SIEMENS

SIMATIC

S7 Distributed Safety Configuring and Programming

Manual

Contents

Preface 1

Product Overview 2

Configuration 3

Access Protection 4

Programming 5

F-Libraries 6

System Acceptance Test 7

Operation and Maintenance 8

Appendices

Checklist 9

Glossary 10

Index

Edition 10/2004
A5E00109537-02

Safety Guidelines

This manual contains notices intended to ensure personal safety, as well as to protect the products and connected equipment against damage. These notices are highlighted by the symbols shown below and graded according to severity by the following texts:

Danger
indicates that death, severe personal injury or substantial property damage will result if proper precautions are not taken.

Warning
indicates that death, severe personal injury or substantial property damage can result if proper precautions are not taken.

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

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

Notice
draws your attention to particularly important information on the product, handling the product, or to a particular part of the documentation.

Qualified Personnel

Only **qualified personnel** should be allowed to install and work on this equipment. Qualified persons are defined as persons who are authorized to commission, to ground and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

Correct Usage

Note the following:

Warning
This device and its components 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.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

Trademarks

SIMATIC®, SIMATIC HMI® and SIMATIC NET® are registered trademarks of SIEMENS AG.

Third parties using for their own purposes any other names in this document which refer to trademarks might infringe upon the rights of the trademark owners.

Copyright Siemens AG 2004 All rights reserved

The reproduction, transmission or use of this document or its contents is not permitted without express written authority. Offenders will be liable for damages. All rights, including rights created by patent grant or registration of a utility model or design, are reserved.

Siemens AG
Bereich Automation and Drives
Geschaeftsgebiet Industrial Automation Systems
Postfach 4848, D- 90327 Nuernberg

Disclaimer of Liability

We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

Siemens AG 2004
Technical data subject to change.

Contents

1 Preface 1-1

2 Product Overview 2-1

2.1 Overview 2-1
2.2 Hardware and Software Components 2-3
2.3 Installing/Removing the *S7 Distributed Safety* V 5.3 Optional Package 2-5

3 Configuration 3-1

3.1 Overview of Configuration 3-1
3.2 Particularities for Configuring the F-System 3-3
3.3 Configuring the F-CPU 3-4
3.4 Configuring the F-I/O 3-13
3.5 Configuring Fail-Safe DP Standard Slaves 3-17
3.6 Assigning Symbolic Names 3-20
3.7 Configuring Safety-Related CPU-CPU Communication 3-22
3.7.1 Configuring Safety-Related Master-Master Communication 3-23
3.7.2 Configuring Safety-Related Master-I-Slave Communication 3-27
3.7.3 Configuring Safety-Related I-Slave-I-Slave Communication 3-32
3.7.4 Configuring Safety-Related Communication via S7 Connections 3-38

4 Access Protection 4-1

4.1 Overview of Access Protection 4-1
4.2 Setting Access Permission for the Safety Program 4-3
4.3 Setting Access Permission for the F-CPU 4-6

5 Programming 5-1

5.1 Overview of Programming 5-1
5.1.1 Structure of Safety Program in S7 Distributed Safety 5-3
5.1.2 Fail-Safe Blocks 5-5
5.1.3 Differences between the F-FBD and F-LAD Programming Languages and the Standard FBD and LAD Programming Languages 5-7
5.2 Creating the Safety Program 5-19
5.2.1 Basic Procedure for Creating the Safety Program 5-19
5.2.2 Defining the Program Structure 5-21
5.3 F-I/O Access 5-23
5.3.1 Process Data or Fail-Safe Values 5-25
5.3.2 F- I/O DB 5-26
5.3.3 Accessing Variables of F-I/O DB 5-31
5.3.4 Passivation and Reintegration of F-I/O after F-System Startup 5-32
5.3.5 Passivation and Reintegration of F-I/O after Communication Errors 5-33
5.3.6 Passivation and Reintegration of F-I/O after F-I/O Faults and Channel Faults 5-36
5.3.7 Group Passivation 5-39
5.3.8 Implementing User Acknowledgment in Safety Program of F-CPU of DP Master 5-41

5.3.9 Implementing User Acknowledgment in Safety Program of F-CPU of I-Slave ... 5-44
5.4 Programming Safety-Related CPU-CPU Communication ... 5-48
5.4.1 Programming Safety-Related Master-Master Communication ... 5-49
5.4.2 Programming Safety-Related Master-I-Slave Communication and I-Slave-I-Slave Communication ... 5-54
5.4.3 Programming Safety-Related CPU-CPU Communication via S7 Connections ... 5-60
5.5 Communication between Standard User Program and Safety Program ... 5-67
5.6 Creating F-Blocks in F-FBD/F-LAD ... 5-71
5.6.1 Creating and Editing an F-FB/F-FC ... 5-72
5.6.2 Creating and Editing an F-DB ... 5-75
5.6.3 Know-How Protection for User-Created F-FBs and F-FCs ... 5-77
5.6.4 Defining F-Runtime Groups ... 5-80
5.7 Programming Startup Protection ... 5-88
5.8 Compiling and Downloading the Safety Program ... 5-89
5.8.1 "Safety Program" Dialog ... 5-89
5.8.2 Safety Program States ... 5-93
5.8.3 Compiling the Safety Program ... 5-94
5.8.4 Downloading the Safety Program ... 5-96
5.8.5 RAM Requirement for Safety Program ... 5-100
5.8.6 Function Test of Safety Program and Protection through Program Identification ... 5-102
5.9 Testing the Safety Program ... 5-106
5.9.1 Deactivating Safety Mode ... 5-106
5.9.2 Testing the Safety Program ... 5-110
5.10 Modifying the Safety Program ... 5-114
5.10.1 Modifying the Safety Program in RUN Mode ... 5-114
5.10.2 Comparing Safety Programs ... 5-116
5.10.3 Deleting the Safety Program ... 5-119
5.11 Printing the Project Data of the Safety Program ... 5-121

6 F-Libraries ... 6-1

6.1 Distributed Safety F-Library (V1) ... 6-1
6.1.1 Overview of Distributed Safety F-Library (V1) ... 6-1
6.1.2 F-Application Blocks ... 6-2
6.1.2.1 FB 179 "F_SCA_I": Scale Values of Data Type INT ... 6-5
6.1.2.2 FB 181 "F_CTU": Count Up ... 6-7
6.1.2.3 FB 182 "F_CTD": Count Down ... 6-8
6.1.2.4 FB 183 "F_CTUD": Count Up and Down ... 6-9
6.1.2.5 FB 184 "F_TP": Create a Pulse ... 6-11
6.1.2.6 FB 185 "F_TON": Create ON Delay ... 6-13
6.1.2.7 FB 186 "F_TOF": Create OFF Delay ... 6-15
6.1.2.8 FB 187 "F_ACK_OP": Fail-Safe Acknowledgment ... 6-17
6.1.2.9 FB 188 "F_2HAND": Two-Hand Monitoring ... 6-19
6.1.2.10 FB 189 "F_MUTING": Muting ... 6-21
6.1.2.11 FB 190 "F_1oo2DI": 1oo2 Evaluation with Discrepancy Analysis ... 6-31
6.1.2.12 FB 211 "F_2H_EN": Two-Hand Monitoring with Enable ... 6-36
6.1.2.13 FB 212 "F_MUT_P": Parallel Muting ... 6-39
6.1.2.14 FB 215 "F_ESTOP1": Emergency STOP up to Stop Category 1 ... 6-50
6.1.2.15 FB 216 "F_FDBACK": Feedback Monitoring ... 6-53
6.1.2.16 FB 217 "F_SFDOOR": Safety Door Monitoring ... 6-57
6.1.2.17 FB 223 "F_SENDDP" and FB 224 "F_RCVDP": Send and Receive Data via PROFIBUS DP ... 6-61

6.1.2.18 FB 225 "F_SENDS7"and FB 226 "F_RCVS7": Communication via S7 Connections ... 6-67
6.1.2.19 FC 174 "F_SHL_W": Shift Left 16 Bits ... 6-73
6.1.2.20 FC 175 "F_SHR_W": Shift Right 16 Bits ... 6-74
6.1.2.21 FC 176 "F_BO_W": Convert 16 Data Elements of Data Type BOOL to a Data Element of Data Type WORD ... 6-75
6.1.2.22 FC 177 "F_W_BO": Convert a Data Element of Data Type WORD to 16 Data Elements of Data Type BOOL ... 6-76
6.1.2.23 FC 178 "F_INT_WR": Write Value of Data Type INT Indirectly to an F-DB ... 6-77
6.1.2.24 FC 179 "F_INT_RD": Read Value of Data Type INT Indirectly from an F-DB ... 6-79
6.1.3 F-System Blocks ... 6-80
6.1.4 F-Shared DB ... 6-81
6.2 User-Created F-Libraries ... 6-82

7 System Acceptance Test ... 7-1

7.1 Overview of System Acceptance Test ... 7-1
7.2 Preliminary Acceptance Test for Configuration of F-I/O ... 7-1
7.3 Safety Program Acceptance Test ... 7-5

8 Operation and Maintenance ... 8-1

8.1 Notes on Safety Mode of the Safety Program ... 8-1
8.2 Replacing Software and Hardware Components ... 8-3
8.3 Guide to Diagnostics ... 8-5

9 Checklist ... 9-1

10 Glossary ... 10-1

1 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
- *STEP 7 s*tandard 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	6ES7833-1FC01-0YA5	V 5.3

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
- S7-300 fail-safe signal modules
- Fail-safe DP standard slaves

What's New

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

- Incorporation of product information for this manual (Product Information, Edition 7/2003, A5E00169432-02)
- Description of important innovations in *S7 Distributed Safety*, V 5.3, as follows:
  - Support of two F-runtime groups in the safety program
  - Fail-safe communication between the F-runtime groups of a safety program
  - Fail-safe CPU-CPU communication via S7 connections
  - *LAD/FBD* operations WAND_W, WOR_W, WXOR_W, and "Call multiple instances"
  - WORD data type
  - New F-application blocks of *Distributed Safety* F-library (V1)
  - Support of S7-PLCSIM for hardware simulation

Approvals

S7 Distributed Safety, ET 200S and ET 200eco 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
- Safety class 6 in accordance with DIN V 19250 (DIN V VDE 0801)
- 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 description	• 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: • The *S7-300, Module Specifications* reference manual describes how to assemble and wire S7-300 systems. • The *CPU Specifications: CPU 31xC and CPU 31x* reference manual describes CPUs 315-2 DP and 317-2 DP. • The product information for CPU 315F-2 DP describes deviations from the CPU 315-2 DP only. • The product information for CPU 317F-2 DP describes deviations from the CPU 317-2 DP only. • The *S7-400, Hardware and Installation* installation manual describes how to assemble and wire S7-400 systems. • The *S7-400, CPU Specifications* reference manual describes the CPU 416-2. • The product information for CPU 416F-2 DP describes deviations from the CPU 416-2 DP only. • The *ET 200S, IM 151-7 CPU Interface Module* manual describes the IM 151-7 CPU. • The product information for IM 151-7 F-CPU describes deviations from the IM 151-7 CPU only.
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 manual	Describes the ET 200S fail-safe module hardware (including installation, wiring, and technical specifications)
Automation System S7-300, Fail-Safe Signal Modules manual	Describes the S7-300 fail-safe signal module hardware (including installation, wiring, and technical specifications)

Documentation	Brief Description of Relevant Contents
STEP 7 manuals	• The *Configuring Hardware and Communication Connections with STEP 7 V 5.x* manual describes how to operate the applicable *STEP 7* standard tools. • The *LAD for S7-300/400* manual describes the Ladder Diagram standard programming language in *STEP 7*. • The *FBD for S7-300/400* manual describes the Function Block Diagram standard programming language in *STEP 7*. • The *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)
- F-libraries
- Support for system acceptance test
- Operation and maintenance of S7 Distributed Safety.

Conventions

Documentation:

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. The non-safety user program is referred to as a "standard user program" to differentiate it from the safety 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. Fail-safe blocks with know-how protection are also labeled with a small symbol (lock).

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.siemens.en/simatic-tech-doku-portal

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 redundant automation systems. The H/F Competence Center can also provide assistance with onsite configuration, commissioning, and troubleshooting.

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

For questions about workshops, etc., contact: hf-cc@nbgm.siemens.de

Automation and Drives, Service & Support

Available worldwide, 24 hours a day:

Worldwide (Nuremberg) **Technical Support** Local time: 24 hours/day, 365 days/year Phone: +49 (180) 5050-222 Fax: +49 (180) 5050-223 mailto:adsupport@siemens.com GMT: +1:00		
Europe/Africa (Nuremberg) **Authorization** Local time: M -F 8:00 a.m. to 5:00 p.m. Phone: +49 (180) 5050-222 Fax: +49 (180) 5050-223 mailto:adsupport@siemens.com GMT: +1:00	**United States** (Johnson City) **Technical Support and Authorization** Local time: M -F 8:00 a.m. to 5:00 p.m. Phone: +1 (423) 262 2522 Fax: +1 (423) 262 2289 mailto:simatic.hotline@sea.siemens.com GMT: -5:00	**Asia/Australia** (Beijing) **Technical Support and Authorization** Local time: M -F 8:00 a.m. to 5:00 p.m. Phone: +86 10 64 75 75 75 Fax: +86 10 64 74 74 74 mailto:adsupport.asia@siemens.com GMT: +8:00
English and German-speaking operators are generally available at Technical Support and Authorization.		

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 onsite service, repairs, spare parts, and Much more is available under "Services".

2 Product Overview

2.1 Overview

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 appropriate newsletter, you can stay abreast of the latest information and make system modifications, as necessary. To subscribe, go to http://my.ad.siemens.de/myAnD/guiThemes2Select.asp?subjectID=2&lang=en

and register for the following newsletters:

- SIMATIC S7-300
- SIMATIC S7-400
- Distributed I/O
- SIMATIC Industrial Software

Select the "Updates" check box for each newsletter.

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:

- Requirement class AK1 to AK6 in accordance with DIN V 19250/ DIN V VDE 0801
- 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 (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 a fault, 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.

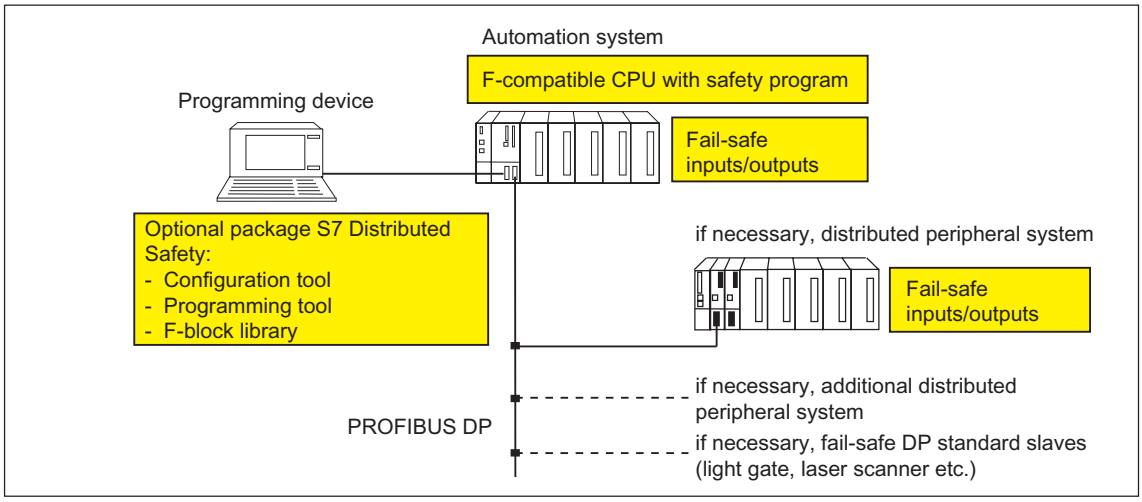
Example of User Safety Function and Fault Reaction Function

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.

2.2 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

Hardware components of an S7 Distributed Safety include the following:

- F-CPU, such as CPU 315F-2 DP
- 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 DP standard slaves

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 description.

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* V 5.3 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 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.

2.3 Installing/Removing the *S7 Distributed Safety* V 5.3 Optional Package

Software Requirements

The following package must be installed in the programming device or PC:

- *STEP 7* V 5.3 + Service Pack 1, or higher

Warning

Use of the *S7 Distributed Safety* V 5.3 optional package with earlier versions of *STEP 7* is not permitted.

Readme File

The readme file contains important up-to-date information about the software (for example, Windows versions supported). You can display the readme file in the setup program or open it at a later time by selecting the **Start > Simatic > Information > English** menu command. The readme file is located in the *S7fprojx* directory of *STEP 7*.

Installing *S7 Distributed Safety*

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

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 assists you in configuring and programming S7 Distributed Safety (in *SIMATIC Manager*, *HW Config,* and *FBD/LAD Editor*).

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 > Work with F-systems**.

Removing *S7 Distributed Safety*

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

- "S7 F Configuration Pack V 5.3 SP2"
- "S7 Distributed Safety Programming V 5.3"

These components can be individually removed. Use the normal procedure in Windows for removing software:

1. In Windows, double-click the "Add/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 enabled file" dialog appears, click "No" in case you are in doubt.

Conversion to *S7 Distributed Safety* V 5.3

Reading a Safety Program with *S7 Distributed Safety* V 5.3:

If you would like to use *S7 Distributed Safety* V 5.3 to read, but not change, a safety program created with an earlier version of *S7 Distributed Safety*, open the "Safety Program" dialog with V 5.3 . Do **not** compile the safety program.

Note

When you open the "Safety Program" dialog for a consistent safety program created with *S7 Distributed Safety* **V 5.1**, the status: "The safety program is consistent." is displayed although different signatures are displayed.

This is due to the fact that the length of the signatures has been changed from 16 to 32 bits.

Changing a Safety Program with *S7 Distributed Safety* V 5.3:

If you want to use *S7 Distributed Safety* V 5.3 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 5.3 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 of V 5.3 are **automatically** replaced following confirmation. The collective signature of all F-blocks and the signatures of individual F-blocks also change, because:

   - The length of the collective signature has been changed from 16 to 32 bits (for conversion from V 5.1 to V 5.3 only)
   - F-blocks of the *Distributed Safety* (V1) F-library were replaced
   - Automatically compiled F-blocks have changed
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.
   - 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 values 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 (see Section, *Safety Program Acceptance Test*).

Conversion of *S7 Distributed Safety* from V 5.3 to V 5.1

Warning

A safety program compiled with *S7 Distributed Safety* V 5.3 must not be read or edited with *S7 Distributed Safety* V 5.1.

Calculation of Maximum Response Time of Your F-System

Use the Microsoft Excel file provided with *S7 Distributed Safety* V 5.**3** to calculate the maximum response time of your F-system. This file can be found on the Internet at:

http://www4.ad.siemens.de/ww/view/en/

under the contribution ID 19138505.

