CPU password does not yet exist, you must first revoke existing access authorization for any other F-CPU.

Transferring the Safety Program to the F-CPU with a Memory Card

Use of MMC or Flash Card

The following warning applies when the safety program is transferred using a:

- Flash Card (e.g., for CPU 416F-2)
- MMC (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 target F-CPU, you must comply with the following procedure when transferring the safety program to the F-CPU with a memory card (MMC or Flash Card) to ensure that the F-CPU does not contain an "old" safety program:

- Turn off the power to the F-CPU. For F-CPUs with battery backup (e.g., CPU 416F-2), remove the battery, if present. (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 unknown, remove the F-CPU.)
- Remove the Memory Card (MMC or Flash Card) with the old safety program from the F-CPU.
- Insert the Memory Card (MMC or Flash Card) with the new safety program in 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 must make sure that the inserted memory card (MMC or Flash Card) contains the correct safety program. You can do so through a program identification or other measures, such as a unique identifier on the memory card (MMC or Flash Card).

When downloading a safety program to a memory card (**MMC** or Flash Card), you must adhere to the following procedure:

- Download the safety program to the memory card (MMC or flash card).
- Perform a program identification - in other words, check whether the collective signatures of all F-blocks with F-attribute in the offline block container and on the memory card (MMC or Flash Card) match.
- Affix an appropriate label to the memory card (MMC or Flash Card).

The procedure outlined must be ensured through organizational measures.

See also

Comparing Safety Programs (Page 273)

10.7 Modifying the Safety Program

10.7.1 Modifying the safety program in RUN mode

Introduction

Changes to the safety program during operation (in RUN mode) can only be made in deactivated safety mode. You make changes to F-blocks offline in the *FBD/LAD Editor* in the same way as for a standard program. F-blocks cannot be modified online.

Note

If you do **not** want to modify the safety program during operation, see Chapter "Creating F-Blocks in F-FBD/F-LAD".

Procedure for Modifying the Safety Program in RUN Mode

1. Modify and save the F-PB or F-FB and its associated instance DB, F-FC, or F-DB in the *FBD/LAD Editor*.
2. Download the modified F-block from the *FBD/LAD Editor* to the F-CPU. If you want to download several modified F-blocks, select and download them in *SIMATIC Manager*. The procedure for downloading F-blocks in deactivated safety mode is the same as for a standard program. Observe the applicable rules for the download sequence in the online Help for *STEP 7* .
3. If safety mode is active, a dialog box for deactivating safety mode will appear. Confirm this dialog box.

   Note

   When downloading in *SIMATIC Manager,* you can only download fail-safe blocks created by you (F-PB, F-FB, F-FC, or F-DB), F-application blocks, or standard blocks and their associated instance DBs in deactivated safety mode. If you download automatically added F-blocks (F-SBs or automatically generated F-blocks and associated instance DBs or F-shared DB), the F-CPU can go to STOP mode or safety mode can be activated.

   Therefore, when downloading in *SIMATIC Manager,* always select individual F-blocks instead of the "Station," "S7 Program," or "Block Container" objects.

Restrictions on Safety-Related CPU-CPU Communication

During operation (in RUN mode), you cannot establish new safety-related CPU-CPU communication by means of a new F_SENDDP/F_RCVDP, F_SENDS7/F_RCVS7 block pair.

To establish new safety-related CPU-CPU communication you must always recompile the relevant safety program and download it in its entirety to the F-CPU in STOP mode after inserting a new block call for F_SENDDP, F_SENDS7, F_RCVDP or F_RCVS7.

Restrictions on F-Runtime Group Communication

You cannot make any changes to the safety-related communication between F-runtime groups in RUN mode. That means you may not add, delete, or modify a DB for F-runtime communication in the "Define New F-Runtime Groups" or "Edit F-Runtime Groups" dialogs or in *SIMATIC Manager*.

Following changes in the F-runtime group communication, you must always recompile the safety program and download it in its entirety to the F-CPU in STOP mode.

Restrictions on F-I/O Access

If during operation (in RUN mode), you insert an F-I/O access to an F-I/O of which no single channel or variable from the associated F-I/O DB in the safety program has yet been used, the F-I/O access only becomes effective when the safety program is recompiled and downloaded in its entirety to the F-CPU in STOP mode.

Modifications to the Standard User Program

You can download modifications to the standard user program when the F-CPU is in RUN mode, regardless of whether safety mode is activated or deactivated.

⚠ WARNING

In safety mode, access by means of the F-CPU password must not be authorized when making changes to the standard user program, since changes to the safety program can also be made. To rule out this possibility, you must configure **Level of Protection 1**. If only **one person** is authorized to change the standard user program **and** the safety program, level of protection "2" or "3" 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).

Modifying the F-Runtime Group Call

If an OB (e.g., OB35) or FB with an F-CALL call is downloaded to the F-CPU during operation (in RUN mode), the mode is only updated after the "Safety Program" dialog has been closed and re-opened.

Procedure for Applying Changes to the Safety Program

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 accept changes to the safety program:

1. Compile the safety program in the "Safety Program" dialog.
2. Use the "Safety Program" dialog to download the complete safety program to the F-CPU in STOP mode and activate safety mode by switching the F-CPU from STOP to RUN mode.
3. Follow the steps described in Chapter "Safety Program Acceptance Test".

See also

Configuring the F-CPU (Page 26)

Creating F-Blocks in FFBD/FLAD (Page 76)

Compiling Safety Program (Page 258)

Downloading the Safety Program (Page 260)

Safety Program Acceptance Test (Page 297)

10.7.2 Comparing Safety Programs

Criteria for Comparing Safety Programs

You can compare two safety programs according to the following criteria:

- Collective signature of all F-blocks with F-attribute in the block container
- Parameters of individual F-blocks
- Signatures of individual F-blocks

You can compare the signatures of F-blocks to identify modified or deleted F-blocks.

Comparable Safety Programs

You can compare a safety program with the following:

- Online safety program (online version of this safety program)
- Offline safety program (any offline safety program)
- Online safety program (any online safety program)
- A safety program on a memory card
- A safety program of a reached station

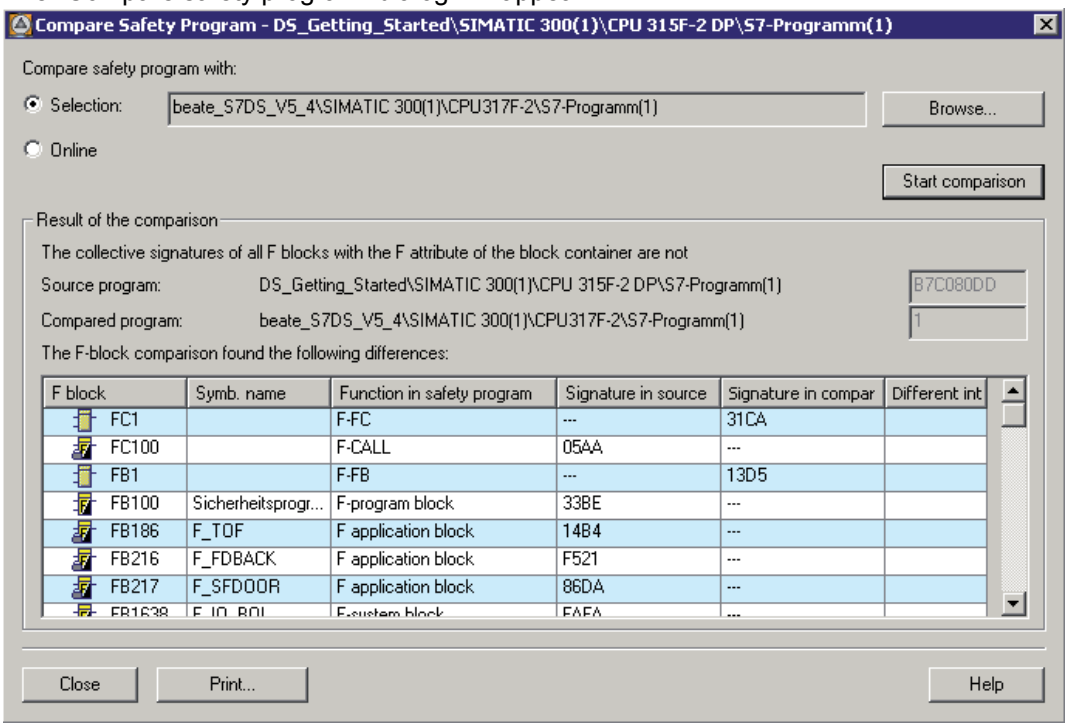
Procedure for Comparing Safety Programs

To compare two safety programs:

1. Select the correct F-CPU or S7 program assigned to it.
2. In *SIMATIC Manager,* select the **Options > Edit Safety Program** menu command.

   The "Safety Program" dialog will appear.
3. Click the "Compare" button.

   The "Compare safety program" dialog will appear.



4. Select the safety program you would like to compare with. Activate the "Browse..." button to indicate its path.
5. Activate the "Start comparison" button.

The required block comparison is executed, and the different F-blocks are displayed in tabular form in the dialog box.

Result of Comparison

The comparison result displays modified F-blocks (different entries in the "Signature in Source Program" and "Signature in Compared Program" columns), F-blocks located in the source program only (entry in "Signature in Source Program" column only), and F-blocks located in the compared program only (entry in "Signature in Compared Program" column only). The **"Interface Different"** column indicates whether or not changes have occurred in the declaration table of F-blocks.