Note

When *S7 Distributed Safety* V 5.3 is installed, the MS Excel file supplied for an earlier version of *S7 Distributed Safety* (*...\Siemens\STEP7\S7Manual\s7fco\s7fcotib.xls)* is deleted. If you made your response time calculations directly in this file rather than in a copy of the file that you created in a **different** folder, save the MS Excel file for the earlier version in another folder **before installing *S7 Distributed Safety* V 5.3**.

Otherwise your calculation entries in the earlier version will be lost when you install *S7 Distributed Safety* V 5.3!

To update the calculations for an earlier version of *S7 Distributed Safety*, you must manually transfer these entries to the file for V 5.3.

---

3 Configuration

3.1 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 Requiring Configuration by User

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

- F-CPU, such as CPU 315F-2 DP
- F-I/O, such as:
  - ET 200S fail-safe modules
  - Fail-safe S7-300 signal modules (for centralized configuration near the F-CPU or distributed configuration in ET 200M)
  - ET 200eco fail-safe I/O modules
  - Fail-safe DP standard slaves

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 description • *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	*Manual for specific fail-safe DP standard slave*

Which Safety-Related CPU-CPU Communication Options can be Configured?

You must configure the following safety-related CPU-CPU communication options in *HW Config:*

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

3.2 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 an S7standard system. That is, you configure and assign parameters to the hardware in *HW Config* as a centralized configuration (F-CPU and, if necessary, S7-300 F-SMs) and/or as a distributed configuration (F-CPU, F-SMs in ET 200M, F-modules in ET 200S and ET 200eco, fail-safe DP standard slaves).
For a detailed description of the configuration options, refer to the *Safety Engineering in SIMATIC S7* system description.

Special F-Relevant Tabs

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

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

For convenience when programming S7 Distribute 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, or an F-CPU, you must recompile the safety program.

3.3 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 permission must not be available through the F-CPU password when changes are made to the standard user program, as this would allow changes to be made to the safety program. To rule out this possibility, you must configure **Level of Protection 1**.

Use the following procedure to configure the level of protection:

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. Select 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 *Access Protection*. Pay particular attention to the warnings in *Setting 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.

   Here, you can change the following parameters or accept the default settings:

   - Basis for the PROFIsafe addresses
   - Band of numbers for F-data blocks
   - Band of numbers for F-function blocks
   - Local data volume provided for the safety program

Note

Changes to the F-parameters of the F-CPU result in modifications to the safety program when it is recompiled, and consequently, a new acceptance test may be required (see *System Acceptance Test*).

"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.

"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 "DB 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) that are used in 2 F-runtime groups**
- **+ 4 x number of F-runtime groups**

If the configured band of numbers is 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 data blocks 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 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-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

+ 4

If the configured band of numbers is 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-function 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 automatically added F-blocks and the F-call blocks of the F-runtime groups of the safety program (for program structure, see *Structure of Safety Program in S7 Distributed Safety*).

Note

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

You must provide **at least 300 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 the amount of local data available for the automatically added F-blocks is insufficient (less than 300 bytes), the runtime of the safety program increases. You will receive a notice via *S7 Distributed Safety*, if the automatically added F-blocks would require more local data than configured. The safety-related program will be compiled anyway.

Warning

The calculated maximum runtime of the safety program using the MS Excel file, s7fcotib.xls , (see also section, Installing/Removing the Option Package S7 Distributed Safety V 5.3) , is no longer be 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)to calculate the maximum response times in the event of an error and for any runtimes of the standard system using the above-mentioned Excel file.

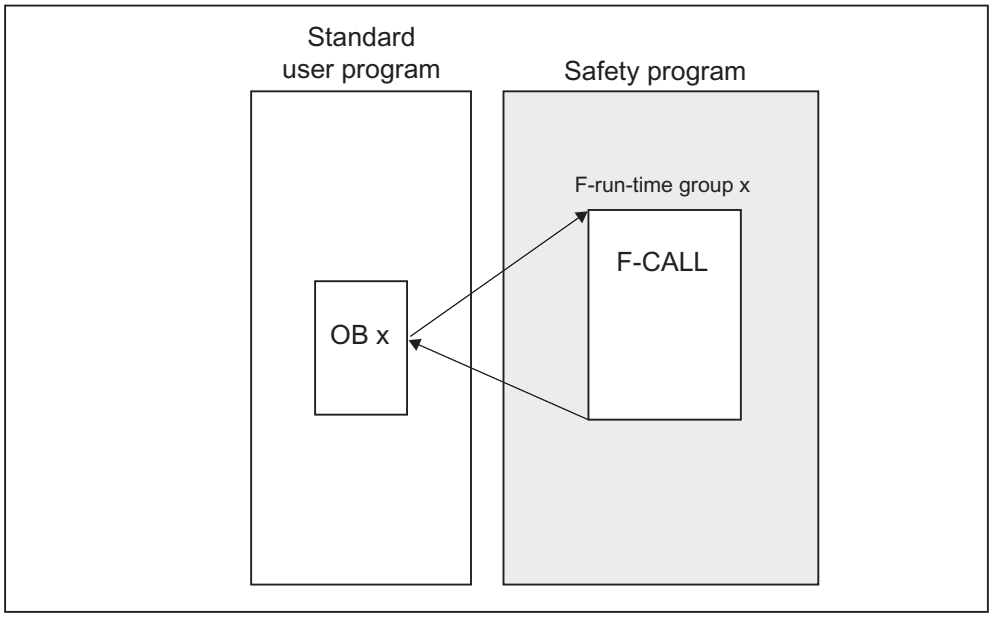
Note

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 OB35s, to the extent 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 Product Information for the utilized F-CPU)*
  For CPU 416F-2, you can configure the local data for each priority class. For this reason, you should allocate the largest possible area
  for local data 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

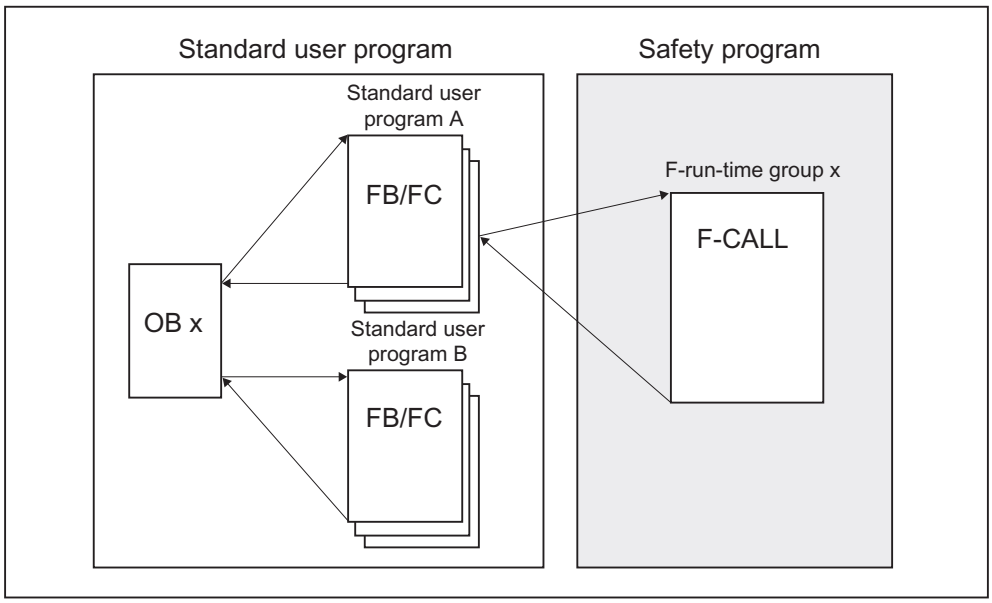


Set the "F-local data" parameter to one of the following:

- 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 local data requirement of OB x (for 2 F-runtime groups of OB x with the greatest local data requirement), if greater than 32 bytes.

Note: You can derive the local data requirement of the OBs from the program structure. In *SIMATIC Manager*, select the **Options > Reference Data > Display** menu command (Setting: "Program structure" selected). This shows you the local data requirement in the path or for the individual blocks (see also *STEP 7 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 or
- Maximum amount of local data of the F-CPU you are using minus the local data requirement of OB x (for 2 F-runtime groups of OB x with the greatest local data requirement) and minus the local data requirement of the standard user program A if this is greater than 32 bytes.

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

Local Data Requirement for the Automatically Added F-Blocks According to Local Data Requirement of User 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 the F-PB across 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 operation is programmed.

The estimated local data requirement for the automatically added F-blocks is thus 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 F-CPUs from the S7-400 range):

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

Mean local data requirement in bytes =

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

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

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

\+ 12

\+ 14 (if a fixed-point function or word logic operation 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.

3.4 Configuring the F-I/O

F-Systems Configured Same as Standard Systems

The ET 200S and ET 200eco 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 the **Edit > Object Properties** menu command or by double-clicking the F-I/O.

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 white fields.

Additional Information

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

For a description of the **parameters**, refer to the *context-sensitive online Help for the tab* and the respective *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 description.

PROFIsafe Address Setting

New PROFIsafe addresses are automatically assigned each time an F-I/O module is placed in *HW Config*. This refers to the PROFIsafe destination address and the PROFIsafe source address.

For ET 200S and ET 200eco F-Modules or fail-safe DP standard slaves, the highest available PROFIsafe destination address (maximum 1022) is automatically assigned by *S7 Distributed Safety*. You can change the PROFIsafe destination address. You must set the PROFIsafe destination address ("DIP switch position" parameter) on the F-module using the DIP switch **before** installing the F-module.

Likewise, the PROFIsafe source address (of the associated F-CPU) is automatically assigned. It is generated from the "Basis for PROFIsafe addresses" CPU-parameter and the PROFIBUS address of the DP interface module.

The **S7-300 F-SMs** are addressed in two ways for safety mode:

- SM 326; DI 24 × 24 VDC (Order No. 6ES7326-1BK0**0**-0AB0), SM 326; DI 8 × Namur, SM 326 DO 10 × 24 VDC/2 A, and SM 336; AI 6 × 13 Bit -- by means of its initial address (range: 8 to 8176, in increments of 8)
  The initial address in *HW Config* must match the DIP switch position on the S7-300 F-SM. The PROFIsafe destination address is the same as the initial address divided by 8. For this reason, assign low initial addresses for S7-300 F-SMs if you are also using ET 200S F-modules or fail-safe DP standard slaves.
- For SM 326; DI 24 × 24 VDC (Order No. 6ES7326-1BK0**1**-0AB0 and higher) and SM 326; DO 8 × 24 VDC/2 A PM, the highest available PROFIsafe destination address (maximum of 1022) is assigned automatically by *S7 Distributed Safety*. You can change the PROFIsafe destination address. You must set the PROFIsafe destination address ("DIP switch position" parameter) on the module using the DIP switch **before** installing the module.

Warning

The switch setting on the address switch of the F-I/O, i.e., its PROFIsafe destination address, must be unique on a network-wide* and station-wide** (system-wide) basis. You can assign a maximum of 1022 PROFIsafe destination addresses in a system, i.e., a maximum of 1022 F I/O can be addressed via PROFIsafe.
Exception: In different I-slaves, F-I/O can have the same PROFIsafe destination address since they are only addressed within the station, i.e., by the F-CPU in the I-slave.

The following restriction applies only to ET 200S F-modules or fail-safe DP standard slaves, in which the predefined PROFIsafe address settings **cannot be changed** in *HW Config*:

If you use ET 200S F-modules or fail-safe DP standard slaves in a PROFIBUS network, in which PROFIsafe addresses cannot be modified in *HW Config*, you can only operate **one DP master with F-CPU** in this network; otherwise, the system-wide uniqueness of the PROFIsafe addresses cannot be guaranteed.

* A network consists of one or more subnets. "Network-wide" means, beyond PROFIBUS subnet limits.

** "Station-wide" means, for one station in *HW Config* (e.g., an S7-300 station or an I-slave)

Note

If you make a change that causes the PROFIsafe destination address to change, you must adjust the DIP switch position (or a corresponding address setting for fail-safe DP standard slaves). For example, if you move an ET 200S F-module from one ET 200S distributed I/O system to another, you must update the DIP switch. Reinserting an F-module within an ET 200S does not count as a change.

Note that a change to the PROFIBUS address of the DP interface module of the F-CPU or CPU parameter "Basis for PROFIsafe addresses" causes a change in the safety-relevant configuration (PROFIsafe source address) of the F-I/O, and, as a result, the safety program is modified the next time it is recompiled.

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 deactivate the group diagnostics on **unused** input and output channels of the following F-SMs:

- SM 326; DI 8 × NAMUR
- SM 326; DO 10 × 24 VDC/2 A
- SM 336; AI 6 × 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 deactivated group diagnostics only for unused input and output channels.

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

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

3.5 Configuring Fail-Safe DP Standard Slaves

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.

Configuration with Device Database Files

As is the case in a standard system, the basis for configuring fail-safe DP standard slaves is the specification of the device in the device database file (*.GSD file). All of the properties of a DP slave are stored in one device database file (*.GSD file). A portion of the specification is protected by the CRC for fail-safe DP standard slaves.

The device database files (*.GSD files) are supplied by the device manufacturer.

Procedure for Configuring with Device Database Files (*.GSD Files)

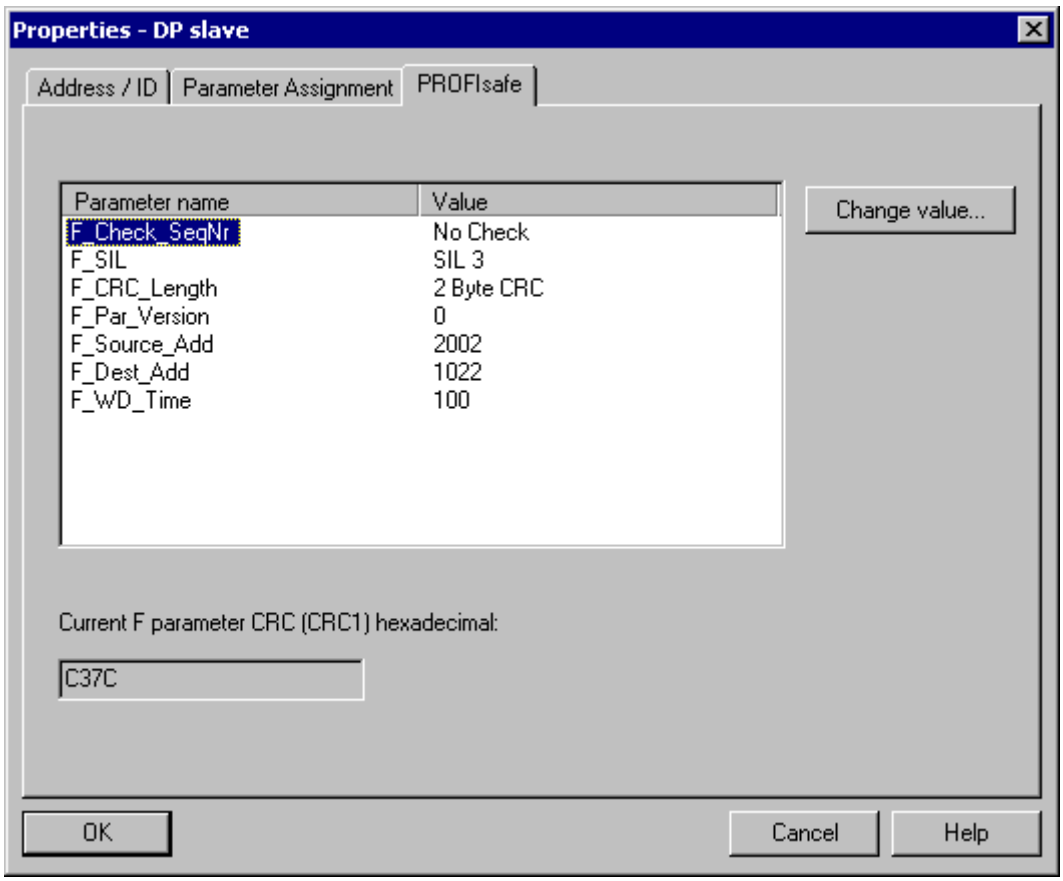
You import the device database files (*.GSD files) in your project (see *STEP 7 Online Help*).

1. Select the fail-safe DP standard slave in the hardware catalog of *HW Config* and insert it in your DP master system.
2. Select the fail-safe DP standard slave.
3. Open the "Properties – DP slave" tab using the **Edit > Object Properties** menu command or by double-clicking the slot for the F-component.

"PROFIsafe" Tab

The text of the parameters specified in the device database file (*.GSD file) is contained in the "PROFIsafe" tab under "Parameter name," and the corresponding current value for each parameter is included 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.

The "F_Check_SeqNr" parameter must be set to "No check." Only fail-safe DP standard slaves that behave accordingly are supported.

"F_SIL" Parameter

This parameter defines the safety class of fail-safe DP standard slaves. The parameter is device-dependent. Possible settings for the "F_SIL" parameter are "SIL 1" to "SIL 3", depending on the device database file (*.GSD file).

"F_CRC_Length" Parameter

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

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 user data lengths greater than 13 bytes and less than or equal to 122 bytes, select "4-byte CRC."

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

"F_Par_Version" Parameter

This parameter identifies the current version of PROFIsafe. This parameter has a starting value of 0 and is increased by 1 with each version.

This parameter cannot be modified.

"F_Source_Add" and "F_Dest_Add" Parameters

The PROFIsafe addresses are used to uniquely identify the source and destination. The addresses are assigned automatically to prevent incorrect assignment of parameters.

The "F_Dest_Add" parameter can be assigned a value between 1 and 1022. You can change the value for "F_Dest_Add." The "F_Source_Add" parameter can accept values between 1 and 65534 (see also PROFIsafe Address Setting in *Configuring the F-I/O*).

"F_WD_Time" Parameter

This parameter defines the monitoring time in the fail-safe DP standard slave.

A valid current safety message frame must reach the F-CPU within the monitoring time. Otherwise, the fail-safe DP standard slave goes to the 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 fault occurs (see *Safety Engineering in SIMATIC S7* system description).

The "F_WD_Time" parameter can be set in 1 ms increments. The range of the "F_WD_Time" parameter is specified by the device database file (*.GSD file).

3.6 Assigning Symbolic Names

Symbolic Name for F-I/O DBs

An F-I/O DB is automatically generated for each F-I/O during compilation in *HW Config* (see *F-I/O DB*), and a symbolic name is simultaneously assigned to it in the symbol table.

The symbolic name is generated in each case by combining the prefix "F" with the initial 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_DC24V). In so doing, any special characters included in the name are replaced with "_".

If a name other than the default name entered in the object properties for the F-I/O module is to be used, you must change the name in the object properties for the F-I/O in *HW Config* **before** compiling for the first time. 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 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

Take care not to use the "description" that can be entered in *HW Config* (instead of the name) for generating the symbolic names for the associated F-I/O DB. The symbolic name is always generated in this case using the prefix "F," the initial address of the fail-safe DP standard slave, 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 assigned to a particular F-I/O module exclusively by means of the number of the F-I/O DB and not by means of the initial address entered in the symbolic names by default.

For this reason, you must not modify the automatically assigned numbers of the F-I/O DB; 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 SM 336; AI 6 x 13 Bit

If you want to assign symbols for the input channels of SM 336; AI 6 x 13 Bit, make sure that the symbols are of data type INT.

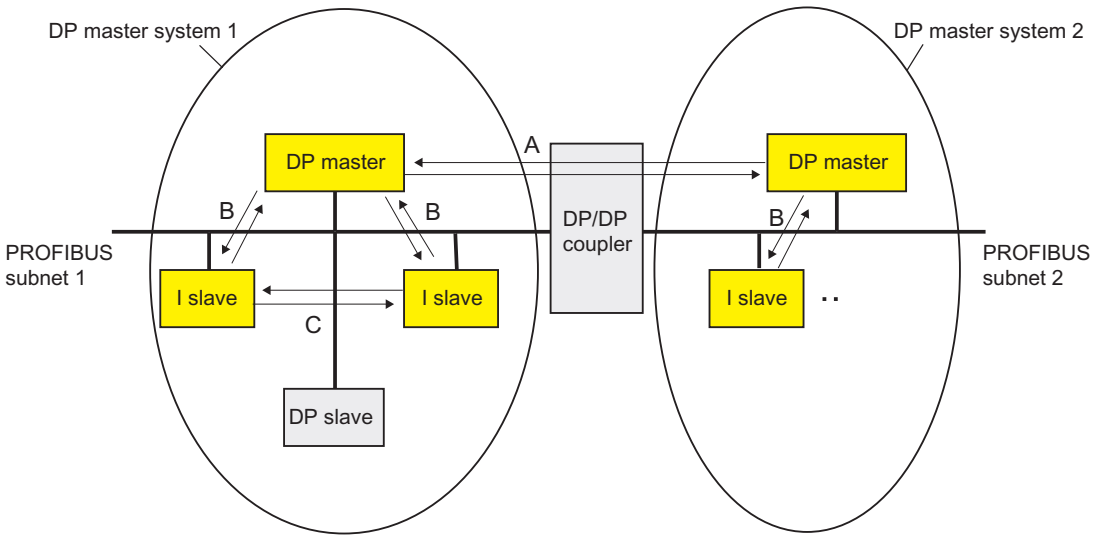
3.7 Configuring Safety-Related CPU-CPU Communication

Overview of Safety-Related CPU-CPU Communication via PROFIBUS DP

The schematic below summarizes the three options for safety-related CPU-CPU communication via PROFIBUS DP in S7 Distributed Safety F-systems.

In safety-related CPU-CPU communication, a fixed amount of fail-safe data of data types BOOL and INT are transmitted in a fail-safe manner between safety programs in F-CPUs of DP masters/I-slaves.

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.



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

Overview of Safety-Related CPU-CPU Communication via Industrial Ethernet

Safety-related CPU-CPU communication via Industrial Ethernet is possible by means of configured S7 connections. The communication takes place from and to CPUs 416F-2 as of firmware version V 3.1.4.

In safety-related communication via S7 connections, a specified amount of fail-safe data of data types BOOL, INT, WORD, or TIME are transmitted in a fail-safe manner between 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.

3.7.1 Configuring Safety-Related Master-Master Communication

Overview

This section describes how to configure safety-related communication between safety programs on F-CPUs of DP masters. A DP/DP coupler is required for this.

DP/DP coupler

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

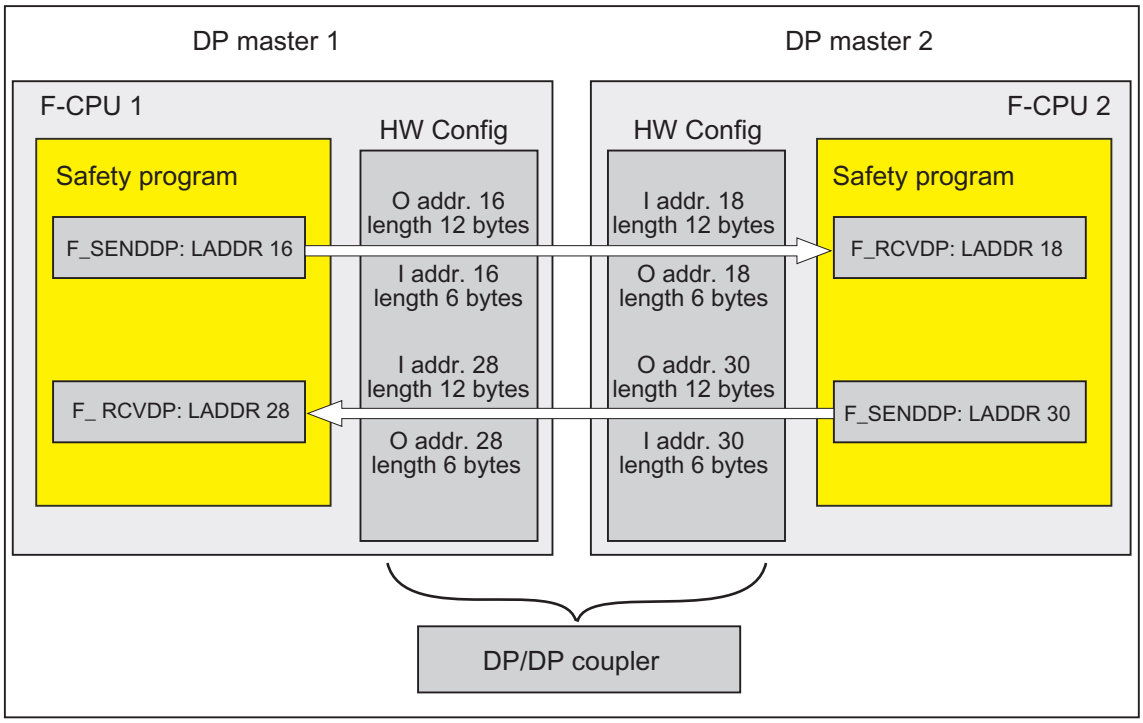
Each of the two F-CPUs is connected to the DP/DP coupler by means of its PROFIBUS-DP interface.

Note

Switch 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 is supposed to be able to send and receive data.



Rules for Defining the Address Areas

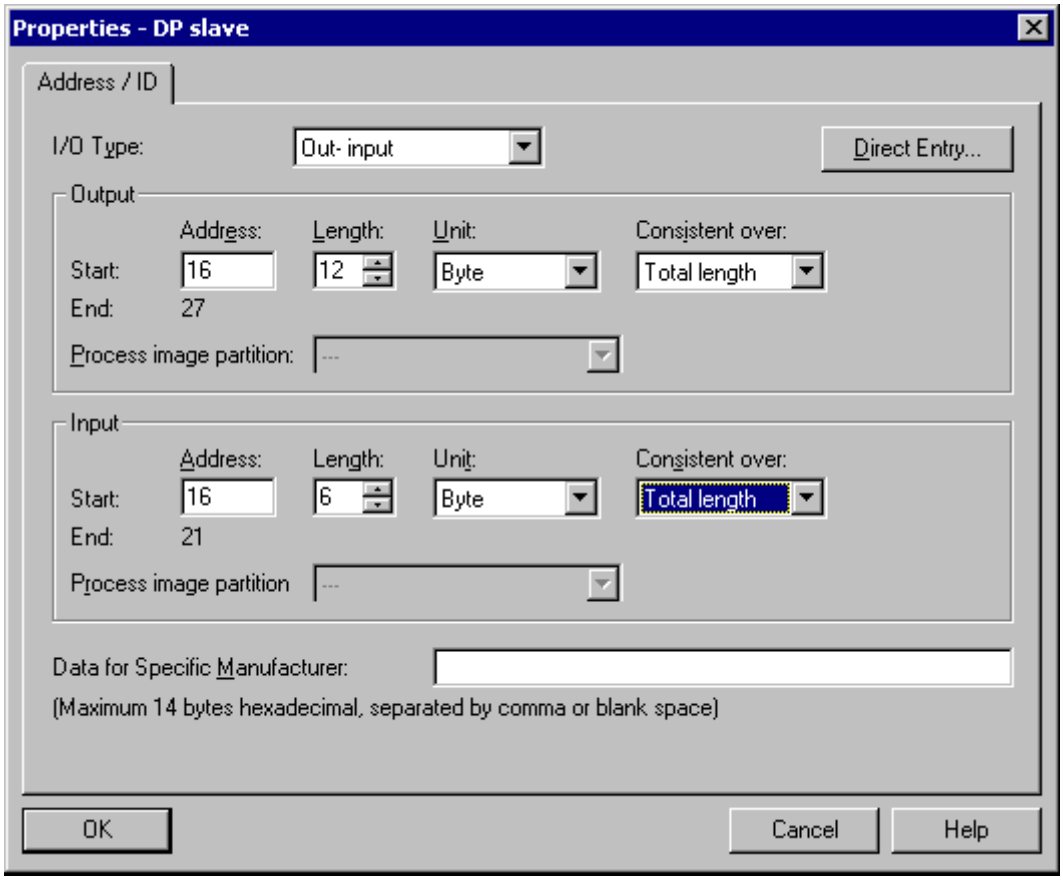
The output data address area for **data to be sent** must begin with the same initial address as the associated input data address area. A total of 12 bytes (consistent) is required for the output data address area, while 6 bytes (consistent) are required for the input data address area.

The input data address area for **data to be received** must begin with the same initial 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.

How to Configure Master-Master Communication

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

1. Open a station.
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 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.
4. In the "Network Settings" tab, you should set the transmission rate to at least "1.5 Mbps. "You must select "DP" as the profile. In order to configure **consistent** safety-related communication between F-CPUs and to use any address and length settings, you must use **universal modules**. Select "DP/DP" on the DP master system and insert the universal module(s) from the DP/DP coupler folder.
5. Two (or more) F-CPUs take part in safety-related master-master communication, for example, F-CPU 1 and F-CPU 2. You must insert a universal module and perform steps 1 through 10 for **each** of these F-CPUs. Use two universal modules for each F-CPU for bidirectional connections, that is, when each F-CPU is to send and receive data.
6. Select the universal module and select the **Edit > Object Properties** menu command.
   The object properties dialog is displayed.



7. In the object properties for the DP/DP coupler, select "Output/Input" as the I/O type.
8. Enter the associated values for the output data address area. In our example, enter "16" for the initial address, "12" for the length, "Byte" for the unit, and "total length" for the "Consistent over" field.
9. Enter the associated values for the input data address area. In our example, enter "16" for the initial address, "6" for the length, "Byte" for the unit, and "total length" for the "Consistent over" field.
10. Click "OK" to confirm. This completes the configuration of master-master communication for F-CPU 1. Perform steps 6 to 10 for F-CPU 2.

Note

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

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.

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

Additional Information

For information on programming the safety-related communication between F-CPUs, refer to *Programming Safety Related CPU-CPU Communication*.

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

3.7.2 Configuring Safety-Related Master-I-Slave Communication

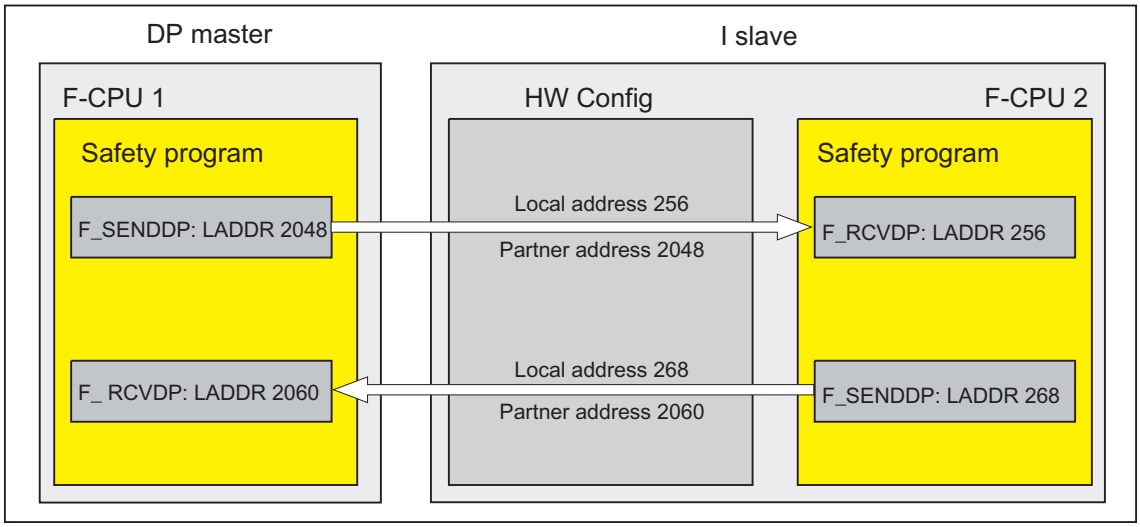
Overview

This section describes how to configure safety-related communication between the safety program on the F-CPU of a DP master and the safety program(s) on the F-CPU(s) of one or more I-slaves. The safety-related communication takes place by means of master-I-slave connections, same 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 is supposed to be able to send and receive data.



You configure the following in the object properties dialog 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

The configured addresses are assigned to the LADDR parameter of the corresponding F_SENDDP and F_RCVDP F-application blocks in the safety program *Programming Safety-Related Master-I-Slave/I-Slave-I-Slave Communication*).

Address Areas

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

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

How to Configure Master-I-Slave Communication

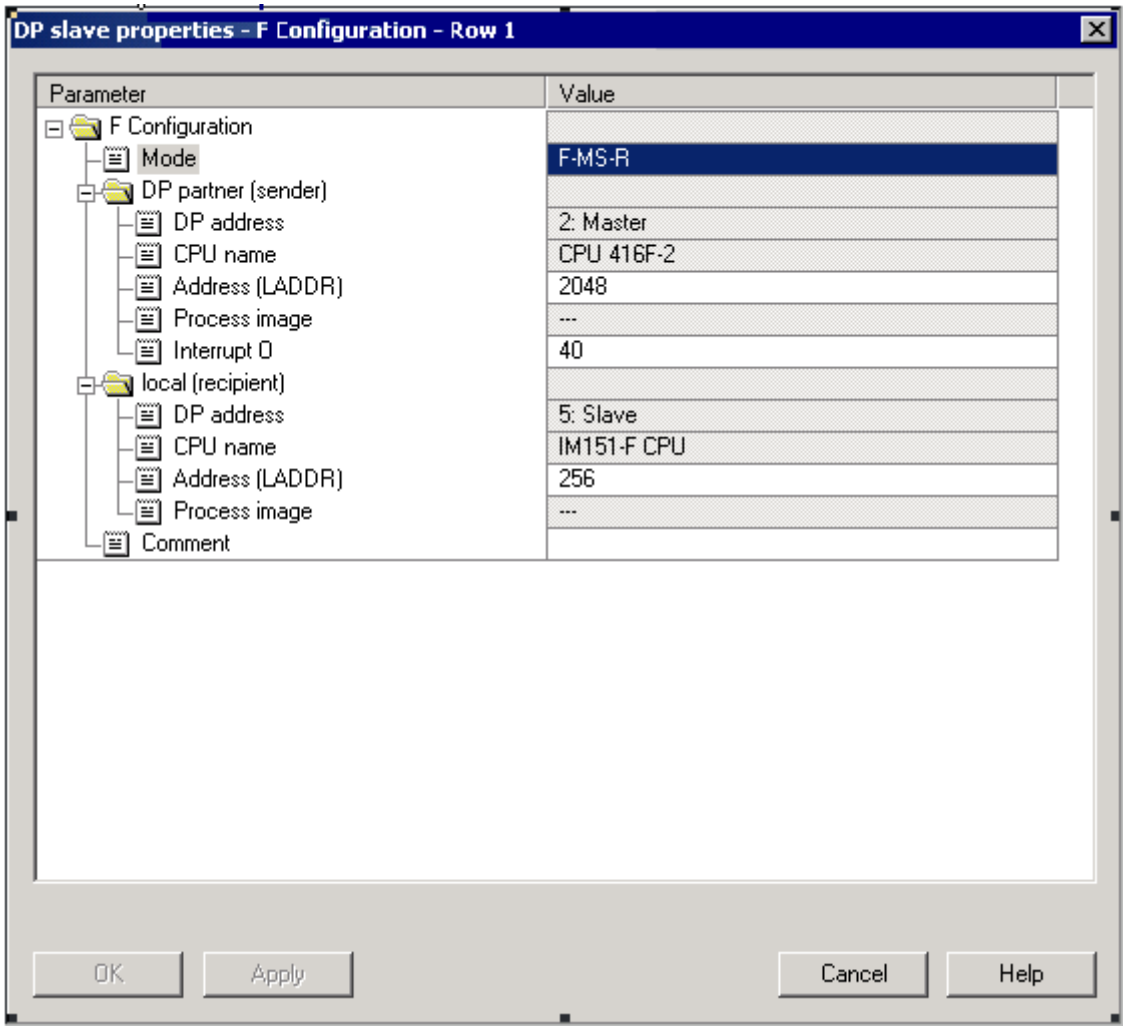
In this section we demonstrate address area configuration using the previous figure as an example.

Requirement:

You have created a project in *STEP 7*.