The result can be printed out with the "Print" button.

If you are comparing an offline safety program with an online safety program and the connection to the F-CPU is interrupted during the comparison, the comparison result will be incorrect.

Assignment of Changes

You can assign the changes in the safety program on the basis of the modified F-blocks indicated in the comparison result:

Modified F-block	Change in Safety Program
F-program block, F-FB, F-FC	• Change in this block • Change the declaration table in called FBs/FCs or in F-PB/F-FB/F-FC of F-DBs used • Change in the declaration table in F-FBs contained as multi-instances • Missing F-FBs called as multi-instances
I-DB for F-program block, I-DB for F-FB	• Change in the declaration table of the F-PB/F-FB for each I-DB
F-application block F-system block	• Modified version of F-block (for example, due to use of F-blocks from a new version of *S7 Distributed Safety*) • Missing F-FBs called as multi-instances
I-DB for F-application block	• Modified version of associated F-application block
F-DB	• Change in the declaration table of the F-DB
F-I/O DB	• Change in the hardware configuration of the respective F-I/O • Change in F-parameters of the F-CPU • Modified version of F-system blocks
Automatically generated F-block	• Change in the maximum cycle time of the F-runtime group • Change in F-parameters of the F-CPU • Modified version of F-system blocks • Change in the F-runtime group communication, for example, change in the number of a DB for F-runtime group communication

Modified F-block	Change in Safety Program
F-CALL	• Change in the assignment of the F-PB and its instance DB • Change in the F-I/O addressed in the safety program • Change in read access to data of the standard user program • Change in F-parameters of the F-CPU • Modified version of F-system blocks • Change in the F-runtime group communication

The changes can also occur in combination, meaning that changes to an F-block can have multiple causes.

If no modified F-blocks are indicated, but the collective signature is different, differences exist in the automatically generated blocks, which are not included in the comparison. This can occur, for example, if you renumber F-blocks or modify the resources reserved for the safety program in the object properties dialog for the F-CPU in *HW Config*.

10.7.3 Deleting the Safety Program

Deleting Individual F-Blocks

To delete an F-block, follow the same procedure as for a standard program.

Deleting an F-runtime Group

1. In the "Edit F-Run-Time Groups" dialog, select the folder of the F-runtime group to be deleted.
2. Press the "Delete" button.
3. Close the dialog with "OK".

The assignment of the F-blocks to an F-runtime group is deleted. However, the F-blocks continue to exist.

Deleting the Entire Safety Program

1. Delete all F-blocks highlighted in yellow offline in *SIMATIC Manager*.
2. In *HW Config*, select the F-CPU and select the **Edit > Object Properties** menu command. Open the "Protection" tab and deactivate the "CPU Contains Safety Program" option. Save and compile the hardware configuration.

   The offline project no longer contains a safety program.
3. The following applies to F-CPUs with an inserted memory card (MMC or Flash Card):

   To delete a safety program on a Memory Card (MMC or Flash Card), insert the Memory Card (MMC or Flash Card) in the programming device or PC. In *SIMATIC Manager,* select the **File > S7 Memory Card > Delete** menu command.

   You can now copy the offline standard user program to the Memory Card (MMC or Flash Card).

   The following applies to F-CPUs without an inserted Flash Card:

   You can delete the safety program by resetting the module in the SIMATIC Manager (menu command**PLC > Reset**).

   You can then download the offline standard user program to the F-CPU.

10.7.4 Logbook of the Safety Program

Logbook

Changes and actions for a safety program are logged in a logbook. Various user actions result in corresponding entries in the logbook.

Each safety program has its own logbook. Entries are listed in chronological order. A logbook can contain up to 300 entries. When the number of entries exceeds 300, the entries are overwritten in order.

The logbook function for the safety program is not safety-related as defined in IEC 61508.

Contents of the Logbook

Entries are made in the safety program logbook for the following actions:

- Changing the hardware configuration for a safety program
- Creating an F-block
- Saving an F-block
- Renaming an F-block
- Rewiring an F-block
- Changing object properties of an F-block
- Deleting an F-block
- Changing an F-runtime group
- Compiling the safety program
- Deactivating safety mode
- Downloading F-blocks
- Downloading a safety program or safety program changes

Example of a logbook entry:

Action: Creating F-block FB1

Entry in the logbook: Date, time (time of entry in the logbook), user ID, program path, action "F-block FB1 created"

Displaying, Saving, Printing, and Copying the Logbook

1. Select the F-CPU or the S7 program assigned to it.
2. In SIMATIC Manager, select the **Options > Edit safety program** menu command or the corresponding icon in the toolbar.

   The "Safety Program" dialog will appear.
3. Click "Logbook…".

   This opens the logbook (message window).

   You can save the logbook as a text file in your Windows directory structure and print it later.

When a safety program is copied, the logbook associated with the safety program, if present, is also copied.

Safety Program < V5.4 SP1

If the safety program was created with an earlier version of *S7 Distributed Safety* (prior to V5.4 SP1), the logbook will not be available until a logbook-relevant action has been performed with V5.4 SP1 or higher.

10.8 Printing out project data

Introduction

The "Print" button in the "Safety Program" dialog allows you to print out all important project data of the hardware configuration and the safety program that you need, for example, for the system acceptance test. The signatures in the footer of the printouts ensure that the printouts are explicitly associated with a safety program.

Note

Before you print out the project data, close the *HW Config* and *LAD/FBD Editor* applications and the symbol table.

Procedure for Printing All Important Project Data of the Hardware Configuration and the Safety Program

1. In *SIMATIC Manager*, select the correct F-CPU or S7 program assigned to it.
2. In *SIMATIC Manager, select the* **Options > Edit Safety Program** menu command.

   The "Safety Program" dialog will appear.
3. Click the "Print" button.

Then, you can select the print content:

- "Function Block Diagram/Ladder Diagram":

  All F-blocks (F-PB, F-FB, F-FC, F-DB) that you created in the safety program in the applicable programming language. For F-DBs, the data view is printed.
- "Safety program":

  List of all F-blocks of the safety program and other data relevant to the acceptance test (see Chapter "Printed Project Data for the Safety Program")
- "Hardware Configuration..." (see Chapter "Printed Project Data for the Hardware Configuration")
- "Symbol table"

You must print out all print content for the system acceptance test.

Footer of the Printouts

The following information is displayed in the footer of the printouts:

- Collective signature of all F-blocks with F-attribute in the block container
- Signature of symbols (only for printout of the offline safety program)
- Version identifier of *S7 Distributed Safety* used to create the printouts
- Depending on the status of the safety program: "Safety program changed," "Safety program not changed," or "Symbols changed"

Note

If "Symbols changed" is output, it signifies that assignments for global or local symbols have changed (e.g., changes in the symbol table or to parameter names of F-DBs or F-FBs) and the changes were not made in all affected F-FB/F-FCs.

To correct this situation, use the "Check block consistency" function (see *STEP 7 online help*). If necessary, you must recompile the safety program.

10.8.1 Printed Project Data for the Hardware Configuration

Procedure

If you have selected the "Hardware Configuration..." print content, a follow-up dialog is displayed.

1. Select "All" as the print area. The printout will then include the "Module description" and the "Address list".
2. Select the "Including parameter description" option to include your parameter descriptions in the printout.

Printed Information

The following information in the printout of the hardware configuration ("Hardware Configuration..." print content) is important for the configuration acceptance test:

- The following F-CPU parameters:
  - Protection level
  - F-parameters
- All parameters of the F-I/O

Procedure for Safety-Related I-Slave-Slave Communication

For the system acceptance test with safety-related I-slave-slave communication, you also need a printout of the parameters of the F-I/O that you address via safety-related I-slave-slave communication. The printout of the hardware configuration of the station with the DP master contains this information.

If the CPU of the DP master is an F-CPU to which a safety program is assigned, you can print out the parameters of these F-I/O by selecting the F-CPU of the DP master or the S7 program assigned to it in *SIMATIC Manager* and initiating a printout using the **Options > Safety Program** menu command as described above.

If the CPU of the DP master is a standard CPU, you print out the parameters of these F-I/O as follows:

1. Select the station with the DP master.
2. In *SIMATIC Manager,* select the **Print > Object Content** menu command.

   A follow-up dialog is displayed.
3. Select "All" as the print area. The printout will then include the "Module description" and the "Address list".
4. Select the "Including parameter description" option to include your parameter descriptions in the printout.

10.8.2 Printed Project Data for the Safety Program

Printed Information

The printout of the safety program ("Safety program" print content) contains the following information important for the safety program acceptance test:

- Collective signatures:
  - "F-blocks with F-attribute in the block container" (= "collective signature of all F-blocks with F-attribute in the block container" in the "Safety Program" dialog; also displayed in the footer of the printout)
  - "Safety Program" (= "collective signature of the safety program" in the "Safety Program" dialog = value of the "F_PROG_SIG" variable in the F-shared DB)

  These two signatures must match for the acceptance test.