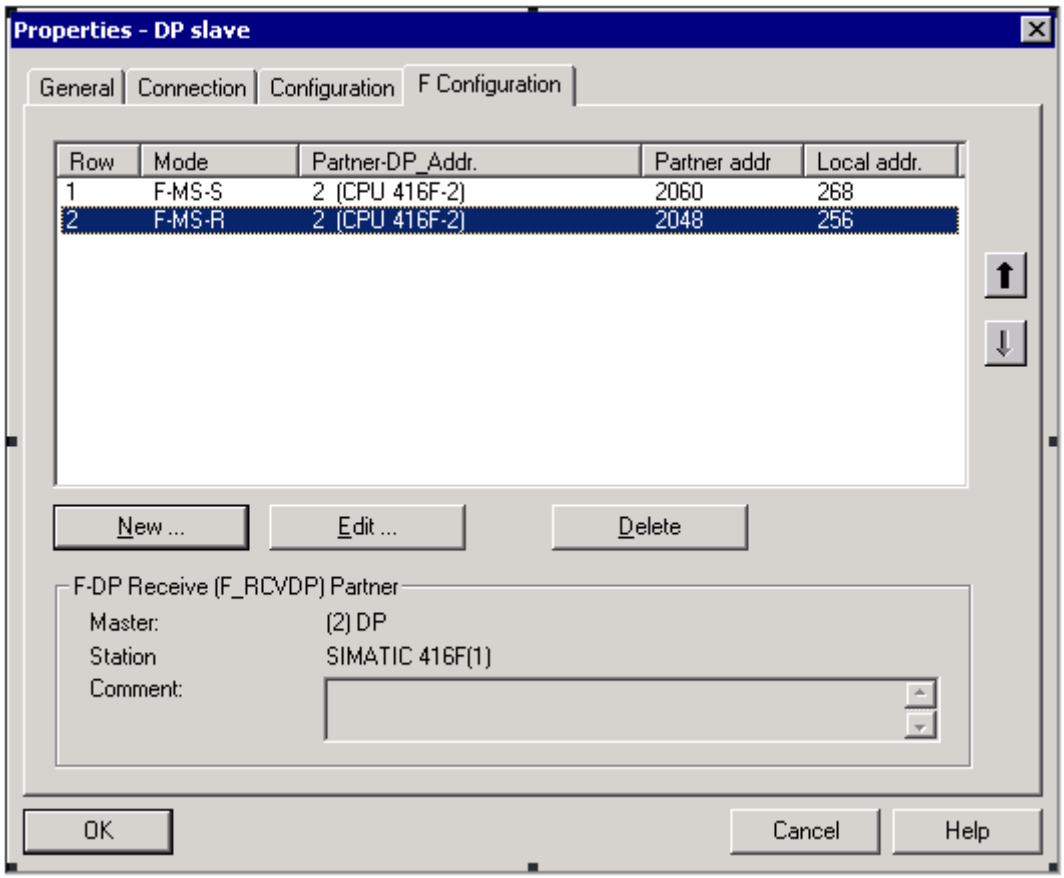
1. Create a station in your project (in *SIMATIC Manager,* for example, an S7-300 station).
2. Assign a CPU with fail-safe capability to this station (in *HW Config*, from the hardware catalog).
3. Configure this CPU as a DP slave (in *HW Config*, in the "Operating Mode" tab of the Object Properties for the DP interface of the CPU).
4. Create another station and assign a CPU with fail-safe capability (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, the "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 specify the address areas for the safety-related master-I-slave communication:
8. In the "F-Configuration" tab of the Object Properties of 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 over a 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.

The dialog box has the following appearance:



10. Confirm your entries with "OK."
11. In the "F-Configuration" tab of the Object Properties of 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-S" (send over a 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 rows for this example:



Note

In the object properties of the 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 master-I-slave communication is not possible.

You can find out the assigned address areas in the DP master and I-slave from the "Configuration" tab.

Additional Information

You will find a description of the parameters in the *context-sensitive online help of the "F-Configuration" tab*.

For information on programming safety-related master-I-slave communication, refer to *Programming Safety-Related Master-I-Slave/I-Slave-I-Slave Communication*).

For information on master-I-slave communication, refer to the *STEP 7 online help*.

For information on address areas, process image partitions, and supported interrupt OBs, refer to the *Technical Specifications for the F-CPU you are using*.

3.7.3 Configuring Safety-Related I-Slave-I-Slave Communication

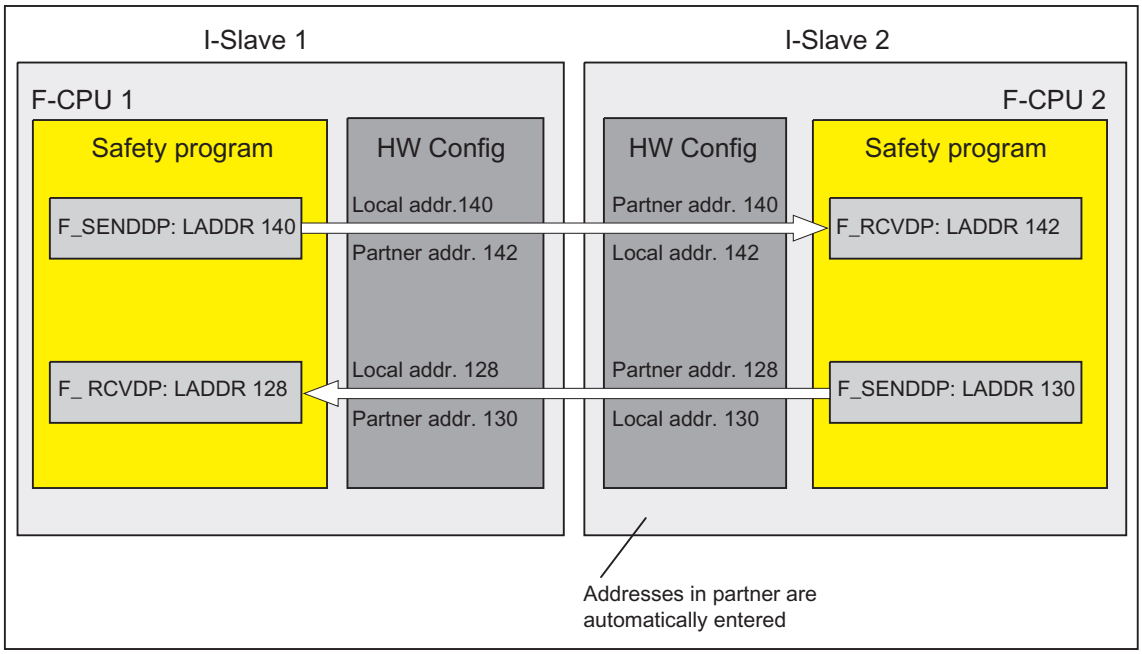
Overview

This section describes how to configure safety-related communication between the safety program on F-CPUs of I-slaves. The safety-related communication takes place by means of direct data exchange, same as in standard systems.

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 is supposed to be able to send and receive data.



You specify the configuration of the following in the object properties dialog for I-slave 1:

- to send to I-slave 2, a local address (I-slave 1) and a partner address (I-slave 2)
- to receive 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 dialog for I-slave 2. The addresses are entered automatically in the object properties dialog for I-slave 2.

The configured addresses are assigned to the LADDR parameter of the corresponding F_SENDDP and F_RCVDP F-application blocks in the safety program *Programming Safety-Related Master-I-Slave/I-Slave-I-Slave Communication*).

Address Areas

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

Communication:	Assigned Address Area on the F-CPU* of the ...
Send: I-slave 1 to I-slave 2	I-slave 1: 12 bytes of output and 6 bytes of input data
	I-slave 2: 12 bytes of input and 6 bytes output data
	DP master: 12 + 6 bytes of input data
Receive: I-slave 1 from I-slave 2	I-slave 1: 12 bytes of input and 6 bytes of output data
	I-slave 2: 12 bytes of output and 6 bytes of input data
	DP master: 12 + 6 bytes of input data

* The CPU of the DP master can be an F-CPU or a standard CPU.

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.

How to Configure I-Slave-I-Slave Communication

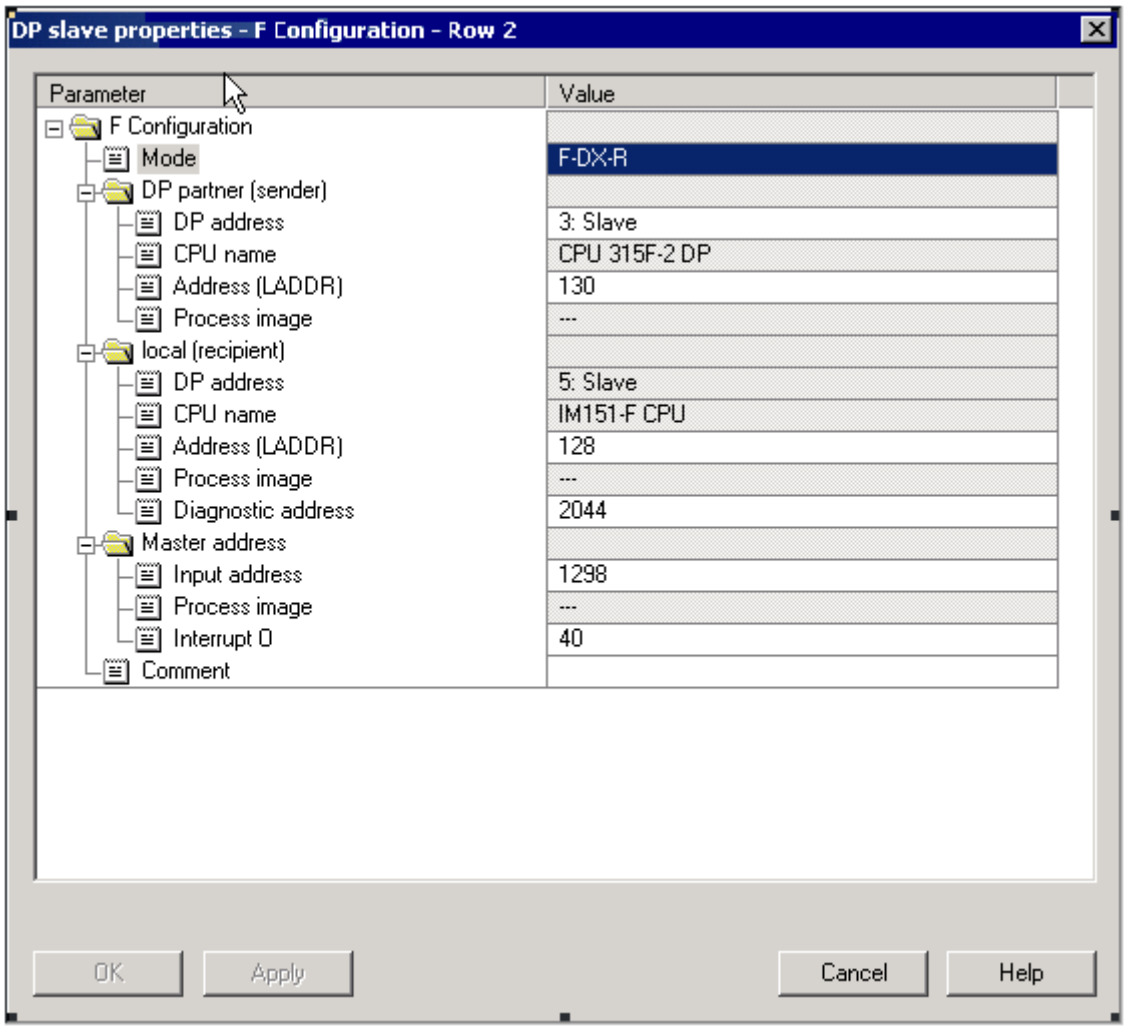
In this section we demonstrate address area configuration using the previous figure as an example.

Requirement:

You have created a project in *STEP 7*.

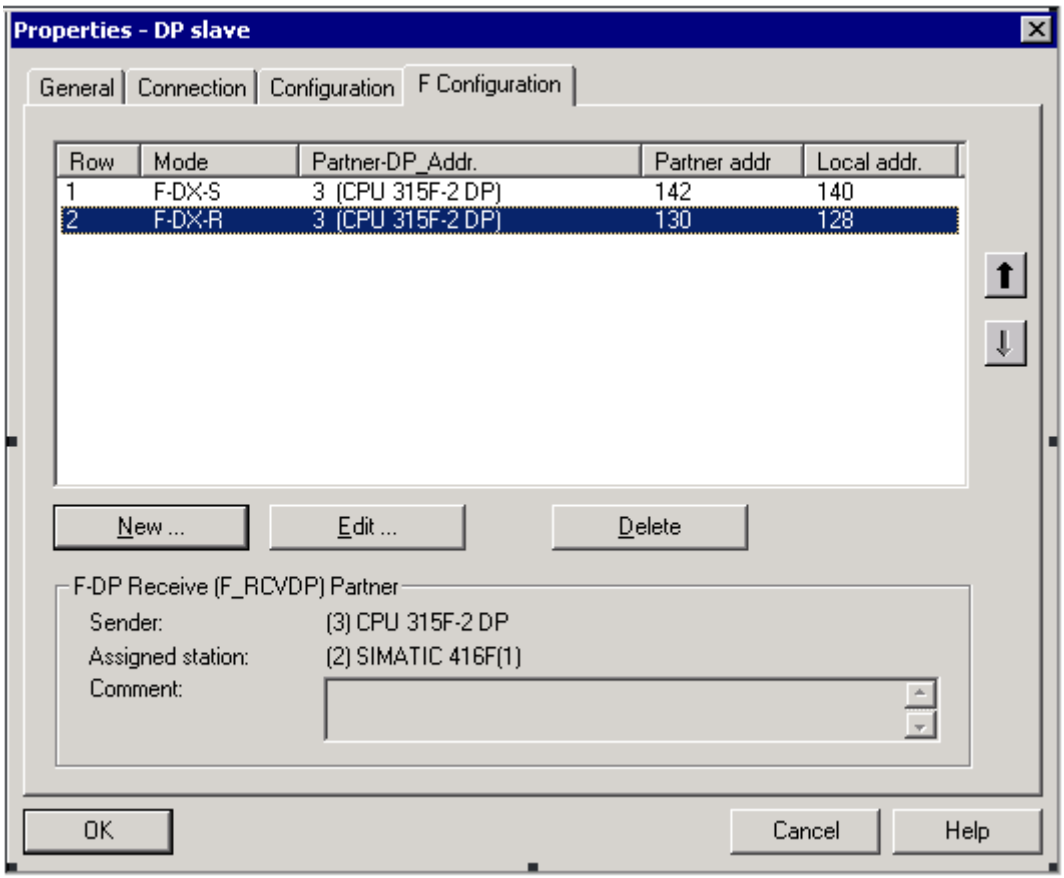
1. Create a station in your project (in *SIMATIC Manager,* for example, an S7-300 station).
2. Assign a CPU with fail-safe capability to this station (in *HW Config*, from the hardware catalog).
3. Configure this CPU as a DP slave (in *HW Config*, in the "Operating Mode" tab of the Object Properties 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 a CPU with fail-safe capability (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 specify the address areas for the safety-related I-slave-I-slave communication:
10. In the object properties for I-slave 1, select the "F-Configuration" tab and click "New."
11. In the next dialog, make the following entries for the receive connection from I-slave 2 in our example:
    - For "Mode: F-MS-R" (receive by means of 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.

The dialog box has the following appearance:



12. Confirm your entries with "OK."
13. In the object properties for I-slave 1, select the "F-Configuration" tab and click "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 using 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.

15. Confirm your entries with "OK."
    This results in two configuration rows for this example:



Note

In the object properties of the relevant 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 find out the assigned address areas in the DP master and in the I-slaves in the "Configuration" tab.

Additional Information

You will find a description of the parameters in the *context-sensitive online help of the "F-Configuration" tab*.

For information on programming safety-related I-slave-I-slave communication, refer to *Programming Safety-Related Master-I-Slave/I-Slave-I-Slave Communication*. For information on address areas, process image partitions, and supported interrupt OBs, refer to the *Technical Specifications for the CPU you are using*.

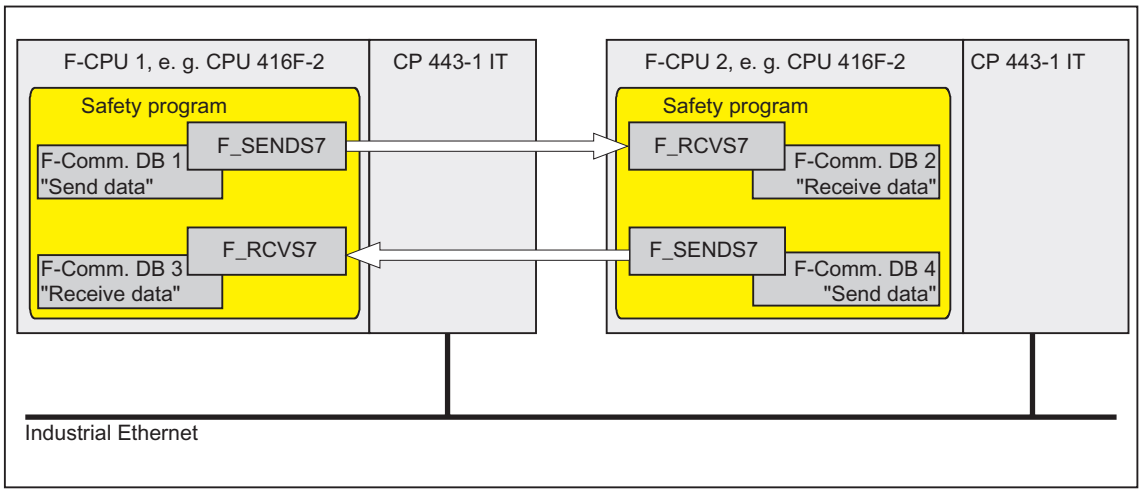
3.7.4 Configuring Safety-Related Communication via S7 Connections

Overview

Safety-related communication via S7 connections is configured the same as in standard systems. This section describes only the particularities involved in configuring safety-related communication between safety programs on F-CPUs via S7 connections.

Communication using F_SENDS7 and F_RCVS7

You use the **F_SENDS7 and F_RCVS7** F-application blocks for sending and receiving data in a fail-safe manner via S7 connections. In the figure below, each of the two F-CPUs is supposed to be able to send and receive data.



Restrictions

Note

In Distributed Safety V 5.3, S7 connections are generally permitted over Industrial Ethernet only!

Safety-related communication via S7 connections to and from CPUs 416F-2 is only possible with **firmware version V 3.1.4** or later.

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. You can change the local ID in *NetPro*, if necessary. You assign the local ID to the ID parameter of the appropriate F-application blocks in the safety program.

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, you have to generate the safety program again.

Additional Information

For a description of configuring S7 connections, refer to the *Configuring Hardware and Communication Connections with STEP 7 V 5.x* and the *STEP 7 Online Help*.

For more information on programming safety-related communication via S7 connections, refer to *Programming Safety-Related CPU-CPU Communication via S7*.

4 Access Protection

4.1 Overview of Access Protection

Introduction

Access to the S7 Distributed Safety fail-safe system is protected by two password prompts: one for the F-CPU and another for the safety program. This chapter shows you how to set up, change, and revoke access permission for the F-CPU and for the safety program.

Password Assignment, Password Prompts, and Validity of Access Permission

Access protection associated with the F-CPU password is not the same as the access protection associated with the safety program password.

	Password for F-CPU	Password for Safety Program
Assignment	During configuration of the F-CPU, in *HW Config,* Properties - CPU dialog, "Protection" tab, Level of protection "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 • When compiling for the first time
Prompt	• When the safety program is loaded in its entirety • When F-blocks with an F-attribute are loaded or unloaded	• When F-blocks are loaded • When compiling in the "Safety Program" dialog • When the F-PB is opened • When F-FBs/F-FCs are opened • When F-DBs are opened • When the "Edit F-Runtime Groups" dialog is opened • When safety mode is deactivated • When the password is changed • When data in the safety program are modified
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 one hour or until permission is revoked (see *Revoking Access Permission for the Safety Program* in *Setting Access Permission for the Safety Program*). Exiting *STEP 7* does not cancel an existing authorization. The validity period is reset each time one of the actions listed in the "Prompt" row is performed, so you only have to enter the password at the beginning of an extended work session.

4.2 Setting Access Permission for the Safety Program

Procedure for Setting Access Permission for the Safety Program

Use the following procedure to enter 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 "Set permission for the safety program" dialog.

Warning

If access protection is not used to limit access to the programming device or PC to those persons authorized to modify the safety program, the following organizational measures must be taken on the part of the programming device or PC to ensure that the password protection is effective:

- 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.

Assigning a New Password for the Safety Program

If you have not yet entered a password for the safety program, the "Set permission for the safety program" dialog will prompt you to enter a password in the "New password" field and reenter the password in the "Confirm password" field.

Warning

You should use different passwords for the F-CPU and the safety program to increase the level of access protection.

Changing the password for the safety program

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

Revoking Access Permission for the Safety Program

You can revoke the access permission for the safety program in the "Set permission for the safety program" dialog by clicking the "Revoke" button. You will then be prompted to enter the password for the safety program again the next time you perform an action requiring a password (for example, opening or loading an F-block). To "revoke" access permission when modifying, the connection to the F-CPU must be terminated (for example, by closing the *STEP 7* applications).

Overview of Password for the Safety Program

There is a distinction between an offline password and an online password for the safety program.
The offline password is included in the safety program in the offline project on the programming device..
The online password is part of the safety program in the F-CPU.

Once the correct password has been entered for the safety program, access is authorized for one hour. After one hour, the password must be reentered. Within this hour, the valid duration of access permission is reset to one hour for each password-protected action. This hour is controlled separately for the online password and the offline password and is set by online operations (modify, safety mode) and offline operations (all other operations). Access permission can be revoked with immediate effect (see above).

Prompt for Offline Password	Response to Incorrect Entry
• When F-blocks are loaded by means of the "Safety Program" dialog	Action is aborted and an error message is given
• When compiling in the "Safety Program" dialog	Action is aborted and an error message is given
• When the "Edit F-Runtime Groups" dialog is opened	Action is aborted and an error message is given
• When F-PB/F-FBs/F-FCs are opened	The F-PB/F-FB/F-FC cannot be changed following an incorrect password entry
• When F-DBs are opened	The F-DB cannot be changed following an incorrect password entry

Prompt for Online-Password Once Safety Program is Downloaded to F-CPU	Response to Incorrect Entry
When safety mode is deactivated (The password must always be entered, even if access to the safety program is still authorized)	Action is aborted and an error message is given
When data in the safety program are modified	Action is aborted and an error message is given

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 you cannot otherwise download it by means of *SIMATIC Manager* and *LAD/FBD Editor*.

4.3 Setting Access Permission for the F-CPU

Procedure for Setting 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 (see *Configuration*).
3. Access permission 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.

Warning

If access protection is not used to limit access to the programming device or PC to those persons authorized to modify the safety program, the following organizational measures must be taken on the part of the programming device or PC to ensure that the password protection for the F-CPU is effective:

- 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.

Once access permission is revoked, 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 they are not identical, download the correct safety program to the F-CPU again (see *Modifying the Safety Program in RUN Mode* and *Comparing Safety Programs*).

Transferring the Safety Program to More than One F-CPU

Warning

If **more than one F-CPU** is to be accessible over a network (such as MPI) from **one programming device or PC**, you must take the following additional measures to ensure that the safety program is downloaded to the correct F-CPU:

Use passwords specific to each F-CPU, such as a uniform password for the F-CPUs having the respective MPI address as an extension: "Password_8". Note the following:

- The first time a password is assigned to an F-CPU, this must be done by means of a point-to-point connection (analogous to the first time an MPI address is assigned to an F-CPU).
- Before downloading a safety program to an F-CPU that has not yet been assigned access protection with an F-CPU password, you must first revoke existing access permission for any other F-CPU.

Changing the Password for the F-CPU

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

5 Programming

5.1 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 also 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 programming languages F-FBD and F-LAD and the standard programming languages FBD and LAD

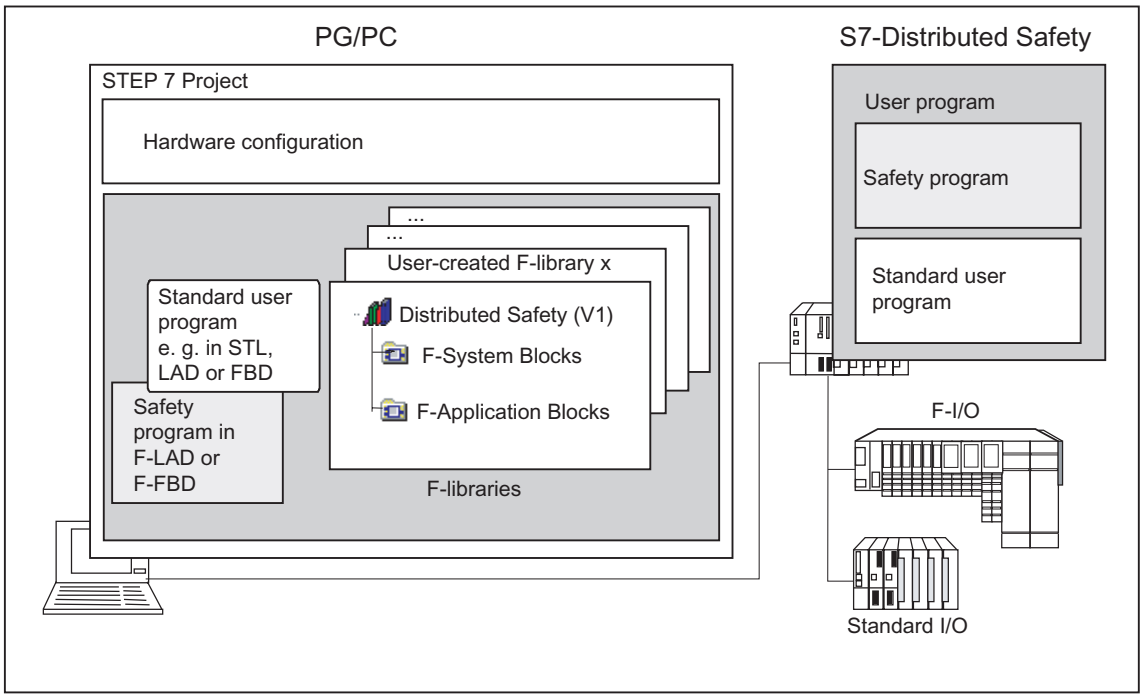
Schematic Structure of a Project with Standard User Program and Safety Program

The following figure 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* (V1) F-block library 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.

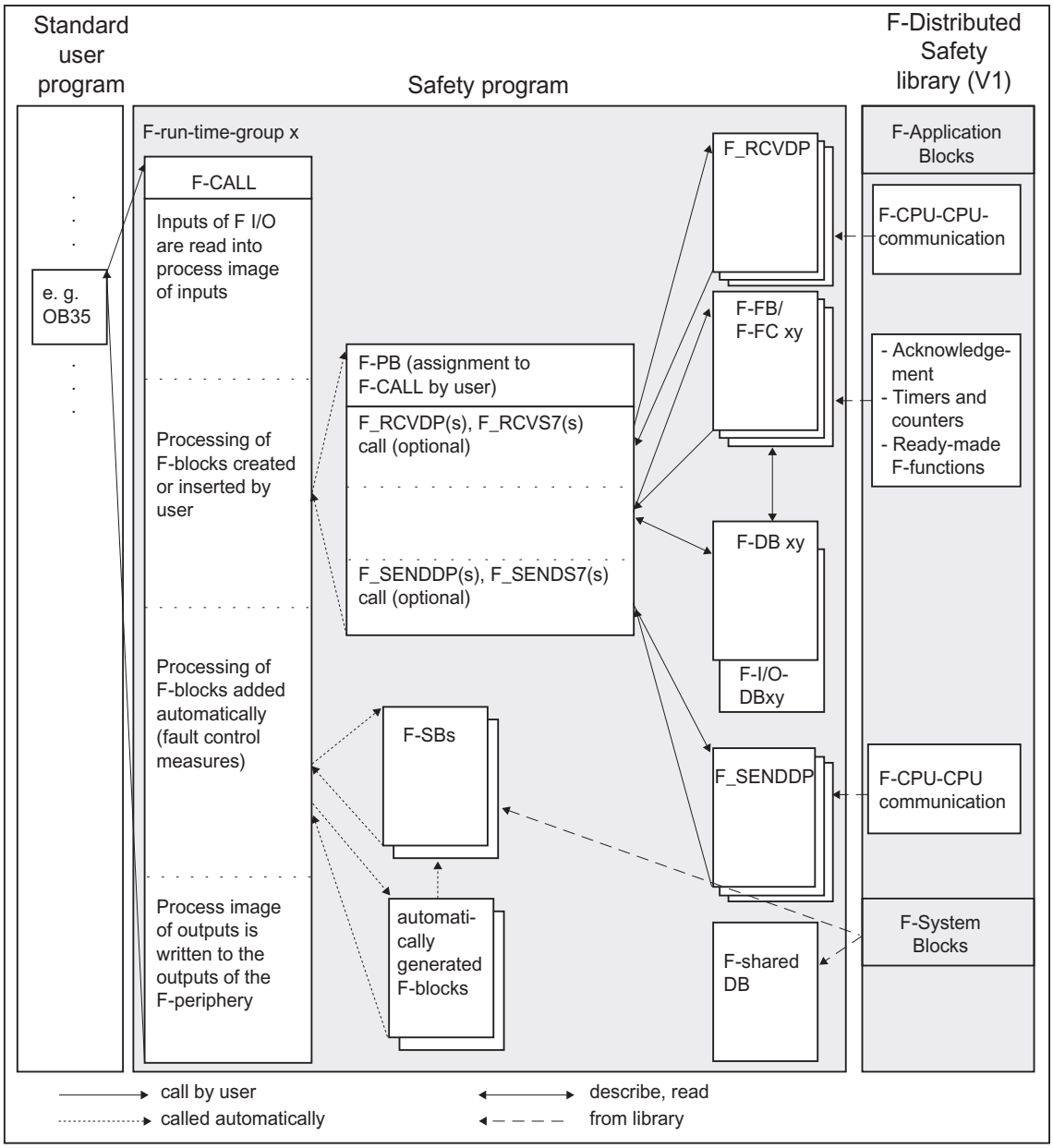


5.1.1 Structure of Safety Program in S7 Distributed Safety

Representation of Program Structure

The following figure shows the schematic structure of a safety program for S7 Distributed Safety. For structuring purposes, a safety program consists of one or two F-runtime groups. Each F-runtime 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

The safety program is accessed 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).

Cyclic interrupt OBs have the advantage of interrupting 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-Runtime Groups

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

An F-runtime group consists of the following:

- One F-call block F-CALL
- One F-program block F-PB (an F-FB/F-FC you assign to the F-CALL)
- Additional F-FBs or F-FCs programmed 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

Structure of Safety Program in Two F-Runtime Groups

Starting in *S7 Distributed Safety* V 5.3, you can divide your safety program into two F-runtime groups. By arranging for portions of your safety program (one F-runtime group) to run in a faster priority class, you achieve a faster safety circuit with short response times. For additional information on F-runtime groups, refer to the section, *Defining F-Runtime Groups*.

5.1.2 Fail-Safe Blocks

F-Blocks of an F-Runtime Group

The F-blocks below are used 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 includes 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.	F-CALL
F-FB/F-FC, F-PB	The user programs 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) (see *Distributed Safety F-Library* (V1)). • Insert F-blocks from "user-created F-libraries" (see *User-Created F-Libraries*) The user defines the call sequence of the F-blocks within the F-PB.	F-FBD/F-LAD
F-DB	Optional fail-safe data blocks that the user can set up with read and write access from the entire safety program	F-DB
F-I/O DB	An F-I/O DB is automatically generated 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 access (see *F-I/O Access*).	–

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

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

- 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 following table gives the F-blocks included in the block containers:

Block Container	Purpose of F-Block	Function/F-Blocks
F-Application Blocks		Block container containing 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
	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 OFF, protective door monitoring, 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		Block container containing the F-system blocks (F-SBs) and the F-global 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 data block	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. The user can evaluate certain data of the safety program in the standard user program by means of its symbolic name F_GLOBDB.

Note

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

5.1.3 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.

Programming Languages F-FBD and F-LAD

The F-FBD and F-LAD programming languages correspond in principle 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 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

Illegal Data and Parameter Types

Illegal types are as follows:

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

For example: You can access input channels of data type BOOL in the process input image of F-I/O only by means of the "input (bit)" unit.

This restriction does not apply when accessing data from the standard user program (for bit memory or process image of standard I/O, refer to Communication between Standard User Program and Safety Program), if a data type declaration was not made for these data in the symbol table. However, you can access these data only by means of the same unit. For example: If you access M0.0, you cannot access MW0, as well.

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

Address Area	Accessible Size Units:	S7 Notation	Description
Process Input Image			
• Of fail-safe 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, 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	Read access to input channels of data type BOOL is possible by means of the "input (bit)" unit only. Access is not possible, for example, with the "input word" unit.
Channels of data type INT (WORD), such as analog channels	Input word	IW	Read access to input channels of data type INT (WORD) is possible by means of the "input word" unit only. It is not possible to access individual bits by means of the "input (bit)" unit.
• Of standard I/O			At the beginning of each OB1 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, bear in mind the update timing when using process image partitions.
	Input (bit) Input word	I IW	Input channels of the standard I/O can only be accessed as read-only channels by means of the units listed. Therefore, transfer to IN_OUT parameters of an F-FB or F-FC is not permitted. Moreover, a validity check is required specifically for each process (see *Communication between Standard User Program and Safety Program*).
Process Output Image			
• Of fail-safe 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 read-only channels. Therefore, 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)	Q	Write access to output channels of data type BOOL is possible by means of the "output (bit)" unit only. Access is not possible, for example, with the "output word" unit.
Channels of data type INT (WORD), such as analog channels	Output word	QW	Write access to output channels of data type INT (WORD) is possible by means of the "output word" unit only. It is not possible to access individual bits by means of the "output (bit)" unit.

Address Area	Accessible Size Units:	S7 Notation	Description
• 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 OB1 cycle, the F-CPU writes the calculated output values to the outputs of the standard I/O (see *Communication between Standard User Program and Safety Program).* With the S7-400, bear in mind the update timing when using process image partitions.
	Output (bit) Output word	Q QW	Output channels of the standard I/O can only be accessed as write-only channels by means of the units listed. Therefore, transfer to IN_OUT parameters of an F-FB or F-FC is not permitted.
Memory Marker	Memory marker (bit) Memory marker word	M MW	This area is used for data exchange with the standard user program. Memory bits can only be accessed by means of the units listed. In addition, read access requires a process-specific validity check. Both read access and write access are possible for a memory marker in the safety program. Therefore, transfer to IN_OUT parameters of an F-FB or F-FC is not permitted. Note that memory bits are only allowed to connect the standard user program and the safety program; they cannot be used as a buffer for F-data (see *Communication between Standard User Program and Safety Program).*
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 they can be 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	Data can only be accessed by means of units that correspond 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 to transfer block parameters and save intermediate results. Note: Transfer of block parameters can also be replaced by fully qualified access: *"Instance DB Name".Parameter Name*.

Address Area	Accessible Size Units:	S7 Notation	Description
	Local data bit Local data word Local data double word	L LW LD	Local data can only be accessed by means of units that correspond to the data type in the declaration table.

Illegal Address Areas

Access by means of units other than those listed in the table above is **illegal**, 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 from the standard user program
- Data blocks (F-DBs) by means of "OPN DI"
- Data blocks that are automatically added
  - Exception: certain data in the F-I/O and 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 "RLO0" and "RLO1" variables in the F-shared DB using fully qualified DB access ("F_GLOBDB".RLO0 or "F_GLOBDB".RLO1).

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/F-FC must always be a write access, which 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 signal state "0" or "1" as 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 addresses: 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	Twos complement integer
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!)
RET	---(RET)	Program control	Return (exit block)
Call multiple instances	Call multiple instances	Program control	Call multiple instances

Instruction		Function	Description
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 attempt to access outputs of F-I/O only with the Assign ("=") (F-FBD) or Output Coil ("--( )") (F-LAD) instruction.

You can evaluate whether an F-I/O is passivated in the associated F-I/O DB (see *F-I/O Access*).

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) instructions 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, this 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 diagnostics events will then be entered in the diagnostics 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 diagnostics events will then be entered in the diagnostics 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.

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 that affected 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 this instruction is increased (see also *Excel File for Response Time Calculation s7fcotib.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 contents of the DB register can be changed following calls of F-FB/F-FC and "fully qualified DB access," 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 access.

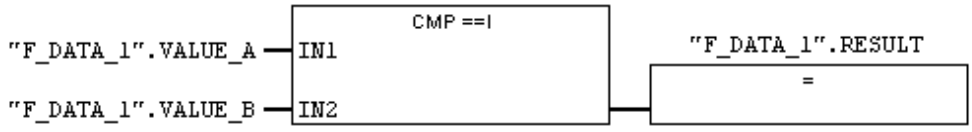
If you still want to use the "OPN DB" instruction, you must ensure that the DB register is restored by repeating the "OPN DB" instruction following calls of F-FB/ F-FC and "fully qualified DB access." Otherwise, an error could result.

"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":

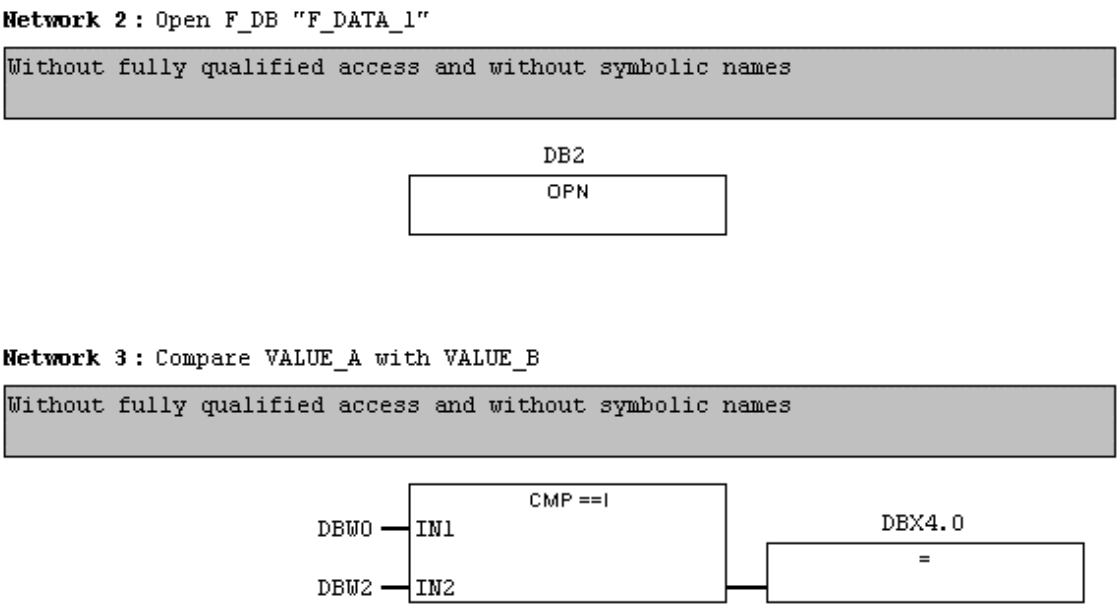
```
FB5 : Title:
Comment:

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

You have to assign a symbolic name for the F DB (e.g. "F_DATA_1") and use the
name assigned in the F DB declaration instead of the absolute address
```



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":



Instance DB Access

You can also access instance DBs of F-FBs with fully qualified access. It is not possible to access static data in instance DBs of other F-FBs.