  Differences between the two signatures generally indicate that the safety program has been changed or is inconsistent. This is also indicated in the footer.
- Version identifier of *S7 Distributed Safety* last used to compile the safety program
- Time when safety program was compiled
- Message if the amount of local data reserved for the safety program has been exceeded
- List of all F-blocks contained in the block container (Square brackets enclosing the block name and signature designate F-blocks without F-attribute)

  Information provided for each F-block:
  - Block number
  - Symbolic name
  - Function in the safety program (F-CALL, F-program block, etc.)
  - Signature
  - Initial value signature for all F-FBs not generated automatically
- List of parameters for safety-related CPU-CPU communication, such as:
  - DP_DP_ID and LADDR of F_SENDDP, F_RCVDP
  - ID, R_ID, and number of the F-communication DB of F_SENDS7, F_RCVS7
  - TIMEOUT of F_SENDDP, F_RCVDP, F_SENDS7, F_RCVS7

The following information is provided for parameters:

- Parameter name
- Name of the associated F-application block
- Numbers of instance DBs used to call the F-application block
- Name of F-block in which the F-application block is called
- Network number of call
- Name of F-runtime group (Name of F-CALL)
- Parameter value

- List of data transferred from the standard user program:
  - Address
  - Symbol
  - F-runtime group in which the data element is used
- List of data for data exchange between the F-runtime groups
  - Number of the F-CALL of the "sender" F-runtime group
  - Number of the F-CALL of the "receiver" F-runtime group
  - Number of the DB for F-runtime group communication
- Runtime group information for each F-runtime group:
  - Number of the F-CALL
  - Symbolic name of the F-CALL
  - Number of the called F-program block
  - Symbolic name of the F-program block
  - Number of the associated instance DB, if applicable
  - Symbolic name of the associated instance DB
  - Maximum cycle time of the F-runtime group
- List of all F-blocks used in the F-runtime group except F-system blocks, the F-shared DB, and automatically generated F-blocks. (Square brackets around the block name and signature designate F-blocks without F-attribute.)

  Information provided for each F-block:
  - Block number
  - Symbolic name
  - Function in the safety program (F-CALL, F-program block, etc.)
  - Signature
  - Initial value signature for all F-FBs not generated automatically

- List of the F-I/O addressed in the F-runtime group (that is, not for all F-I/O configured in *HW Config*, but rather only for those F-I/O actually used):
  - Symbolic name of the F-I/O DB
  - Number of the F-I/O DB
  - Start address
  - Name/identifier of the F-I/O
  - Module type
  - F_Monitoring_Time
  - Cyclic redundancy check by means of parameter assignment (in order to allow quick detection of changes on the I/O)
  - PROFIsafe source and target address
  - PROFIsafe mode
  - Type of passivation
- The following information is indicated for the F-shared DB of the safety program:
  - Number of the F-shared DB
  - Symbolic name F_GLOBDB
  - Absolute and symbolic address of the safety program's collective signature
  - Absolute and symbolic address for reading out the operating mode
  - Absolute and symbolic address for reading out error information
  - Absolute and symbolic address for reading out the compilation time
  - Absolute and symbolic address of the RLO 0
  - Absolute and symbolic address of the RLO 1
- Additional information
  - The setting of the "Safety mode can be deactivated" parameter for the safety program
  - Printout created on
  - Total number of pages in this printout

See also

"Safety Program" Dialog (Page 253)

10.9 Testing the Safety Program

10.9.1 Overview of Testing the Safety Program

Testing Options

In general, all read-only test functions (such as variable monitoring) are also available for safety programs and in safety mode. While all F-blocks can be used as the monitored object, this is only useful for the F-blocks created by you (F-PB, F-FB, F-FC, and F-DB). Monitoring is available without restrictions.

It is possible to modify data of the safety program using the "Monitor/modify variable" function and to gain write access using *HW Config* or *FBD/LAD Editor.* However, restrictions apply and safety mode must be deactivated. Other write accesses to the safety program are not permitted and can cause the F-CPU to go to STOP mode.

Testing with S7-PLCSIM Function of STEP 7

You can test the safety program with the S7-PLCSIM V5.3 and higher (hardware simulation) function of STEP 7. You use S7-PLCSIM in the same way as for standard user programs.

Note

You can use F-application blocks F_SENDDP, F_RCVDP, F_SENDS7, F_RCVS7 in conjunction with the S7-PLCSIM function (hardware simulation) of *STEP 7*. Note, however, that the F-application blocks constantly signal "communication errors" when they are run in the simulation CPU.

10.9.2 Deactivating 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 deactivate safety mode of the safety program to download changes to the safety program in RUN mode. Safety mode remains deactivated until F-CPU is next switched from STOP to RUN mode.

You can enable or disable the option for deactivating the safety mode in the object properties of the F-CPU, "F-Parameter" tab.

⚠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:

- Deactivation of 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 operation monitoring and manual safety shutdown.
- Deactivation of safety mode must be indicated.
  The printout of the safety program contains the address of the variables in the F-shared DB ("F_GLOBDB".MODE) that you can evaluate to read out the operating mode (1 = deactivated safety mode). Thus, not only is the deactivated safety mode displayed on the programming device or PC in the dialog box for deactivating 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 operator control and monitoring system generated by evaluating the "Deactivated Safety Mode" variable in the F-shared DB.
- Changes in the safety program in RUN mode when safety mode is deactivated can cause changeover effects to occur. The procedure for downloading F-blocks in deactivated safety mode is the same as for a standard program. Observe the applicable rules for the download sequence in the online Help for *STEP 7* .
- To the extent possible, the standard user program and the safety program should be modified separately, and changes should be downloaded; otherwise, an error could be downloaded simultaneously to the standard user program, thus disrupting a necessary protective feature or causing changeover effects to occur in both the safety program and the standard program.
- It must be possible to verify that safety mode has been deactivated. A log is required, if possible by recording messages to the operator control and monitoring system, but if necessary, through organizational measures. In addition, it is recommended that deactivation of safety mode be indicated on the operator control and monitoring system.
- Safety mode is deactivated across the F-CPU only. You must take the following into account for safety-related CPU-CPU communication: If the F-CPU with the F_SENDDP or F_SENDS7 is in deactivated 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 F_RCVDP or F_RCVS7 by evaluating SENDMODE.

Requirements for Deactivating Safety Mode

The "Safety mode can be deactivated" parameter in the "F-parameter" tab of the F-CPU in *HW Config* is enabled (see Chapter "Configuring the F-CPU").

The F-CPU is in RUN mode and safety mode is activated.

Procedure for Deactivating Safety Mode

1. Select the correct F-CPU or S7 program assigned to it.
2. In *SIMATIC Manager,* select the **Options > Edit Safety Program** menu command.

   The "Safety Program" dialog will appear.
3. If you are prompted to enter the password for the F-CPU, do so now.
4. Check to see whether "Safety mode activated" is indicated as the "Current mode" . If so, continue with the next step; if not, stop the process, because safety mode is already deactivated or cannot be deactivated.

   Note

   If the text below "Current mode:" is enclosed in square brackets [abc], this indicates that the collective signatures of the safety program and/or the passwords for the safety program do not match online and offline. This means one of the following:
   - The offline safety program was modified after downloading.
   - The wrong F-CPU was addressed. You can verify the latter based on the online collective signature of all F-blocks with F-attribute in the block container.
5. Activate the "Safety mode" button, and enter the password for the online safety program.

   If the password is not valid, safety mode is not deactivated and remains active.
6. If you enter the correct password, another prompt will appear, which also contains the collective signature of the safety program in the F-CPU. Check to see whether this is the collective signature you expected.
7. If it is not the collective signature you expected, verify that you have addressed the correct F-CPU and check to see whether the F-CPU contains the correct F-blocks. To do this, close all *STEP 7* applications and then open the "Safety Program" dialog; this is necessary to prevent multiple applications from accessing the F-CPU simultaneously.
8. Confirm the prompt to deactivate safety mode with "OK".

   Safety mode will be deactivated.

You can now download changes in the safety program to the F-CPU during operation (in RUN mode).

Note

To activate safety mode, the F-CPU must be switched from STOP to RUN mode.

Switching the F-CPU from STOP to RUN mode always activates safety mode, even if the safety program has been modified or is not consistent. The MODE variable in the F-shared DB is set to "0". Keep this in mind when you evaluate the MODE variable to read out the operating mode.

If you have modified your safety program, but have not recompiled and downloaded it, the F-CPU can revert to STOP mode.

Evaluating Safety Mode/Deactivated Safety Mode

If you wish to evaluate safety mode/deactivated safety mode in the safety program, you can evaluate the "MODE" variable in the F-shared DB (1 = deactivated safety mode). You access this variable with fully qualified access ("F_GLOBDB".MODE). The number and symbolic name of the F-shared DB and the absolute addresses of variables are indicated in the printout of the safety program.

You can use this evaluation, for example, to passivate F-I/O when the safety program is in deactivated safety mode. To do so, assign the "MODE" variable in the F-shared DB to all "PASS_ON" variables in the F-I/O DBs of the F-I/O that you wish to passivate.

⚠ WARNING

When the safety program is in deactivated safety mode, the "MODE" variable in the F-shared DB is also evaluated in deactivated safety mode.

Even if the F-I/O are passivated in deactivated safety mode as a result of evaluation of the "MODE" variable, system safety must be ensured in deactivated safety mode through other organizational measures, such as operation monitoring and manual safety shutdown.

See also

Modifying the safety program in RUN mode (Page 271)

10.9.3 Testing the Safety Program

Introduction

In deactivated 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 *STEP 7* tools.

Modifying the Data of the Safety Program with "Monitor/Modify Variable" Function

In addition to data in the standard user program, which can always be modified, you can modify the following data in a safety program using the "Monitor/Modify Variable" function in deactivated safety mode:

- Process image of F-I/O
- F-DBs (except DB for F-runtime group communication), instance DBs of F-FBs
- Instance DBs of F-application blocks
- F-I/O DBs (for permitted signals, see Chapter "F-I/O DB")

Note

F-I/O can only be modified in RUN mode of the F-CPU. You must allocate a separate row in the variable table for each channel 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 one variable table. You can use more than one variable table.

You cannot modify configured F-I/O in which no single channel or variable from the associated F-I/O DB has been used. Therefore, always use at least one variable from the associated F-I/O DB or at least one channel of the F-I/O to be controlled in your safety program.

As a trigger point, you must set "Begin scan cycle" or "End scan cycle". Note, however, that regardless of the trigger point setting, requests to modify inputs (PII) of F-I/O always become effective before the F-PB is executed and requests to modify outputs (PIQ) always become effective after execution of the F-PB.

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 object properties for the F-I/O in *HW Config* (see *F-I/O manuals*), modify requests affect the PIQ only, and not the F-I/O.

As the trigger frequency, you can set "Once" or "Permanently".

⚠WARNING

Permanent modification of F-I/O remains active in the following cases:

- The connection between the programming device and the F-CPU is broken (by removing the bus cable)
- The variable table no longer responds

These modify requests can only be deleted through a memory reset of the F-CPU or by switching the F-CPU from STOP to RUN mode while at the same time disconnecting the F-CPU from the programming device or PC.

Wiring Test

The wiring test is simplified by using symbolic names for the signals.

You can carry out a wiring test for an input by modifying an input signal and verifying whether or not the new value arrives at the PII.

You can carry out a wiring test for an output by modifying the output with the Modify function and verifying whether the required actuator responds.

For the wiring test (for both inputs and outputs), 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 variable 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 requires the use of *STEP 7* with the *S7 Distributed Safety* optional package. If an operator control and monitoring system or *STEP 7* without the *S7 Distributed Safety* optional package is used to modify variables, the F-CPU can go to STOP mode.

Testing and commissioning functions are selected with standard *STEP 7* tools (*FBD/LAD Editor*/*Variable Editor/HW Config*). An attempt to modify a safety program in safety mode is rejected with a corresponding error message, or a dialog box for deactivating safety mode is provided. In certain circumstances, a modify request can cause the F-CPU to go to STOP mode.

Opening F-Blocks

The *FBD/LAD Editor* can be used to open an F-block online in the F-CPU as a write-protected block only, that is, you cannot modify an F-block directly in the F-CPU, even if safety mode is deactivated. Instead, you must edit it offline and then download it.

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 actual value and compiling the safety program offline, as well.

Modify only the parameters described in this documentation.

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 deactivate safety mode. 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 in the "Safety Program" dialog.

Note

If you use the "Monitor/Modify Variable" function to test a safety program, this function does not detect all additional changes you make using other applications in the F-CPU.

For example, if the collective signature of the safety program is changed through revision/modification while safety mode is deactivated, the change may not be detected and an old collective signature may continue to be displayed.

In such cases, terminate the "Monitor/Modify Variable" function and restart the function in order to work with updated data.

"Control at contact" function

The "Control at contact" function supported in *STEP 7* V5.2 and higher is not supported for F-blocks.

Procedure for Testing the Safety Program

The following procedure is used for testing:

1. Deactivate safety mode.
2. Monitor and modify the required F-data and/or F-I/O from a variable table, *HW Config*, or *FBD/LAD Editor*.
3. Terminate existing modify requests after testing is complete before activating safety mode.
4. To activate safety mode, switch the F-CPU from STOP to RUN 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 and immediately continuing testing until the safety program behaves according to your requirements.

You can find additional information about modifying the safety program in RUN mode in Chapter "Modifying the Safety Program in RUN Mode".

Testing the Safety Program with S7-PLCSIM

You can monitor and modify variables of your safety program in an S7-PLCSIM and perform other write access functions in your safety program.

To use S7-PLCSIM, you only have to download your consistent safety program to an S7-PLCSIM.

Note

If you would like to modify variables in an S7-PLCSIM, you must deactivate safety mode beforehand.

Otherwise, the S7-PLCSIM can go to STOP mode. You can only deactivate safety mode in the "Safety Program" dialog.

For a detailed description of the S7-PLCSIM function of *STEP 7*, refer to the *S7-PLCSIM V5.x* user manual.

Program structure of the safety program in S7 Distributed Safetyprocess or fail-safe valuesChanges to the safety program in RUN)

See also

Structure of the Safety Program in S7 Distributed Safety (Page 57)

Process Data or Fail-Safe Values (Page 97)

F-I/O DB (Page 98)

Downloading the Safety Program (Page 260)

Modifying the safety program in RUN mode (Page 271)

Deactivating Safety Mode (Page 286)

System Acceptance Test

11

11.1 Overview of System Acceptance Test

Introduction

During the system acceptance test, all relevant application-specific standards must be adhered to as well as the procedure described below. This also applies to systems that are not "subject to acceptance testing". For the acceptance test, you must consider the systems in the Certification Report.

As a general rule, the acceptance test of an F-System is performed by independent experts.

Requirements

The hardware configuration and parameter assignment is complete.

The safety program has been created and compiled and is consistent.

Procedure

Use the following procedure for the system acceptance test:

1. Back up the entire the STEP 7 project.
2. Select the "Offline" tab in the "Safety Program" dialog.
3. Print the project data with all print content (see Chapter "Printing the Project Data").
4. Check all printouts (see Chapter "Checking the Printouts").
5. Download the complete safety program to the F-CPU (see Chapter "Checks after Downloading the Safety Program to the F-CPU").
6. Carry out a complete function test.

See also

Downloading the Safety Program (Page 260)

Printing out project data (Page 279)

Testing the Safety Program (Page 289)

11.2 Checking the Printouts

Procedure

Check the printouts as follows:

1. Check whether the two signatures in the footer of the printout matches in all four printouts:
   - Collective signature of all F-blocks with F-attribute in the block container
   - Signature of symbols
2. Check whether "Symbols changed" is output in the footer of the printout.
3. Check the printout of the hardware configuration (see Chapter "Configuration Acceptance Test of F-CPU and F-I/O").
4. Check the printout of the F-blocks you created (F-PBs, F-FBs, F-FCs, and F-DBs).
5. Check the printout of the symbol table.
6. Check the "Safety program" printout (see Chapter "Safety Program Acceptance Test").

Note

If "Symbols changed" is output, it signifies that assignments for global or local symbols have changed (e.g., changes in the symbol table or to parameter names of F-DBs or F-FBs) and the changes were not made in all affected F-FB/F-FCs.

To correct this situation, use the "Check block consistency" function (see *STEP 7 online help*). If necessary, you must recompile the safety program.

11.2.1 Acceptance Test for the Configuration of the F-CPU and the F-I/O

Checking the Hardware Configuration ("Hardware Configuration..." print content)

1. Check the parameters of the F-CPU in the printout.

   In particular, check the protection level setting of the F-CPU and whether the "CPU contains a safety program" option is selected.

   ⚠WARNING

   In safety mode, access by means of the F-CPU password must not be authorized when making changes to the standard user program, since changes to the safety program can also be made. To rule out this possibility, you must configure Protection Level 1. If only one person is authorized to change the standard user program and the safety program, level of protection "2" or "3" 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).

2. Check the safety-related parameters of all configured F-I/O in the printout.

   The safety-related parameters are found under "Parameters – F-Parameters" and "Parameters – Module parameters".

   For F-I/O that you address via safety-related I-slave-slave communication, the parameters can be found in the printout of the hardware configuration of the station of the DP maser (see Chapter "Printed Project Data for the Hardware Configuration").

   For fail-safe DP standard slaves/standard I/O devices, the safety-related parameters are found under "PROFIsafe". In addition, note the documentation for the relevant fail-safe DP standard slave/standard I/O device regarding any other safety-related (technological) 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 "Parameter CRC (without F-addresses)" of the copied F-I/O, or "F_Par_CRC (without F-addresses)" in the case of fail-safe DP standard slaves/standard I/O devices, with the corresponding cyclic redundancy check of the already checked F-I/O. The "Parameter CRCs (without F-addresses)" can be found in the printout of the hardware configuration in the respective module description of the F-I/O.

3. Check that the PROFIsafe destination addresses are unique from one another.

⚠ WARNING

Rule for PROFIBUS subnets:

The PROFIsafe destination address and, thus, the switch setting on the address switch of the F-I/O must be unique network-wide* and station-wide** (system-wide). For S7-300 F-SMs and ET 200S, ET 200eco and ET 200pro F-modules, you can assign a maximum of 1022 different PROFIsafe destination addresses.

Exception: The F-I/O in different I-slaves may be assigned the same PROFIsafe destination address, as they are only addressed within the station, that is, by the F-CPU in the I-slave.

Rules for Ethernet subnets and hybrid configurations of PROFIBUS and Ethernet subnets:

The PROFIsafe destination address and, thus, the address switch setting on the F-I/O have to be unique only*** within the Ethernet subnet, including all lower-level PROFIBUS subnets, and station-wide** (system-wide). For S7-300 F-SMs and ET 200S, ET 200eco and ET 200pro F-modules, you can assign a maximum of 1022 different PROFIsafe destination addresses.

Exception: The F-I/O in different I-slaves may be assigned the same PROFIsafe destination address, as they are only addressed within the station, that is, by the F-CPU in the I-slave.

The networked nodes of an Ethernet subnet are characterized by having IP addresses with the same subnet address, i.e. the IP addresses match in the digits that have the value "1" in the subnet mask.

Example:

IP address: 140.80.0.2.