Make sure that "Report cross accesses as faults" is not enabled 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 result in STOP mode in the F-CPU .

MOVE Instruction: Particularities

Note

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

For data from the standard user program, the length of the data types at 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" property.
- Static data of a multiple instance must not be accessed within the F-FB in which the multiple instance is declared.
- Inputs and outputs of a multiple instance must not be accessed outside the F-FB in which the multiple instance is declared.

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.

Illegal Instructions

All instructions other than those listed in the table above are **illegal**, 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 programmed 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 cannot interconnect, supply with "0," or evaluate the enable input EN or the enable output ENO.

5.2 Creating the Safety Program

5.2.1 Basic Procedure for Creating the Safety Program

Software Requirements

The software requirements are described in the section, Installing/Removing the S7 Distributed Safety V 5.3 Optional Package.

Additional Requirements

- A project structure must be created in the *SIMATIC Manager*.
- The hardware components of the project, particularly the F-CPU and the F-I/O, must have been configured prior to programming.
- The safety program must be assigned to a central processing unit, such as a CPU 315F-2 DP, with fail-safe capability.

Steps for Creating an S7 Distributed Safety Program

The most important 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 F-CPU if necessary.	*Configuration*
2	Defining the Program Structure	*Defining the Program Structure*
3	Create F-FBs and F-FCs with the F-FBD or F-LAD programming language in the *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 and F-FC to become an F-PB) • If the F-PB is a function block, assign an instance DB • Set maximum cycle time of F-runtime group • If one runtime group is to provide data for evaluation for another F-runtime group of the safety program, assign a DB for F-runtime group communication.	*Defining F-Runtime Groups* *Safety Engineering in SIMATIC S7* system description *Defining F-Runtime Groups*
6	Compile safety program in the "Safety Program" dialog box	*Compiling the Safety Program*
7	Call F-CALL blocks directly in OBs (time interrupt OBs, to the extent possible)	*Defining F-Runtime Groups*
8	Download entire user program (standard user program and safety program) in the "Safety Program" dialog box to the F-CPU	*Downloading the Safety Program*

5.2.2 Defining the Program Structure

Structure of Safety Program in Two F-Runtime Groups

Starting in *S7 Distributed Safety* V 5.3, you can divide your safety program into two F-runtime groups. By arranging for portions of your safety program (one F-runtime group) to run in a faster priority class, you achieve a faster safety circuit with short response times.

Note

You can improve the structure your safety program by dividing it into two F-runtime groups. However, note that the following actions cannot be performed for individual F-runtime 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 across all F-blocks of the safety program (see *Safety Program Acceptance Test*).

For additional information on F-runtime groups, refer to *Defining F-Runtime Groups*.

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-runtime groups.
- The safety program consists of one or two F-runtime 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 be accessed from only one F-runtime group.
- Variables of the F-I/O DB in 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).
- For optimal use of local data, you must call the F-CALL blocks (the F-runtime 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 box of the F-CPU. If you do not make any settings explicitly, meaningful default values are used (see Configuration).
- Create your program according to the general *STEP 7* rules. Consider, for example, the data flow.

You will find additional information in the *Differences between the F-FBD and F-LAD Programming Languages and the Standard FBD and LAD Languages* .

Note

You can improve performance by writing parts 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 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.

5.3 F-I/O Access

Overview

This section describes how to access the F-I/O and the special characteristics you must consider for access programming.

Access by Means of Process Image

As with standard I/O, F-I/O (e.g., S7-300 F-SMs) are accessed by means of the **process image** (PII and PIQ). The I/O cannot be accessed directly. The channels of an F-I/O can be accessed from only one F-runtime group.

The process input image is updated at the beginning of the fail-safe runtime group, before the fail-safe 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 *Structure of Safety Program in S7 Distributed Safety*).

The actual communication between the F-CPU (process image) and the F-I/O to update the process image is hidden and takes place by means of a special safety protocol in accordance with PROFIsafe.

Warning

Due to its special safety protocol, F-I/O occupy a larger area of the process image than required for the channels that are actually present on the F-I/O. When the process image is accessed in the safety program, only the actually existing channels can 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 set. In this case, only the less significant of the channels grouped by the "1oo2 evaluation of the sensors" can be accessed in the safety program.

Signal Chart Figures

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, ET 200S F-modules, ET 200eco F-modules, or fail-safe DP standard slaves, version of PROFIsafe bus profile for the F I/O).
- Cycle time of OBs of safety program
- Target rotation time of PROFIBUS DP

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 status charts, status changes between process 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.

5.3.1 Process Data or Fail-Safe Values

Use of Fail-Safe Values

The safety function requires fail-safe values (0) instead of process data for an F-I/O in the following cases (**passivation of an F-I/O**). This applies both to (digital) channels of data type BOOL and (analog) channels of data type INT (WORD):

- For startup of the F-system
- For errors in safety-related communication (communication errors) between the F-CPU and F-I/O using the safety protocol in accordance with PROFIsafe
- For F-I/O faults and channel faults (for example, wire break, short circuit, and discrepancy errors)
- As long as you enable passivation of the F-I/O with PASS_ON = 1 in the F-I/O DB (see below)

Fail-Safe Value Output for F-I/O

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

The F-system recognizes an overflow or underflow of a channel of the **SM 336; AI 6 x 13**-bit as an F-I/O fault or channel fault. The fail-safe value 0 is provided in place of 7FFFH (for overflow) or 8000H (for underflow) in the PII for the safety program.

If in the case of an F-I/O with inputs, you want to process other fail-safe values besides "0" in the safety program when fail-safe values are output, you can specify individual fail-safe values when QBAD = 1.

In an **F-I/O module with outputs**, the F-system transfers fail-safe values (0) to the fail-safe outputs during **passivation**, instead of the output values provided by the safety program in the PIQ.

Reintegration of an F-I/O

Transfer of 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 factors:

- Cause of passivation of the F-I/O
- Parameter assignment to be made by you in the F-I/O DB (see below)

Note

Keep in mind that when a channel fault occurs in the F-I/O, the fail-safe value (0) is output not only for the faulty channel, but for **all** channels of the affected F-I/O.

5.3.2 F- I/O DB

Introduction

An F-I/O DB is automatically generated 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 (one exception is the DIAG variable, which can only be evaluated in the standard user program). The initial values or current values of the variables cannot be changed directly in the F-I/O DB because the F-I/O DB is know-how protected.

Reasons to Access an F-I/O DB

You access variables of the F-I/O DB for the following reasons:

- For reintegration of F-I/O following communication errors, F-I/O faults, or channel faults
- If you want to passivate the F-I/O as a result of particular states of the safety program (for example, group passivation)
- For reassigning parameters for fail-safe DP standard slaves
- 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 reassigning fail-safe DP standard slave parameters	0
Variables that 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 reassigning fail-safe DP standard slave parameters	0
	DIAG	BYTE	Service information	

PASS_ON

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

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

See *Group Passivation*.

ACK_NEC

If an F-I/O fault or a channel fault is detected by the F-I/O, the affected F-I/O are **passivated**. Once the F-I/O fault or channel fault has been eliminated, the affected 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** by **user acknowledgment**.

See *Passivation and Reintegration of F-I/O Following F-I/O Faults or Channel Faults*.

Warning

The variable ACK_NEC can be set to 0 only if automatic reintegration for the relevant process is permissible from a safety standpoint.

Note

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

ACK_REI

When the F-system for an F-I/O detects a communication error or an F-I/O fault or channel fault, the affected F-I/O are passivated. **Reintegration** of the F-I/O after the error or fault has been eliminated requires a **user acknowledgment** with a positive edge at the ACK_REI variable of the F-I/O DB:

- After every communication errors
- After F-I/O faults or channel faults when parameter ACK_NEC = 1

Acknowledgment is only possible when ACK_REQ = 1.

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

IPAR_EN

The IPAR_EN variable corresponds to the iPar_EN_C variable in the PROFIsafe V1.2 bus profile.

If you have to set or reset this variable when parameters are reassigned for fail-safe
DP standard slaves, consult the PROFIsafe V1.2 bus profile or the documentation for the fail-safe DP standard slave.

Warning

Please note that starting with *S7 Distributed Safety* V 5.2, the affected F I/O are **not passivated** when IPAR_EN = 1.

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

PASS_OUT/QBAD

The variables PASS_OUT and QBAD indicate whether or not the F-I/O are passivated.

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

However, when **you** enable passivation by means of PASS_ON = 1, only QBAD = 1 is set. PASS_OUT does not change value in the event of passivation by means of PASS_ON = 1. For this reason, PASS_OUT can be used for group passivation of additional F-I/O (see *Group Passivation*).

ACK_REQ

When the F-system for an F-I/O detects a communication error or an F-I/O fault or channel fault, the affected F-I/O are passivated. ACK_REQ = 1 signals that **user acknowledgment** is required for reintegration of the affected F-I/O.

The F-system sets ACK_REQ = 1, as soon as the error or fault has been eliminated and user acknowledgment is possible. 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_EN variable corresponds to the iPar_OK_S variable in the PROFIsafe V1.2 bus profile.

To evaluate this variable when parameters are reassigned for fail-safe DP standard slaves, consult the PROFIsafe V1.2 bus profile or the documentation for the fail-safe DP standard slave.

DIAG

The DIAG variable provides non-fail-safe information (1 byte) for servicing purposes about errors or faults that have occurred. You can read out this information by means of operator control and monitoring systems or, if applicable, you can evaluate it in your standard user program. DIAG bits are saved until you execute an acknowledgment in ACK_REI or until automatic reintegration takes place.

Note

This variable cannot be accessed in the safety program.

Structure of DIAG

Bit no.	Meaning	Possible causes of problems	Remedies
Bit 0	Timeout detected by F-I/O	The PROFIBUS 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 PROFIBUS connection and ensure that there are no external sources of error. • 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 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	-	-

5.3.3 Accessing Variables of F-I/O DB

Symbolic Name of F-I/O DB

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

The symbolic name is made up of the fixed prefix "F," the start address of the F-I/O, and the names (maximum 17 characters) entered in the F-I/O object properties in *HW Config* (example: F00005_4_8_F_DI_DC24V). (See Assigning Symbolic Names.)

Rule for Accessing Variables of F-I/O DB

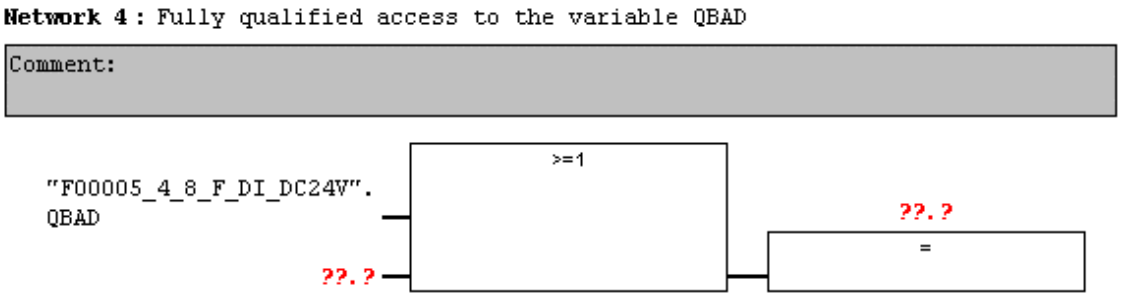
Variables of the F-I/O DB in an F-I/O can only be accessed from one F-runtime group and only from the F-runtime group from which the channels of this F-I/O are accessed (if access is made).

"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 accesses as faults" is not enabled 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



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

Behavior after Startup

After F-system startup, communication must first be established between the F-CPU and F-I/O using the PROFIsafe safety protocol. During this time, the F-I/O are **passivated**.

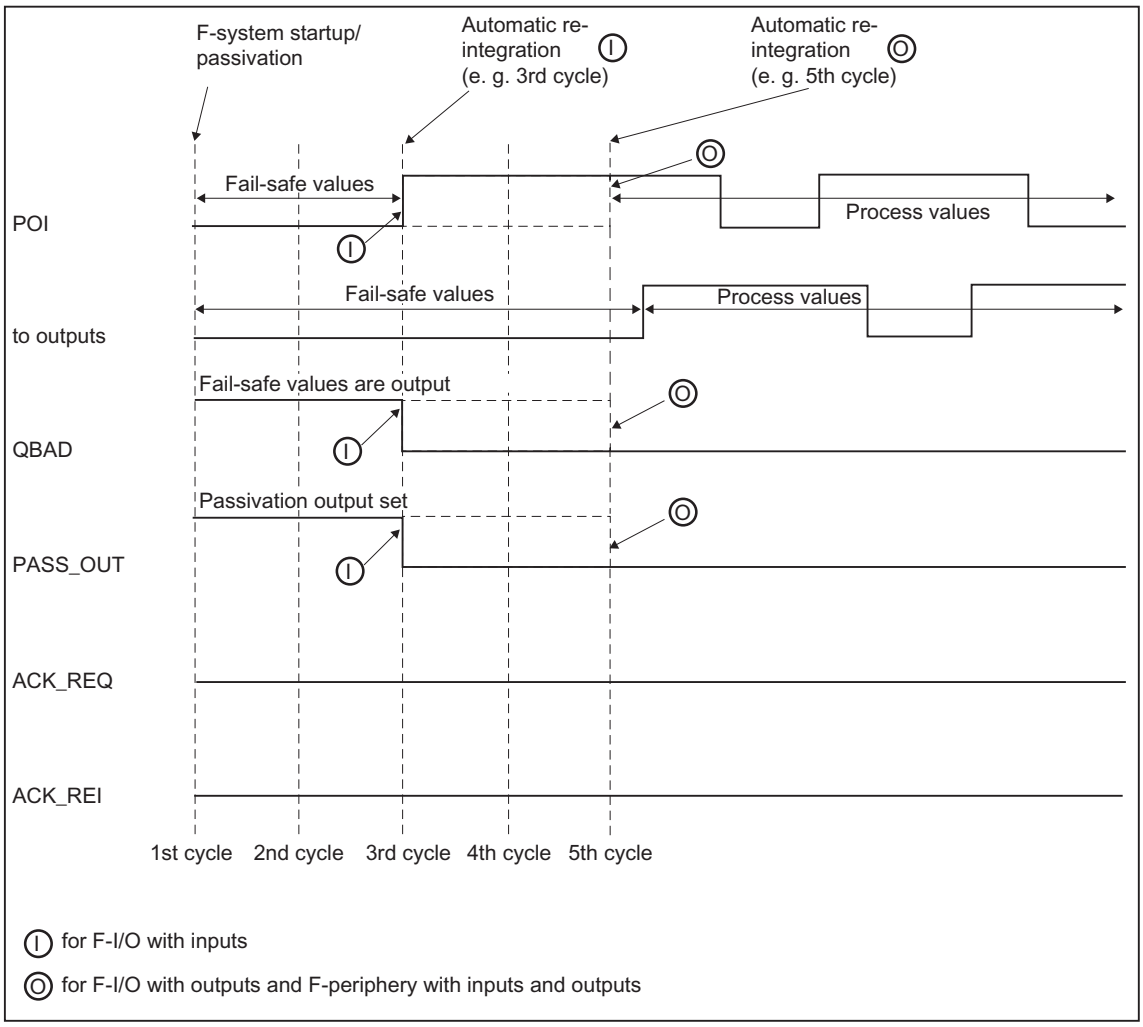
While fail-safe values (0) are being used, QBAD = 1 and PASS_OUT = 1.

Reintegration of F-I/O

Reintegration of the F-I/O, that is, providing of process data in the PII and transferring 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-runtime group after startup of the F-system; this happens regardless of the setting of the variable ACK_NEC. Depending on the F-I/O you are using and the cycle time of the F-runtime group and the PROFIBUS DP, several cycles of the F-runtime group can elapse before reintegration occurs.

If communication between the F-CPU and F-I/O takes longer to establish than the monitoring time set in the Object Properties dialog box of *HW Config*, automatic reintegration does not take place (see *Passivation and Reintegration of F-I/O after Communication Errors*).

Signal Chart for Passivation and Reintegration of F-I/O after F-System Startup



Note

If automatic reintegration does not take place after startup of the F-system, you must program startup protection (see *Programming Startup Protection*).

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

Behavior after Communication Errors

If the F-system detects an error in the safety-related communication (communication error) between the F-CPU and an F-I/O using the PROFIsafe safety protocol, **passivation** of the affected F-I/O takes place.

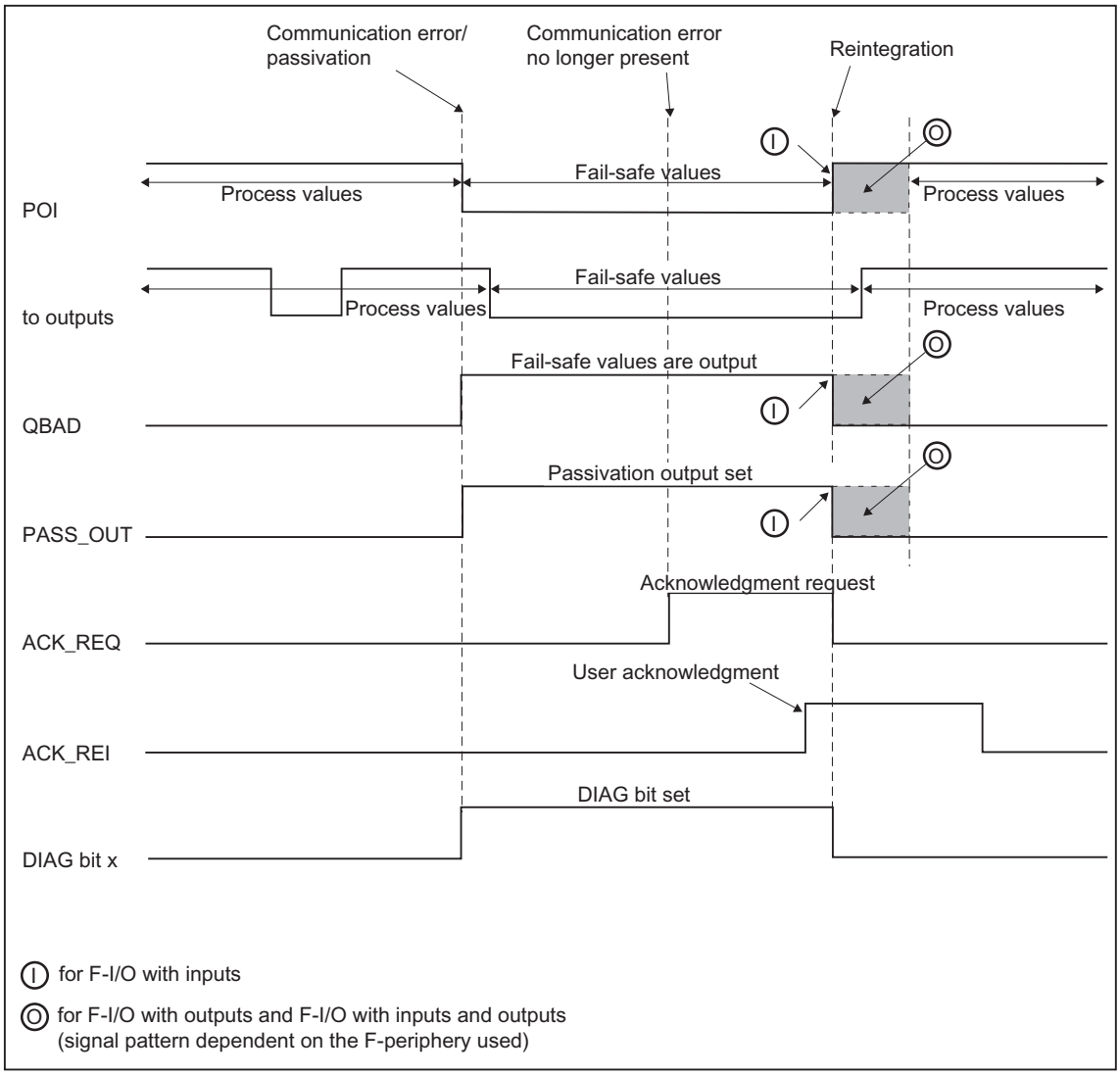
While fail-safe values (0) are being used, QBAD = 1 and PASS_OUT = 1.