Subnet mask: 255.255.0.0 = 11111111.11111111.00000000.00000000

Meaning: Bytes 1 and 2 of the IP address define the subnet; subnet address = 140.80.

* A network consists of one or more subnets. "Network-wide" means beyond the boundaries of the subnet.

** The address is unique for a station configured in *HW Config* (for example, an S7-300 station or I-slave)

*** Across Ethernet subnets, excluding cyclic PROFINET IO communication (RT communication)

11.2.2 Safety Program Acceptance Test

Checking the Safety Program ("Safety Program" Print Content)

1. In the printout, check whether the two collective signatures match:
   - Collective signature of all F-blocks with F-attribute in the block container
   - Collective signature of the safety program
2. Check whether the version of *S7 Distributed Safety* used to create the printout (footer of the printout) is greater than or equal to the version used to compile the safety program (information section of the "Safety program" printout).
3. Check whether the version of *S7 Distributed Safety* used to compile the safety program (information section of the "Safety program" printout) corresponds to the version in Annex 1 of the Certification Report.
4. Check whether the signatures and initial value signatures of all F-application blocks and F-system blocks match the signatures specified in Annex 1 of the Certification Report.
5. Ensure that you have assigned a unique DP_DP_ID parameter throughout the network for all safety-related communication connections for safety-related master-master, master-I-slave, I-slave-I-slave and IO controller-IO controller communication.
6. Ensure that you have assigned a unique R_ID parameter throughout the network for all safety-related communication connections for safety-related communication via S7 connections.
7. Check to determine whether a validity check was programmed for all data in the safety program transferred from the standard user program.
8. Check the number of F-runtime groups in the safety program (maximum of 2) and whether all necessary F-blocks are present in the F-runtime group).
9. For each F-runtime group, check whether the following values in the F-runtime group information correspond to the values you configured:
   - Number of the F-CALL
   - Number of the called F-program block
   - Number of the associated instance DB, if applicable
   - Maximum cycle time of the F-runtime group
   - Number of the DB for F-runtime group communication, if applicable
10. Check the following For each F-I/O addressed in the F-runtime group:
    - Based on the start address of the F-I/O, check whether the symbolic name used in the safety program and the number of the F-I/O DB belong to the proper F-I/O.
    - Check whether the value of F_Monitoring_Time matches the corresponding value of the F-I/O with the same start address in the "Hardware configuration" printout (or "F_WD_Time" for fail-safe DP standard slaves/standard I/O devices).
    - Check whether PROFIsafe is in V2 mode when F-I/O are used on PROFINET IO or in a hybrid configuration on PROFIBUS DP and PROFINET IO based on IE/PB Links.
    - Check whether the type of passivation corresponds the value you configured.
11. Check the additional information:
    - Check whether the "Safety mode can be deactivated" setting corresponds the value you configured.
    - Check whether the printout of the project data is complete based on the total page count.

See also

Printing out project data (Page 279)

11.3 Checks after Downloading the Safety Program to the F-CPU

Introduction

You download the S7 program to the F-CPU as described in Chapter "Downloading the Safety Program". Afterwards, you perform the checks described below.

Note

The "Safety Program" dialog must be used to download F-blocks the last time prior to the acceptance test. Downloading the changes is not sufficient.

Checks after Downloading

1. Once the safety program has been downloaded to the F-CPU, check the following:
   - Check whether the online collective signature of all F-blocks with F-attribute in the block container match the collective signatures in the accepted offline printout.
   - Verify that the online safety program contains no unused F-CALLs.
   - Verify that no more than two F-CALL blocks are present in the F-CPU.

If this is not the case, check to determine whether you downloaded the safety program to the correct F-CPU and, if necessary, download the safety program again.

Note

In the case of recurring tests, you can determine whether the F-CPU contains the correct safety program by comparing the online collective signature of all F-blocks with F-attribute in the block container with the collective signature in the accepted offline printout.

If there is no programming device or PC with *S7 Distributed Safety* V5.4 available for recurring tests, you can read out the collective signature of the safety program from the F-shared DB using an operator control and monitoring system. You can obtain the address in the F-shared DB where the collective signature of the safety program is found ("F_PROG_SIG" variable) from the "Safety program" printout. This option should only be used if you do not have to perform a manipulation.

See also

Downloading the Safety Program (Page 260)

11.4 Acceptance Test of Changes

Introduction

For the acceptance test of changes, you must use the same procedure as for the initial acceptance test (see Chapter "Overview of System Acceptance Test").

For the acceptance test of changes, it is sufficient to check the following aspects of the hardware configuration and the F-blocks and to perform the following function test:

- Check the safety-related parameters of the changed or newly added F-I/O in the printout of the hardware configuration.
- Check the changed or newly added F-blocks in the printout of the F-blocks.
- Check whether the signatures and initial value signatures of the modified F-application blocks and F-system blocks in the printout of the safety program match the signatures specified in Annex 1 of the Certification Report.
- Perform a function test of the changes.

Basic Procedure for Determining Changes

To determine safety-related changes, compare the two collective signatures in the information section of the "Safety program" printout of the modified safety program undergoing acceptance testing with the signatures in the printout of the accepted safety program.

If the signatures are different, there is a safety-related change in the configuration of the F-CPU and/or F-I/O and/or in the safety program.

Detection of Safety-Related Changes in the Parameter Assignment of the F-I/O

To determine safety-related changes in the parameter assignment of the F-I/O addressed in the safety program, compare the parameter CRCs of all F-I/O in the "Addressed F-I/O" section of the "Safety program" printout with those in the printout of the accepted safety program.

If the "Parameter CRC" for an F-I/O is different, there is a safety-related change in the parameter assignment of this F-I/O, e.g., for PROFIsafe addresses.

In this case, also compare the "Parameter CRC" (without F-addresses)", or "F-Par-CRC (without F-addresses)" for fail-safe DP standard slaves/standard I/O devices, in the printout of the modified hardware configuration with the corresponding cyclic redundancy check in the printout of the accepted hardware configuration.

The printout of the hardware configuration contains this information in the relevant module description for the F-I/O. 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 PROFIsafe destination addresses of all configured F-I/O continue to be unique from all others.

If you want to determine safety-related changes in the parameter assignment of all configured F-I/O, you must compare the "Parameter CRC", or the "F-Par_CRC" for fail-safe DP standard slaves/standard I/O devices, directly in the printout of the hardware configuration.

Do not forget to compare the parameter CRCs of the F-I/O that you address via safety-related I-slave-slave communication.

Detection of Changes of the Start Addresses of F-I/O

To determine changes in the start addresses of the F-I/O addressed in the safety program, compare the start addresses of all F-I/O in the "Addressed F-I/O" section of the "Safety program" printout with those in the printout of the accepted safety program.

If you want to determine changes in the start addresses of all configured F-I/O, you must compare them directly in the printout of the hardware configuration.

Detection of Changes in the Safety Program

To determine changes in the safety program, compare the changed safety program offline with the saved accepted program (using the "Compare..." button in the "Safety Program" dialog).

This enables you to identify which F-blocks were changed. Determine the changes in the F-blocks you created (F-PBs, F-FBs, F-FCs, and F-DBs) by comparing the printouts ("Function Block Diagram/Ladder Diagram" print content).

Use of Software Packages in the Standard User Program

For software packages that can be used in parallel with the standard program and safety program (for example, SW Redundancy), general conditions may apply that must be observed:

Note

If the safety program occupies block numbers (for FBs, DBs, and FCs) that are required by the software package, it may be necessary to change the safety program to release the block numbers for subsequent use of the software package. This requires another acceptance test for the changes in the safety program.

See also

Downloading the Safety Program (Page 260)

Comparing Safety Programs (Page 273)

Testing the Safety Program (Page 289)

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.

Using Simulation Devices / Simulation Programs

⚠WARNING

If you operate simulation devices or simulation programs that generate safety message frames, e.g., based on PROFIsafe, and make them available to an S7 Distributed Safety F-system via the bus system (such as PROFIBUS DP or PROFINET IO), you have to ensure the safety of the F-system using organizational measures, such as operational monitoring and manual safety shutdown.

If you use the S7-PLCSIM function of STEP 7 to simulate safety programs, these measures are not necessary because S7-PLCSIM cannot establish an online connection to a real S7 component.

Note, for example, that a protocol analyzer may not perform functions that reproduce recorded frame sequences with correct time behavior.

STOP by Means of Programming Device or PC, Mode Selector, or Communication Function

⚠WARNING

Switching from STOP to RUN mode using a programming device or PC interface, mode selector, or communication function is not interlocked. 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.

Therefore, always switch off the F-CPU directly at the device when performing maintenance work.

F-CPU Stop Initiated by SFC 46 "STP"

⚠ WARNING
A STOP state initiated by SFC46 "STP" can be canceled very easily (and unintentionally) from the programming device or PC. For this reason, an F-CPU STOP initiated by SFC46 is not a fail-safe STOP.

See also

Programming Startup Protection (Page 94)

Overview of Testing the Safety Program (Page 285)

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 observe the notes regarding upward and downward compatibility in the documentation and readme files for these products.

Replacement of Hardware Components

Hardware components for S7 Distributed Safety (F-CPU, F-I/O, batteries, etc.) are replaced in the same way as in a standard automation system.

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 while in service can cause a communication error in the F-CPU.

You must acknowledge the communication error in your safety program in the ACK_REI variable of the F-I/O DB. Otherwise, the F-I/O will remain passivated.

CPU Operating System Update

Check of the CPU operating for F-validity: When using a new CPU operating system (operating system 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 annex of the Certification Report. This information and any notes on the new CPU operating system must be taken into account.

Operating System Update for Interface Module

When using a new operating system for an interface module, e.g., IM 151-1 HIGH FEATURE of ET 200S (operating system update, see online Help for STEP 7), you must observe the following:

If the "Activate firmware after download" check box is selected for the operating system update, the IM will be automatically reset following a successful loading operation and will then run on the new operating system. The entire F-I/O is passivated after startup of the IM. The F-I/O is reintegrated in the same way as when a communication error occurs, that is, an acknowledgment in the ACK_REI variable of the F-I/O DB is required.

Preventive Maintenance (Proof Test)

The probability values for the certified F-system components guarantee a proof-test interval of 10 years for ordinary configurations. For detailed information, refer to the F-I/O manuals. Proof test for complex electronic components generally means replacement with unused items. If for particular reasons you require a proof-test interval in excess of 10 years, contact your Siemens representative.

As a rule, a shorter proof-test interval is required for sensors and actuators.

Removing *S7 Distributed Safety*

To remove the software, see Chapter "Installing/Removing *S7 Distributed Safety* V5.4 SP4 Optional Package".

F-system hardware is removed and disposed of in the same way as with standard automation systems; refer to the appropriate *hardware manuals*.

See also

Installing/Removing the S7 Distributed Safety V5.4 SP4 Optional Package (Page 17)

F-I/O Access (Page 95)

12.3 Guide to Diagnostics

Introduction

This chapter presents 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 one recommendation.

Steps for Evaluating Diagnostic Capabilities

Step	Procedure	Reference
1	**Evaluating LEDs on the hardware (F-CPU, F-I/O):** • BUSF LED on the F-CPU: flashes when a communication error occurs on PROFIBUS DP/PROFINET IO; when OB85 and OB121 are programmed, illuminates when a programming error occurs (e.g., instance DB is not loaded) • STP LED on the F-CPU: illuminates when the F-CPU is in STOP mode • Fault LEDs on the F-I/O: SF-LED (group error LED) illuminates if any fault occurs in the individual F-I/O	F-CPU and F-I/O manuals
2	**Evaluating diagnostic buffer in STEP 7:** In *HW Config,* read out the diagnostic buffer for the modules (F-CPU, F-I/O, CPs) using the **PLC > Module Information** menu command	*STEP 7 online help* and F-CPU and F-I/O manuals
3	**Evaluating stacks in STEP 7:** If the F-CPU is in STOP mode, read out the following in consecutive order in *HW Config* using the **PLC > Module Information** menu command: • B stack: Check whether STOP mode of the F-CPU was triggered by an F-block of the safety program • U stack • L stack	*Online Help for STEP 7*
4	**Evaluating diagnostic variable of the F-I/O DB using testing and commissioning functions or in the standard user program:** Evaluate the DIAG variable in the F-I/O DB	Chapter "F-I/O Access"

Step	Procedure	Reference
5	**Evaluating diagnostic parameters of instance DBs of F-application blocks using testing and commissioning functions or in the standard user program:** • Evaluate the following for F_MUTING, F_1oo2DI, F_2H_EN, F_MUT_P, F_ESTOP1, F_FDBBACK, F_SFDOOR in the assigned instance DB: – DIAG parameter • Evaluate the following for F_SENDDP or F_RCVDP in the assigned instance DB: – RETVAL14 parameter – RETVAL15 parameter – DIAG parameter • Evaluate the following for F_SENDS7 or F_RCVS7 in the assigned instance DB: – STAT_RCV parameter – STAT_SND parameter – DIAG parameter	Chapter on relevant F-application block

Evaluation of the Diagnostic Variable or Parameters of F-I/O DBs or Instance DBs

Note

The following diagnostic variables/parameters provide you with detailed diagnostic information: DIAG, RETVAL14, RETVAL15, STAT_RCV, and STAT_SND. These can be read out using the testing and commissioning functions on the programming device or using an operator control and monitoring system, or they can be evaluated in your standard user program.

These parameters must not be accessed in the safety program.

Evaluation of Diagnostic Variable or Parameters in the Standard User Program

Do not evaluate the diagnostic variable or parameters in the safety program, rather, use the following procedure:

1. Load the diagnostic information of the above-mentioned variables/parameters from the F-I/O DB or the corresponding instance DB with fully qualified DB access into your standard user program (example for F-I/O DB: L "F00005_4_8_F_DI_DC24V".DIAG). If necessary, assign a symbolic name for the instance DB in the symbol table.
2. Place the diagnostic information in your standard user program, e.g., in a bit memory address area using the "T MB x" instruction.
3. You could then evaluate the individual bits of the diagnostic information in your standard user program, that is, M x.y in this example.

Tip on RETVAL14 and 15

The diagnostic information contained in the RETVAL14 and RETVAL15 parameters corresponds to that of SFC14 and SFC15. For a description, refer to the *Online Help for STEP 7* on SFC14 and SFC15.

Tip on STAT_RCV and STAT_SND

The diagnostic information contained in the STAT_RCV parameter corresponds to the diagnostic information contained in the STATUS parameter of SFB9/FB9. The diagnostic information contained in the STAT_SND parameter corresponds to the diagnostic information contained in the STATUS parameter of SFB8/FB8. For a description, refer to the *Online Help for STEP 7* on SFB8 and SFB9.

See also

F-I/O Access (Page 95)

Checklist

A

A.1 Checklist

Life Cycle of Fail-Safe Automation Systems

The table below contains a checklist summarizing all activities in the life cycle of a fail-safe S7 Distributed Safety system, including requirements and rules that must be observed in the various phases.

Checklist

Key:

- Stand-alone chapter references refer to this documentation.
- "*SM*" stands for the *Safety Engineering in SIMATIC S7* system manual.
- "*F-SMs Manual*" stands for the *Automation System S7-300, Fail-Safe Signal Modules* manual.
- "*F-Modules Manual*" stands for the *ET 200S Distributed I/O System, Fail-Safe Modules* manual.
- *"ET 200eco Manual"* stands for the *ET 200eco Distributed I/O Station, Fail-Safe I/O Module* manual.
- *"ET 200pro Manual"* stands for the *ET 200pro Distributed I/O Station, Fail-Safe Modules* manual.

Phase	Requirement/Rule	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	Chapter "Product Overview"; *SM*, Chapters 1.5, 2.4	
Selection of sensors and actuators	Requirements for actuators	*F-SMs Manual*, Chapter 6.5; *F-Modules Manual*, Chapter 4.5; *ET 200eco Manual*, Chapter 5.5 *ET 200pro Manual*, Chapter 4.4	

Phase	Requirement/Rule	Reference	Check
Specification of required safety properties for individual components	• DIN V 19 250 • IEC 61508	*SM*, Chapters 4.7, 4.8	
Configuration			
Installing the optional package	Requirements for installation	Chapter "Installing/Removing..."	
Selection of S7 components	Rules for configuration	Chapter "Hardware and Software Components"; *SM*, Chapter 2.4; *F-SMs Manual*, Chapter 3; *F-Modules Manual*, Chapter 3; *ET 200eco Manual*, Chapter 3 *ET 200pro Manual*, Chapter 2	
Configuration of hardware	• Rules for F-systems • Verification of utilized hardware components based on Annex 1 of Certification Report	Chapter "Overview of Configuration, Particularities for Configuring..."; Annex 1 of Certification Report	
Configuration of F-CPU	• Protection level, "CPU contains safety program" • Password • Define/set F-specific parameters • Call 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	Chapter "Configuring the F-CPU"; S7-300 standard; S7-400 standard; IM 151-7 CPU	
Configuration of F-I/O	• Settings for safety mode • Configure monitoring times • Define type of sensor interconnection/evaluation • Define diagnostic behavior • Assign symbolic names	Chapter "Configuring the F-I/O" and subsequent chapters; *SM*, Appendix A; *F-SMs Manual*, Chapters 3, 9, 10; *F-Modules Manual*, Chapters 2.4, 7; *ET 200eco Manual*, Chapters 3, 8; *ET 200pro Manual,* Chapters 2.4, 8	
Saving, compiling, and loading of hardware configuration	• System data are generated • F-shared DB, F-system blocks, and F-I/O DBs are generated	–	
Programming			
Define program design and structure	• Observe warnings and notes on programming • Verify software components used with Annex 1 of Certification Report	Chapter "Overview of Programming, Program Structure..., Defining the Program Structure; Programming a Startup Protection; Annex 1 of Certification Report	