Reintegration of F-I/O

Reintegration of the affected F-I/O, i.e., providing process data in the PII or transferring process data provided in the PIQ to the fail-safe outputs, takes place only after the following condition is met:

- All communication errors have been eliminated and the F-system has set ACK_REQ = 1
- A **user acknowledgment** takes place with a rising edge in the ACK_REI variable of the F-I/O DB (see *Implementing User Acknowledgment in Safety Program of F-CPU of DP-Master* or *Implementing User Acknowledgment in Safety Program of F-CPU of I-Slave*)

Signal Chart for Passivation and Reintegration of F-I/O after Communication Errors



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

Behavior after F-I/O Faults and Channel Faults

If the F-system detects an F-I/O fault or a channel fault (for example, wire break, short circuit, or discrepancy error), **passivation** of the affected F-I/O takes place.

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

Reintegration of F-I/O

Reintegration of the affected F-I/O, i.e., providing process data in the PII or transferring process data provided in the PIQ to the fail-safe outputs, takes place only after the following condition is met:

- All F-I/O faults and channel faults have been eliminated

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

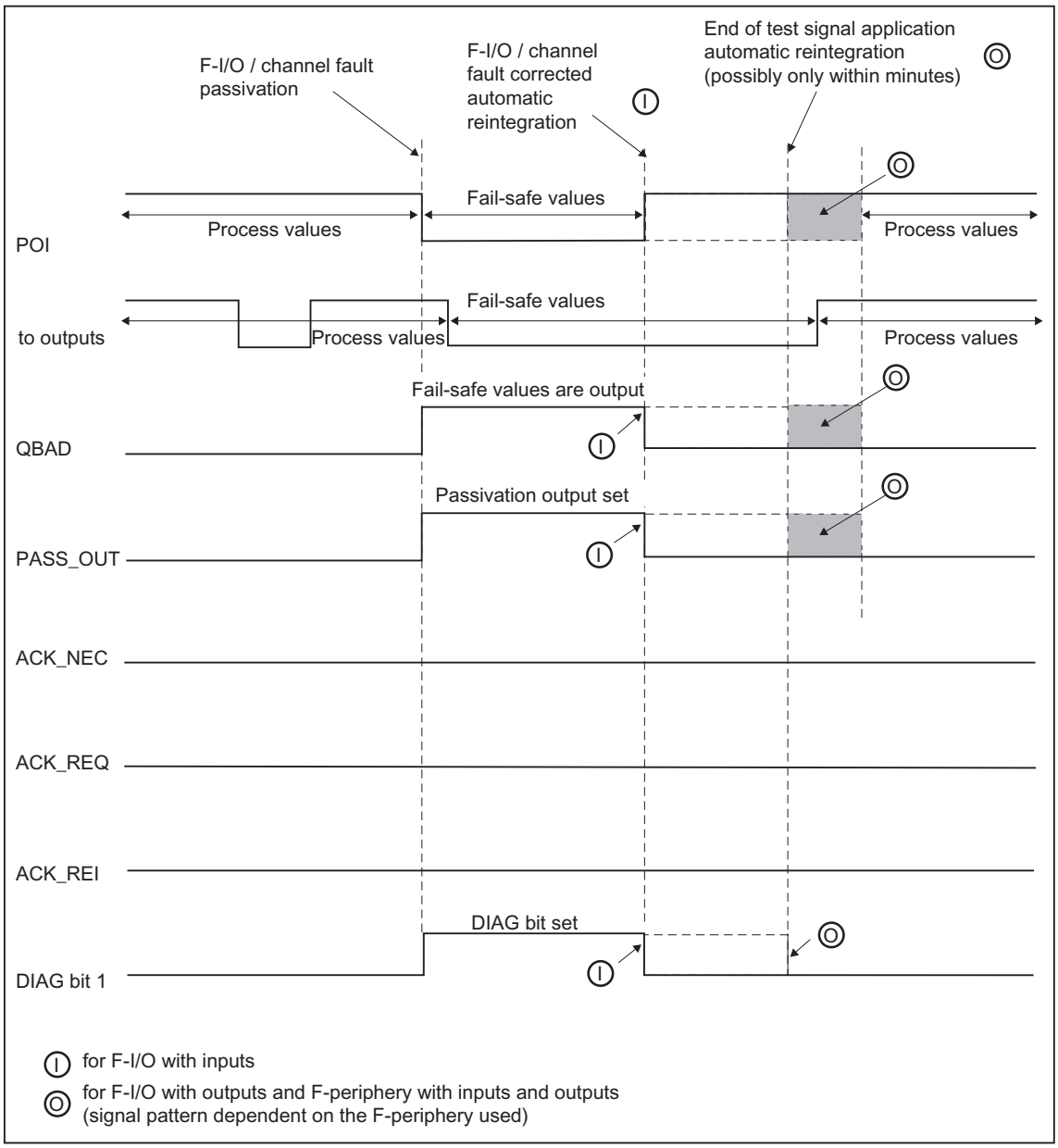
- When ACK_NEC = 0, **automatic reintegration** takes place as soon as the F-system detects that the fault has been eliminated. For F-I/O with inputs, reintegration takes place right away. For F-I/O with outputs, or inputs and outputs, depending on the F-I/O applied, reintegration can take place several minutes after completion of necessary test signal inputs used by the F-I/O to determine that the fault has been eliminated.
- When ACK_NEC = 1, reintegration does not take place until there is a rising edge in the ACK_REI variable of the F-I/O DB as a result of a **user acknowledgment** (see *Implementing User Acknowledgment in Safety Program of F-CPU of DP Master* or *Implementing User Acknowledgment in Safety Program of F-CPU of I-Slave*) Acknowledgment is only possible after the F-system detects that the fault has been eliminated and after it has set ACK_REQ = 1.

Warning

Following a power failure in the F-I/O that is shorter than the monitoring time set for the F-I/O in *HW Config* (see *Safety Engineering in SIMATIC S7* system description), automatic reintegration can occur, as is the case when ACK_NEC = 0, regardless of your setting for ACK_NEC.
If automatic reintegration for the affected process is not permitted for this case, you must program startup protection by evaluating the variables QBAD or PASS_OUT (see *Programming Startup Protection*).

When a power failure occurs in the F-I/O that lasts longer than the monitoring time set for the F-I/O in *HW Config*, the F-system detects a communication error (see *Passivation and Reintegration of F-I/O after Communication Errors*).

Signal Chart for Passivation and Reintegration of F-I/O after F-I/O Faults or Channel Faults When ACK_NEC = 0



Signal Chart for Passivation and Reintegration of F-I/O after F-I/O Faults or Channel Faults When ACK_NEC = 1

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

5.3.7 Group Passivation

Programming Group Passivation

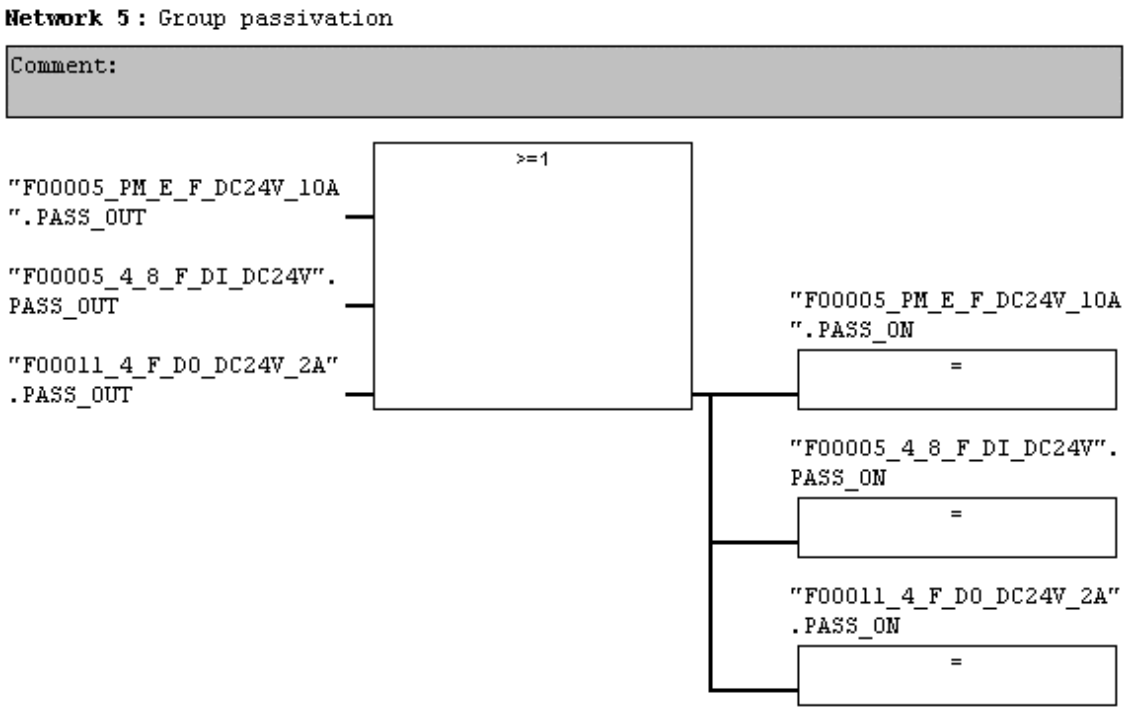
If you want passivation of additional F-I/O to occur during passivation of one F-I/O by the F-system, you can use the variables PASS_OUT/PASS_ON to carry out **group passivation** of associated F-I/O.

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 variables of the F-I/O in the group are set to 1.

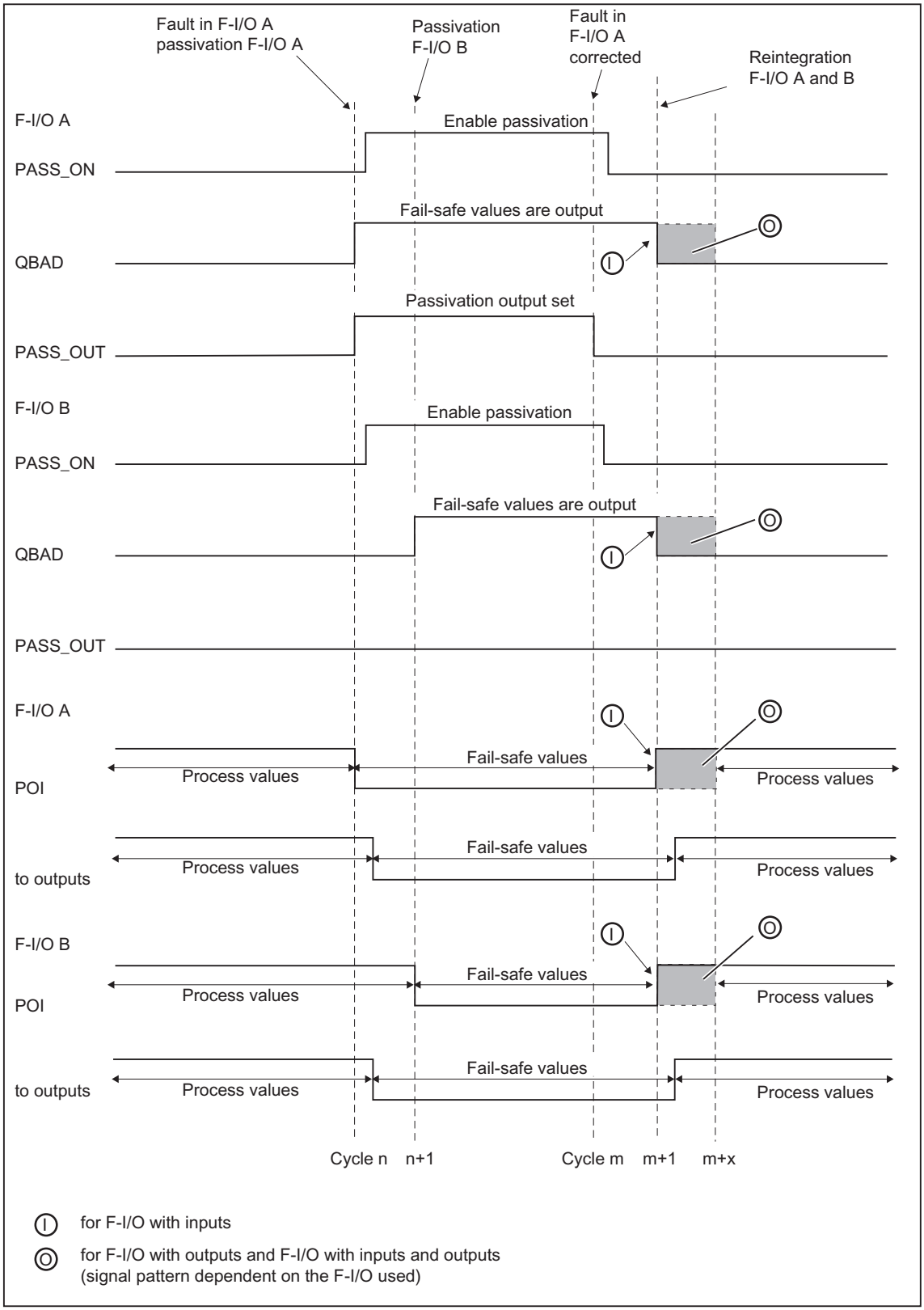
Example of Group Passivation



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



5.3.8 Implementing User Acknowledgment in Safety Program of F-CPU of DP Master

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 acknowledgment by means of an operator control and monitoring system for acknowledgment 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) (see *FB 187 "F_ACK_OP": Fail-Safe Acknowledgment*).

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 pass 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 can only carry out user acknowledgment from one programming device/PC using the "Monitor/Modify Variable" function, and you do not want to deactivate safety mode, then you must transfer an address (flag 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/PC by modifying the flag word. The flag word cannot be described by the program.

Note

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

Warning

The two acknowledgment steps **cannot** 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.
Having two separate acknowledgment steps also prevents faulty triggering of an acknowledgment by your non-fail-safe operator control and monitoring system.

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 name for the F-CPU that is unique for the entire network 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 online from the DB 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 easily determine whether the intended F-CPU is being addressed by comparing the F-CPU name read out online with the permanently stored name.

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

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

Warning

The variable ACK_NEC can be set to 0 only if automatic reintegration for the relevant process is permissible from a safety standpoint. See description of the ACK_NEC variable in *F-I/O DB*.

2. Optional: Evaluate QBAD or DIAG 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 to your operator control and monitoring system in your standard user program by evaluating QBAD or DIAG; these messages can be evaluated before the acknowledgment process is carried out.
3. Optional: Evaluate the variable ACK_REQ 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. In the respective F-I/O DB, assign the input of the acknowledgment key or output OUT of F_ACK_OP to ACK_REI (see above).

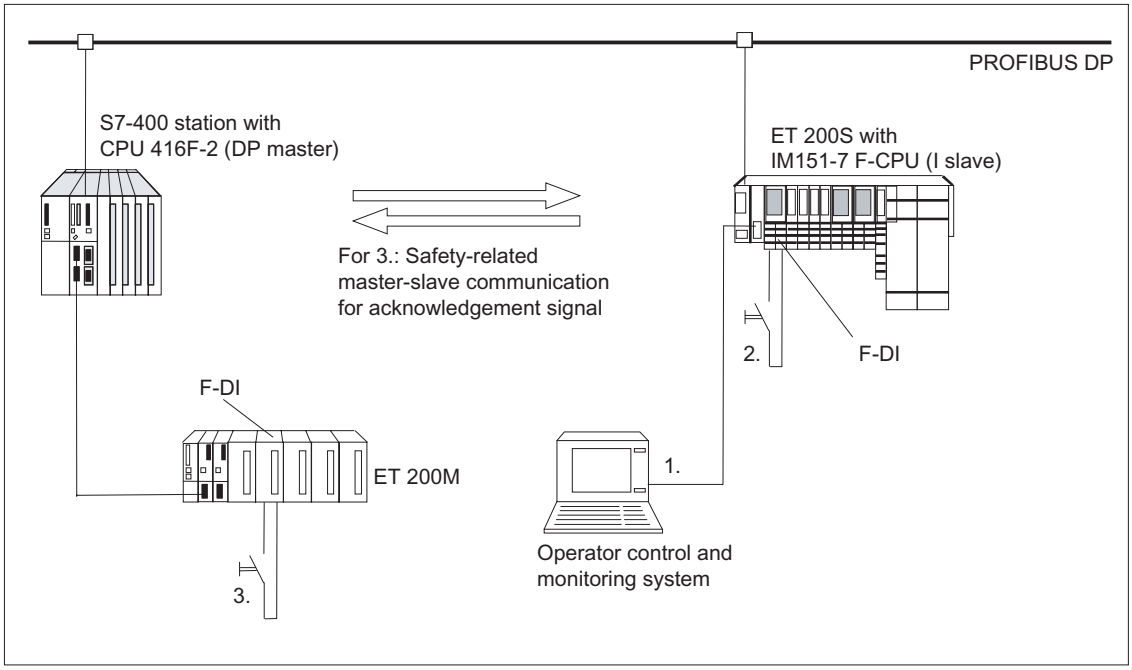
5.3.9 Implementing User Acknowledgment in Safety Program of F-CPU of I-Slave

Options for User Acknowledgment

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

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

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) (see *FB 187 "F_ACK_OP": Fail-Safe Acknowledgment*).