Phase	Requirement/Rule	Reference	Check
Creating/inserting the F-blocks	• Generate, edit, and save F-FBs, F-FCs, and F-DBs in accordance with the requirements of the program structure • Rules for: – F-I/O access – Passivation and reintegration of F-I/O – Inserting F-blocks from Distributed Safety F-library (V1) and user-created F-libraries – Safety-related CPU-CPU communication – Communication with the standard user program	Chapter "Creating F-Blocks in F-FBD/F-LAD, Distributed Safety F-Library (V1)" Chapter "F-I/O Access" Chapter "Implementation of User Acknowledgment" Chapter "F-Libraries" Chapter "Configuring and Programming Communication" Chapter "Data Exchange between Standard User Programs and Safety Program"	
Creating the F-runtime groups	• Create F-CALL • Assign F-FB/F-FC to F-CALL • Set maximum cycle time for the F-runtime group in accordance with requirements (dependent on process and safety regulations) • Create DB for F-runtime group communication	Chapter "Defining F-Runtime Groups"; *SM*, Appendix A	
Compiling the safety program		Chapter "Compiling a Safety Program"	
Implementing call of safety program	Call of F-CALL blocks directly in OBs (e.g., OB35), FBs, or FCs	Chapter "Defining F-Runtime Groups"	
Installation			
Hardware configuration	• Rules for mounting • Rules for wiring	Chapter "Overview of Configuration, Particularities for Configuring..."; *F-SMs Manual*, Chapters 5, 6; *F-Modules Manual*, Chapters 3, 4; *ET 200eco Manual*, Chapters 3, 4; *ET 200pro Manual,* Chapters 2, 3	
Commissioning, Testing			
Powering up	Rules for commissioning – same as for standard system	S7-300 standard; S7-400 standard	
Downloading safety program and standard user program	• Rules for downloading • Rules for program identification • Comparing safety programs	Chapter "Downloading the Safety Program" Chapter "Comparing Safety Programs"	
Testing safety program	• Rules for deactivating safety mode • Procedures for changing safety program data	Chapter "Function Test of Safety Program or Protection through Program Identification; Testing of Safety Program; Deactivating Safety Mode"	
Changing the safety program	• Rules for deactivating safety mode • Rules for modifying the safety program	Chapter "Modifying the Safety Program in RUN Mode, Deactivating Safety Mode, Deleting the Safety Program"	

Phase	Requirement/Rule	Reference	Check
Testing the safety-related parameters	Rules for configuration	Chapter "Printing Out Project Data of the Safety Program"; *F-SMs Manual*, Chapters 4, 9, 10; *F-Modules Manual*, Chapters 2.4, 7; *ET 200eco Manual*, Chapters 3, 8; *ET 200pro Manual,* Chapters 2.4, 8	
Acceptance test	• Rules and notes on the acceptance test • Printouts	Chapter "System Acceptance Test"	
Operation, Maintenance			
General operation	Notes on operation	Chapter "Notes on Safety Mode..."	
Access protection		Chapter "Access Protection"	
Diagnostics	Responses to faults and events	Chapter "Guide to Diagnostics"	
Replacement of software and hardware components	• Rules for module replacement • Rules for updating the operating system of the F-CPU – same as for standard system • Rules for updating software components • Notes on IM operating system update • Notes on preventive maintenance	Chapter "Replacing Software and Hardware Components, F-I/O Access"; Online Help for *STEP 7*	
Removing, disassembly	• Notes for removing software components • Notes for disassembling modules	Chapter "Installing/Removing..., Replacing Software and Hardware Components"	

See also

Overview (Page 13)
Installing/Removing the S7 Distributed Safety V5.4 SP4 Optional Package (Page 17)
Overview of Configuration (Page 23)
Particularities for Configuring the F-System (Page 25)
Configuring the F-CPU (Page 26)
Configuring the F-I/O (Page 36)
Overview of Access Protection (Page 47)
Overview of Programming (Page 55)
Structure of the Safety Program in S7 Distributed Safety (Page 57)
Defining the Program Structure (Page 74)
Creating F-Blocks in FFBD/FLAD (Page 76)
Rules for F-Run-Time Groups of the Safety Program (Page 86)
F-I/O Access (Page 95)
Overview of Distributed Safety F-Library (V1) (Page 175)
Custom F-Libraries (Page 251)
Compiling Safety Program (Page 258)
Downloading the Safety Program (Page 260)
Modifying the safety program in RUN mode (Page 271)
Comparing Safety Programs (Page 273)
Deleting the Safety Program (Page 277)
Printing out project data (Page 279)
Deactivating Safety Mode (Page 286)
Testing the Safety Program (Page 289)
Overview of System Acceptance Test (Page 293)
Notes on Safety Mode of the Safety Program (Page 301)
Replacing Software and Hardware Components (Page 302)
Guide to Diagnostics (Page 304)

Glossary

Access protection

-> Fail-safe systems must be protected against dangerous, unauthorized access. Access security is implemented in F-Systems by assigning two passwords (one for the -> F-CPU, and one for the -> safety program).

Automatically Generated F-Blocks

-> F-blocks that are created and if necessary launched automatically when the -> safety program is generated in order to produce an executable safety program from the user programmed safety program.

Category

Category as defined by EN 954-01

S7 Distributed Safety can be used in -> safety mode up to Category 4.

Channel Fault

Channel-related fault, such as a wire break or short circuit.

Collective Signatures

The collective 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 preliminary acceptance test of the safety program, e.g., by -> experts.

The following signatures are displayed by the programming software and can also be printed out:

- Collective signature of all F-blocks with F-attribute of the block container
- Collective signature of the safety program

These two signatures must match for the acceptance test.

CRC

Cyclic Redundancy Check -> CRC Signature

CRC Signature

The validity of the process data in the -> safety message frame, the accuracy of the assigned address references, and the safety-related parameters are ensured by means of a CRC signature contained in the safety message frame.

Custom F-Libraries

User-created F-libraries are F-libraries created by the user containing F-FBs, F-FCs, and application templates (network templates).

DB for F-Runtime Group Communication

-> F-DB for safety-related communication between F-runtime groups of a safety program.

Deactivated Safety Mode

Deactivated safety mode is the temporary deactivation of -> safety mode for test purposes, commissioning, etc.

The following actions are possible only in deactivated 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.

Depassivation

-> Reintegration

Discrepancy Analysis

Discrepancy analysis for equivalence or nonequivalence is used for fail-safe inputs to determine errors based on 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 (for non-equivalence testing, when the same levels are detected). The signals are checked to establish whether the difference (when checking for non-quality: has disappeared after the so called -> discrepancy time has expired. 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

Discrepancy time is a period of time configured for the -> discrepancy analysis. If the discrepancy time is set too high, the fault detection time and -> fault reaction time are extended 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 that is required in S7 Distributed Safety for master-master communication between -> safety programs in different -> F-CPUs.
Two F-CPUs (minimum) are involved in safety-related master-master communication via a DP/DP coupler. Each F-CPU is linked to the DP/DP coupler by means of its PROFIBUS DP interface.

Expert

A system is generally approved, that is, the safety acceptance test of the system is usually carried out by an independent expert (for example, from TÜV).

Fail-safe DP standard slaves

Fail-safe DP standard slaves are standard slaves that are operated on PROFIBUS with the DP protocol. They must behave in accordance with IEC 61784-1:2002 Ed1 CP 3/1 and the PROFIsafe profile. A GSD file is used to configure fail-safe DP standard slaves.

Fail-Safe I/O Modules

ET 200eco modules are fail-safe I/O modules that can be used for safety-related operation (in -> safety mode). These modules feature integrated -> safety functions. They behave according to IEC 61784-1:2002 Ed1 CP 3/1 and the PROFIsafe bus profile.

Fail-safe I/O standard devices

Fail-safe standard I/O devices are standard devices that are operated on PROFINET with the IO protocol. They must behave in accordance with IEC 61784-1:2002 Ed1 CP 3/3 and the PROFIsafe bus profile in the V2 mode. A GSD file is used to configure fail-safe DP standard slaves.

Fail-Safe Modules

ET 200S and ET 200pro modules that can be used for safety-related operation (in -> safety mode) in the ET 200S and ET 200pro distributed I/O systems. These modules are equipped with integrated -> safety functions. They behave according to IEC 61784-1:2002 Ed1 CP 3/1 and 3/3 and 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.

F-application blocks

Block container for the *Distributed Safety* F-library containing the -> F-application blocks.

F-Application Blocks

F-application blocks are F-blocks (F-FBs, F-FCs) with ready made functions in the *Distributed Safety* F-library. The F-application blocks can be called by the user in the -> F-PB and in additional -> F-FBs and -> F-FCs.

F-attribute

All -> F-blocks in a -> safety program are provided an F-attribute (identified in the "Safety Program" dialog box by an "F" in the F-block symbol). Only the blocks of the -> safety program have the F-attribute after the -> safety program is successfully compiled.

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:

- Blocks created by the user in programming languages -> F-FBD/F-LAD, F-CALL, and F-DB
- Blocks selected by the user from an F-library
- Blocks automatically added in the -> safety program (-> F-SBs, -> automatically generated F-blocks, -> F-shared DB)

All F-blocks are depicted with a yellow background in the "Safety Program" dialog box and *SIMATIC Manager*.

F-CALL

F-CALL is the "F-call block" for the -> safety program in S7 Distributed Safety.

F-CALL is created by the user as a function in the "F-CALL" programming language and cannot be edited. F-CALL calls the -> F-runtime group from the -> standard user program. It contains a call for the -> F-PB and calls for the F-blocks (-> F-SBs, -> automatically generated F-blocks, -> F-shared DB) of the F-runtime group that were automatically added.

F-Communication DBs

F-communication DBs are fail-safe data blocks for safety-related CPU-CPU communication via S7 connections.

F-CPU

An F-CPU is a central processing unit with fail-safe capability that is permitted for use in S7 Distributed Safety and in which a -> safety program can run in addition to the -> standard user program.

F-DBs

Optional fail-safe data blocks with read/write access from anywhere within the safety program (exception: DBs for F-runtime group communication)

F-FBD

F-FBD is a programming language for -> safety programs in S7 Distributed Safety. The standard *FBD/LAD Editor* in *STEP 7* is used for programming.

F-FBs

F-FBs are fail-safe function blocks (with instance DBs) in which the user programs the -> safety program in -> F-FBD or -> F-LAD.

F-FCs

F-FCs are fail-safe FCs in which the user programs the -> safety program in -> F-FBD or -> F-LAD.

F-I/O

F-I/O is a group designation for fail-safe inputs and outputs available in *SIMATIC S7* for integration in S7 Distributed Safety, among others. The following F-I/O modules are available for S7 Distributed Safety:

- ET 200eco fail-safe I/O module
- S7-300 fail-safe signal modules (-> F-SMs)
- -> Fail-safe modules for ET 200S
- -> Fail-safe modules for ET 200pro
- -> Fail-safe DP standard slaves
- -> Fail-safe standard I/O devices

F-I/O DB

An F-I/O DB is a fail-safe data block for an -> F-I/O in S7 Distributed Safety. An F-I/O DB is automatically created for each F-I/O during compilation in *HW Config*. The F-I/O data block contains variables that the user can evaluate in the safety program, or that he can or must write to as follows:

- For reintegration of F-I/O after communication errors, F-I/O faults, or channel faults
- If F-I/O should be passivated as a result of particular safety program conditions (for example, group passivation)
- For reassignment of parameters for fail-safe DP standard slaves
- In order to evaluate whether fail-safe values or process data are output

F-I/O faults

An F-I/O fault is a module related fault for F-I/O, such as a communication error or a parameter assignment error

F-LAD

-> F-FBD

F-modules

-> Fail-safe modules

F-PB

The F-PB is the "introductory fail-safe block" for fail-safe programming of the -> safety program in S7 Distributed Safety. The F-PB is an -> F-FB or -> F-FC that the user assigns to the -> F-CALL of an -> F-runtime group.

The F-PB contains the F-FBD or F-LAD safety program, any calls of additional -> F-FBs/F-FCs for program structuring, and any F-application blocks from the block container of -> F-application blocks of the *Distributed Safety* F-library and F-blocks from -> user-created F-libraries.

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:

-> F-CALL, -> F-PB, -> F-FBs/ -> F-FCs (if applicable), -> F-DBs (if applicable), -> F-I/O DBs, F-blocks of *Distributed Safety* F-library and user-created F-libraries, instance DBs, -> F-SBs, and -> automatically generated F-blocks.

F-SBs

F-SBs are fail-safe system blocks which 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

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. When the hardware configuration is saved and compiled in HW Config, the F-shared DB is automatically inserted and expanded. Using its symbolic name F_GLOBDB, the user can evaluate certain data of the -> safety program.

F-SMs

F-SMs are S7-300 fail-safe signal modules that can be used for safety-related operation (in -> safety mode) as centralized modules in an S7 300 or as distributed modules in the ET 200M distributed I/O system. F-SMs are equipped with integrated -> safety functions.

F-system blocks

Block container of *Distributed Safety* F-library containing -> F-SBs and the -> F-shared-DB.

F-systems

-> Fail-safe systems

i-Parameter

Individual parameter of -> fail-safe DP standard slaves

MSR

Instrumentation and control technology

Non-equivalent Sensor

A non-equivalent sensor is a two-way switch that is connected in -> fail-safe systems (two-channel) to two inputs of an -> F-I/O module (for 1oo2 evaluation of sensor signals; -> sensor evaluation).

Passivation

When passivation occurs in a -> F-I/O module with inputs the -> F-system provides fail-safe values (0) for the safety program instead of the process data pending in the PII at the fail-safe inputs.

When passivation occurs in a F-I/O module 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.

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.

PROFIsafe Address

Every -> F-I/O module has a PROFIsafe address. You must configure the PROFIsafe address in *HW Config* of STEP 7 and set the address via a switch on the F-I/O.

Program Signature

-> Collective signature

Proof Test Interval

A component must be put into fail-free state following the proof-test interval. That is, it is replaced by an unused component or it is proven to be completely error-free.

Reintegration

Switching from substitute values (0) to process data (reintegration of a -> F-I/O module) occurs automatically or after user acknowledgment in the F-I/O DB. The reintegration method depends on the following:

- The cause for -> passivation of the F-I/O or channels of the F-I/O
- Parameter assignment in the -> F-I/O DB

For an -> F-I/O module with inputs, the process data in the PII pending at the F-inputs are provided again for the safety program after reintegration. The F-System transfers the PIO output values provided in the safety program to the fail-safe outputs of the F-I/O.

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 entirely in STEP 7, you do not require any hardware (CPU, I/O).

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, the value is always "0".

Safety Function

Safety function is a mechanism built into the -> F-CPU and -> F-I/O that allows them to be used in -> fail-safe systems.

In accordance with IEC 61508: safety functions are implemented by a safety system in order to maintain the system in a -> safe state or to place it in a safe state in the event of a particular error. (-> user safety function)

Safety Integrity Level

Safety Integrity Level (SIL) is the safety level defined in IEC 61508 and prEN 50129. The higher the Safety Integrity Level is, the more stringent the actions are for avoiding and controlling system faults and random hardware failures.

S7 Distributed Safety can be used in safety mode up to SIL3.

Safety Message Frame

In -> safety mode, data is 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. Safety mode is the operating mode of the -> F-I/O that allows -> safety-related communication by means of a -> safety frame.
2. Operating mode of the safety program. In safety mode of the safety program, all safety mechanisms for fault detection and reaction are activated. In safety mode, the safety program cannot be modified during operation. Safety mode can be deactivated by the user (-> deactivated safety mode).

Safety Program

The safety program is a 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 in once
- 1oo2 evaluation - sensor signal is read in twice by the same ->F-I/O and compared internally

Signature

-> Collective signatures

Standard Communication

Standard communication is used to exchange non-safety-related data.

Standard Mode

Standard mode is the operating mode of -> F-I/O in which -> safety-related communication by means of -> safety frames is not possible; only -> standard communication is possible in this operating mode.

Standard User Program

The standard user program is a non-safety-related user program.

Startup of F-system

When an -> F-CPU switches from STOP to RUN mode, the -> standard user program is started as usual. When the -> safety program is started, all data blocks with -> F-attribute are initialized with values from the load memory (as with a cold start). This means that saved error information is lost.

The -> F-system automatically performs -> 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 a fault, and the F-system can no longer execute its actual user safety function, it will execute the fault reaction function: for example, the associated outputs are deactivated and the F-CPU switches to STOP mode if necessary.

Voltage Group

In the ET 200S and ET 200pro distributed I/O systems: A voltage group is a group of electronic modules supplied by one power module.

Index

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

SIEMENS

Siemens AG

A&D AS SM ID
Postfach 1963
D-92209 Amberg

Telefax: +49(9621)80-3103
mailto:doku.automation@siemens.com

Your Address:

Name:
Company:
Position:
Street:
Postal code / Place:
Email:
Phone:
Fax:

Your Feedback as regards the S7 Distributed Safety

Dear SIMATIC user,

Our goal is to provide you information with a high degree of quality and usability, and to continuously improve the SIMATIC documentation for you. To achieve this goal, we require your feedback and suggestions. Please take a few minutes to fill out this questionnaire and return it to me by Fax, e-mail or by post.

We are giving out three presents every month in a raffle among the senders. Which present would you like to have?

☐ **SIMATIC Manual Collection** ☐ **Automation Value Card** ☐ **Laser pointer**

Dr. Thomas Rubach,
Head of Information & Documentation

General Questions

1. Are you familiar with the SIMATIC Manual Collection?

☐ yes ☐ no

2. Have you ever downloaded manuals from the internet?

☐ yes ☐ no

3. Do you use Getting Starteds?

☐ yes ☐ no

if yes, which:

4. How much experience do you have with the S7 Distributed Safety?

☐ Expert
☐ Experienced user
☐ Advanced user
☐ Beginner

Please specify the documents, for which you want to answer the questions below:

- ☐ **A: Manual S7 Distributed Safety, Configuring and Programming**
- ☐ **B: Manual S7-300, Fail-Safe Signal Modules**
- ☐ **C: Manual ET 200S, Distributed I/O System Fail-Safe Modules**
- ☐ **D: Manual ET 200eco, Distributed I/O Fail-Safe I/O Module**
- ☐ **E: System Description Safety Engineering in SIMATIC S7**
- ☐ **F: Getting Started S7 Distributed Safety**
- ☐ **G: ET 200pro Distributed I/O Device - Fail-Safe Modules**

1. **In which project phase do you use this document frequently?**

☐ Information ☐ Assembly
☐ Planning ☐ Commissioning
☐ Configuration ☐ Maintenance & Service
☐ Programming ☐ others:

2. **Finding the required information in the document:**

- How quickly can you find the desired information in the document?

☐ immediately ☐ not at all
☐ after a brief search ☐ after a long search

- Which search method do you prefer?

☐ Table of contents ☐ Index
☐ Full-text search ☐ others:

- Which supplements/improvements would you like in order to help you find the required information quickly?

3. **Your judgement of the document as regards content.**

- How satisfied are you with this document

☐ Totally satisfied ☐ not very satisfied
☐ Very satisfied ☐ not satisfied
☐ Satisfied

- Were able to find the required information?

☐ yes ☐ no

which was not:

4. **What is the scope of the information?**

☐ Just right
☐ Not enough - which topic:
☐ Too detailed – which topic:

5. **Is the information easy to understand (texts, figures, tables)?**

☐ yes ☐ no

if no, which was not:

6. **Are examples important to you?**

☐ no, of less importance
☐ yes, important –were the examples enough?

☐ yes ☐ no

if no, on which topic:

7. **What are your suggestions as regards the contents of the document?**

Thank you for your cooperation