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

Follow the procedure described in *Procedure for Programming User Acknowledgment by Means of an Operator Control and Monitoring System* in *Implementing a 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 on 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

If a communication error or an F-I/O fault or channel fault occurs 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/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 to enable acknowledgment for reintegration of an F-I/O to which an acknowledgment key is connected. (see option 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 (see *Configuring Safety-Related CPU-CPU Communication*).

Procedure for Programming User Acknowledgment by Means of an Acknowledgment Key at an F-I/O with Inputs Assigned to the F-CPU of the DP Master

1. Call the F_SENDDP F-application block in the safety program on the F-CPU of the DP master
   (see *FB 223 "F_SENDDP" and FB 224 "F_RCVDP": Sending and Receiving Data via PROFIBUS DP*).
2. Call the F_RCVDP F-application block in the safety program on the F-CPU of the I-slave (see *FB 223 "F_SENDDP" and FB 224 "F_RCVDP": Sending and Receiving Data via PROFIBUS*).
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 block.
   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 block with the fail-safe value "RLO0," so that an unintended 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 provided in the F-shared-DB. At input SUBBO_xx, enter "F_GLOBDB".RLO0 fully qualified.

Note

If a communication error or an F-I/O fault or channel fault occurs 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/RUN change 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 to enable acknowledgment for reintegration of F-I/O to which an acknowledgment key is connected (see *Implementing User Acknowledgment in Safety Program of F-CPU of DP Master*).

If a safety-related master to I-slave communication error occurs, the acknowledgment signal cannot be transferred, and an acknowledgment for reinclusion in the safety-related communication is no longer possible.

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

To allow acknowledgment and reinclusion in safety-related communication, it is therefore advisable to plan a further acknowledgment over an operator control and monitoring system, with which you can access the F-CPU of the I-slave and transfer an acknowledgment signal (see 1.).

5.4 Programming Safety-Related CPU-CPU Communication

Overview

This section describes how to program safety-related communication between safety programs on different F-CPUs:

- 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 communication by means of S7 connections (via Industrial Ethernet)

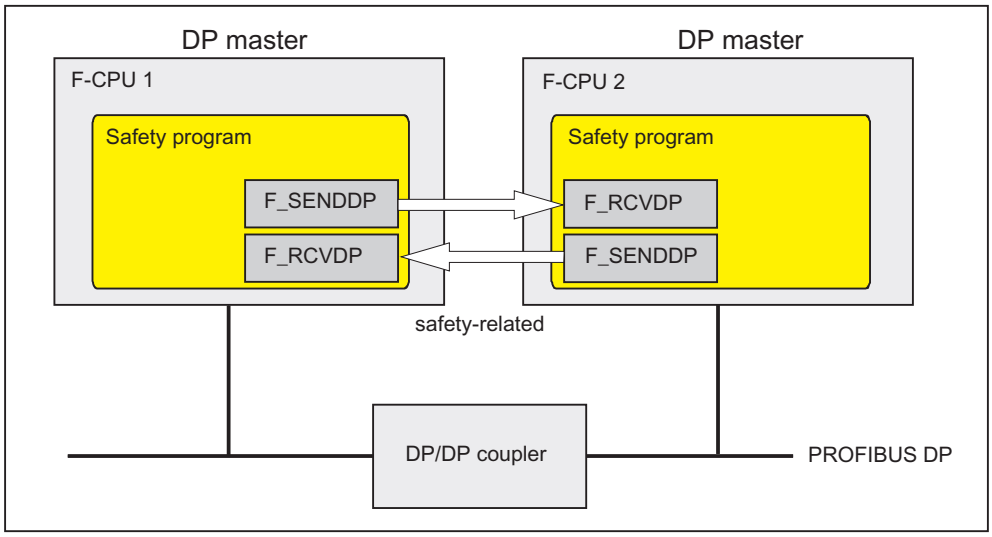
5.4.1 Programming Safety-Related Master-Master Communication

DP/DP coupler

Safety-related communication between safety programs on different F-CPUs takes place over a DP/DP coupler (Order No. 6ES7158-0AD01-0XA0).

Each of the two F-CPUs is connected to the DP/DP coupler by means of its PROFIBUS DP interface.

Communication by Means of F_SENDDP and F_RCVDP



Safety-related communication is carried out with the aid of 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 m 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, and the F_SENDDP at the end of the F-PB.

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

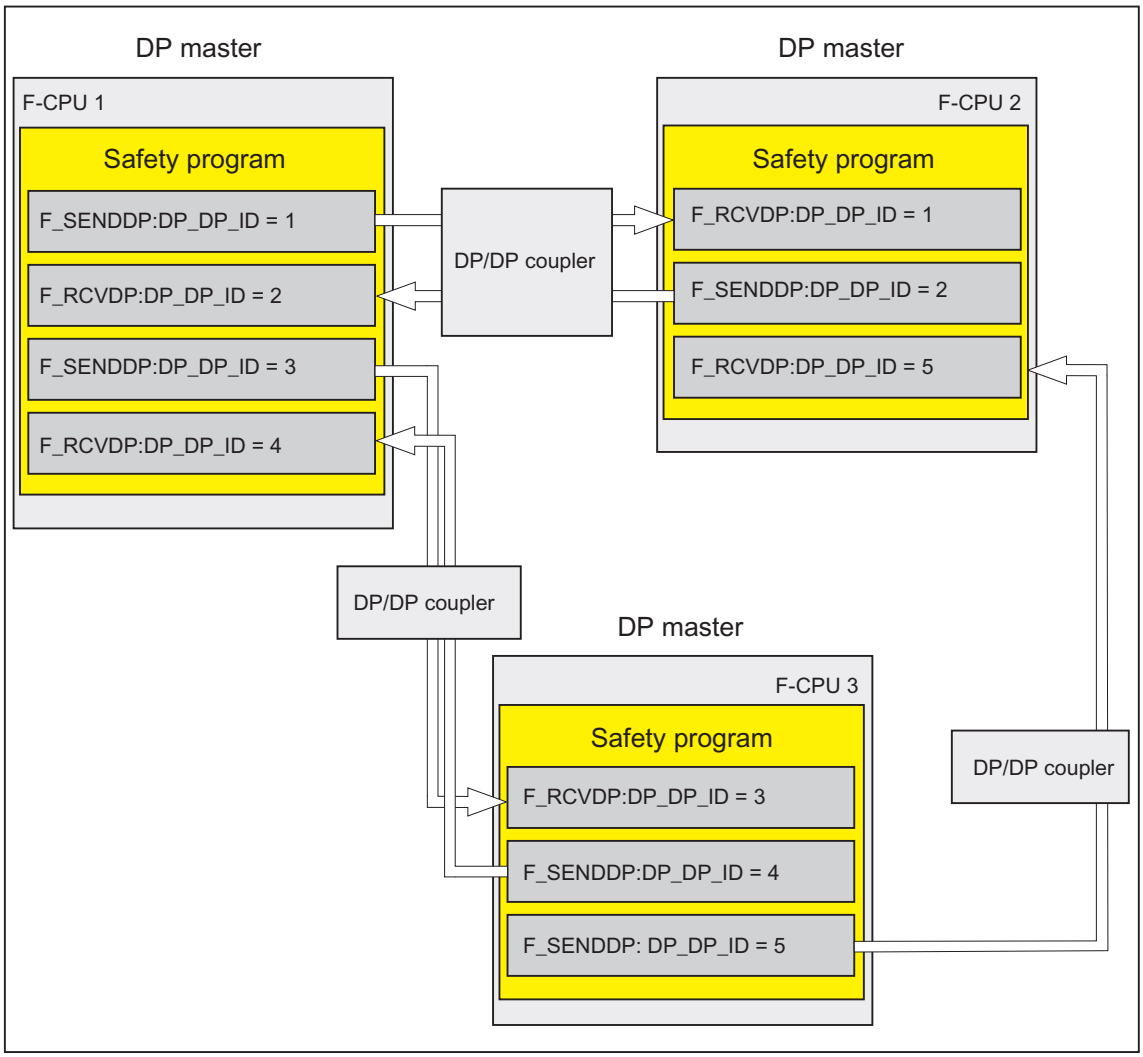
Requirements for Programming

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* (see Section *Configuring Safety-Related Master-Master Communication*)
- Both CPUs must be configured as F-CPUs:
  - "CPU contains safety program" option must be activated
  - The password for the F-CPU must be entered

Programming Procedure

1. In the safety program from which data are to be sent, call F-application block F_SENDDP to send data at the end of the F-PB.
2. In the safety program from which data are to be received, call F-application block F_RECVDP to receive data at the beginning 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 (see *Configuring Safety-Related Master-Master Communication).*
   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 an association between F_SENDDP in an F-CPU and an F_RCVDP in another 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 block.

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 diagnostics events will then be entered in the diagnostics 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 inputs SD_BO_xx of the F_SENDDP with the send signals.
   To economize intermediate signals when transferring block parameters, you can write the value directly to the instance DB of the F_SENDDP by means of symbolic, fully qualified access (for example, Name F_SENDDP1".SD_BO_02) before calling the F-SENDDP.
6. Provide outputs RD_BO_xx of the F_RCVDP with the signals that you would like to process further in other parts of the program.
   Alternatively, use fully qualified access to read the received signals in the parts of the program in which additional processing is to take place directly in the associated instance DB (for example, "Name F_RCVDP1".RD_BO_02).
7. Provide inputs SUBBO_xx and SUBI_xx of the F_RCVDP with the fail-safe values that should be output by F_RCVDP in place of the process data until communication is established for the first time after the sending and receiving F-system is started up, 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".VKE0 with fully qualified access if you want to specify a fail-safe value of "0", and "F_GLOBDB".VKE1 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

Keep in mind that to dynamically change a variable for a dynamic fail-safe value, your safety program can only be processed after F_RCVDP has been called, because prior to an F_RCVDP call, there can be no network in the F-PB; at most, the F-DB can contain one additional F_RCVDP. You must therefore assign appropriate initial/actual values for these variables to be output by F_RCVDP in the first cycle after 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 is recorded on the sender end and transferred to the receiver only if the signal is pending for at least as long as the configured monitoring time (TIMEOUT).
You will find information about the calculation of F-system monitoring times in the *Safety Engineering in SIMATIC S7* system description.

9. Optional: Evaluate output ACK_REQ of the F_RCVDP, 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.
10. Supply the ACK_REI input of F_RCVDP with the signal for acknowledgment for reintegration (see *Implementing User Acknowledgment in Safety Program of F-CPU of DP Master*).
11. Optional: Evaluate output SUBS_ON of the F_RCVDP or the F_SENDDP, to query whether the F_RCVDP is outputting the fail-safe values programmed at inputs SUBBO_xx and SUBI_xx of the F_RCVDP.
12. Optional: Evaluate output ERROR of the F_RCVDP or the F_SENDDP, for example, in the standard user program or on the operator control and monitoring system, to query or to indicate whether a communication error has occurred.
13. Optional: Evaluate output SENDMODE of the F_RCVDP, 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 by the 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 *Deactivating Safety Mode*).

Note

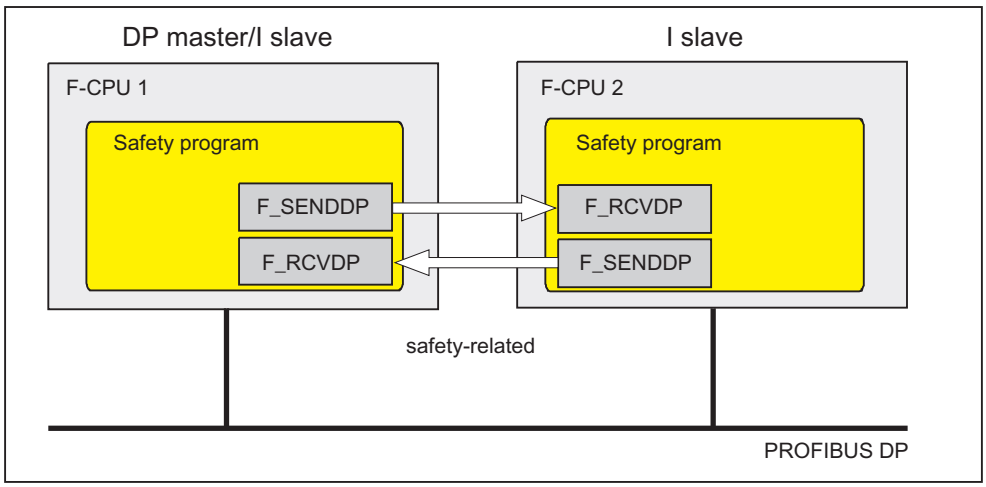
If the data quantities to be transmitted exceed the capacity of the F_SENDDP/F_RCVDP block pair, a second (or third) F_SENDDP/F_RCVDP call can be used. This requires configuration of an additional connection via the DP/DP coupler. Whether or not this is possible with one single DP/DP coupler depends on the capacity restrictions of the DP/DP coupler.

5.4.2 Programming Safety-Related Master-I-Slave Communication and I-Slave-I-Slave Communication

Overview

You use the same procedure for programming safety-related master-I-slave communication and safety-related I-slave-I-slave communication as for programming safety-related master-master communication (see *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 use 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 m 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, and the F_SENDDP at the end of the F-PB.

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

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* (see *Configuring Safety-Related Master-I-Slave Communication* and *Configuring Safety-Related I-Slave-I-Slave-Communication*)
- Specify the following 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 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

The settings for safety-related master to I-slave and I-slave to I-slave communication are always as follows:

- For 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).
- For 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).

Requirements for Programming

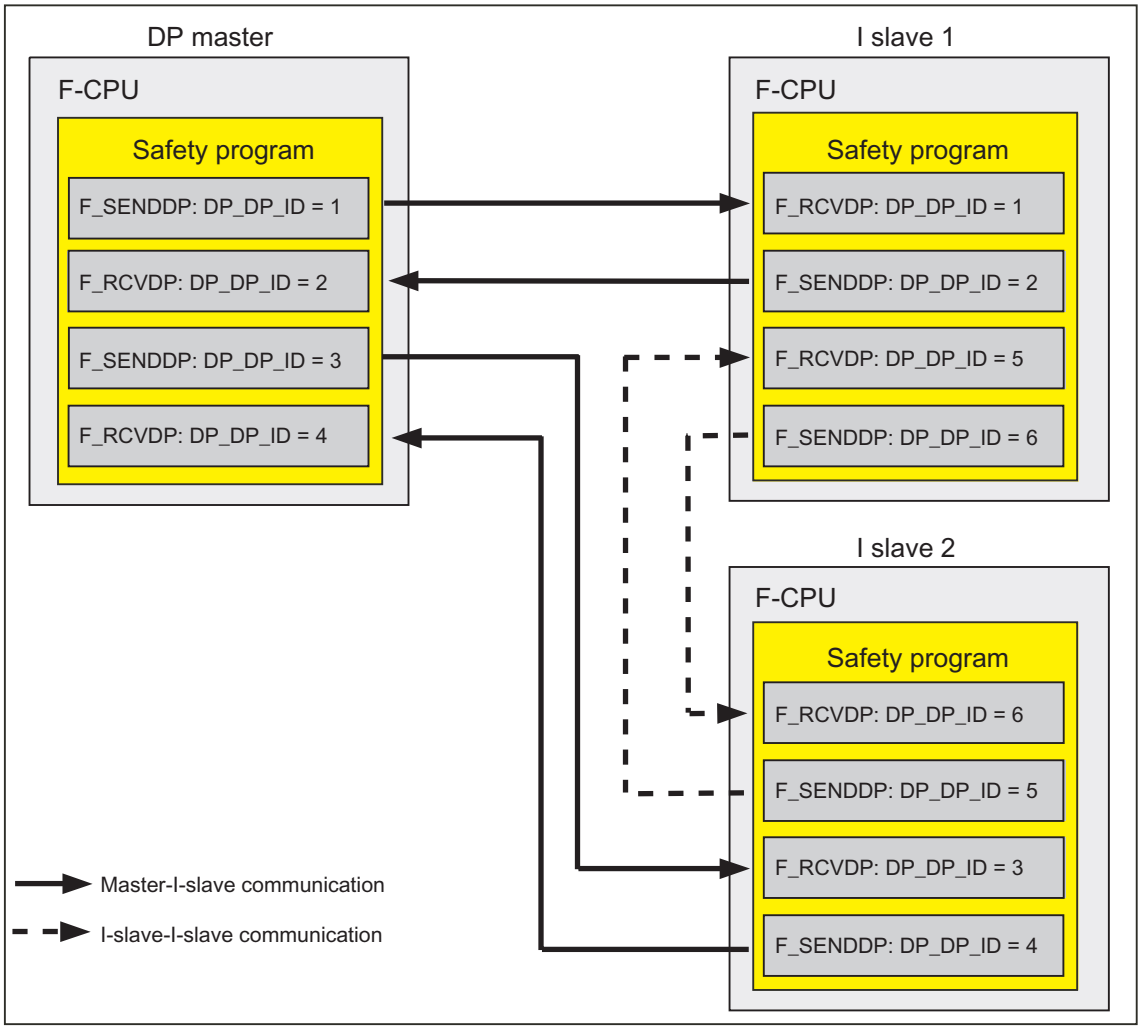
The following requirements must be met prior to programming:

- The address areas (local and partner addresses) for the DP master and I-slave(s) must be configured in *HW Config* (see *Configuring Safety-Related Master-I-Slave Communication* and *Configuring Safety-Related I-Slave-I-Slave-Communication*)
- Both CPUs must be configured as F-CPUs:
  - "CPU contains safety program" option must be activated
  - The password for the F-CPU must be entered

Programming Procedure

The procedure for programming safety-related master to I-slave communication or I-slave to I-slave communication is exactly the same as for programming safety-related master to master communication. The information in *Programming Safety-Related Master-Master Communication* is applicable.

The schematic 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 and one I-slave-I-slave communication relations.



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 block.

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 diagnostics events will then be entered in the diagnostics 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 by the 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 *Deactivating Safety Mode*).

Limits for Data Transfer

If the amount of data to be transmitted is greater than the capacity of an F_SENDDP/F_RCVDP block pair, you can use additional F_SENDDP/ F_RCVDP calls. These require further communication connections. 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 how many output and input data are required for safety-related communication connections:

Safety-related Communication	**Communication Connection**	**Required 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 to 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	-

Take into account 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 and, if applicable, master-slave connections (MS) or direct data exchange connections (DX) used to exchange data within your standard user program.

You can check whether you are within the maximum limit of 244 bytes of input and 244 bytes of output data for all the configured safety-related and standard communication connections in the "Configuration" tab in the object properties of the I-slave. Include all rows with MODE "MS" in the "Configuration" tab. The rows with MODE "DX" are not included.

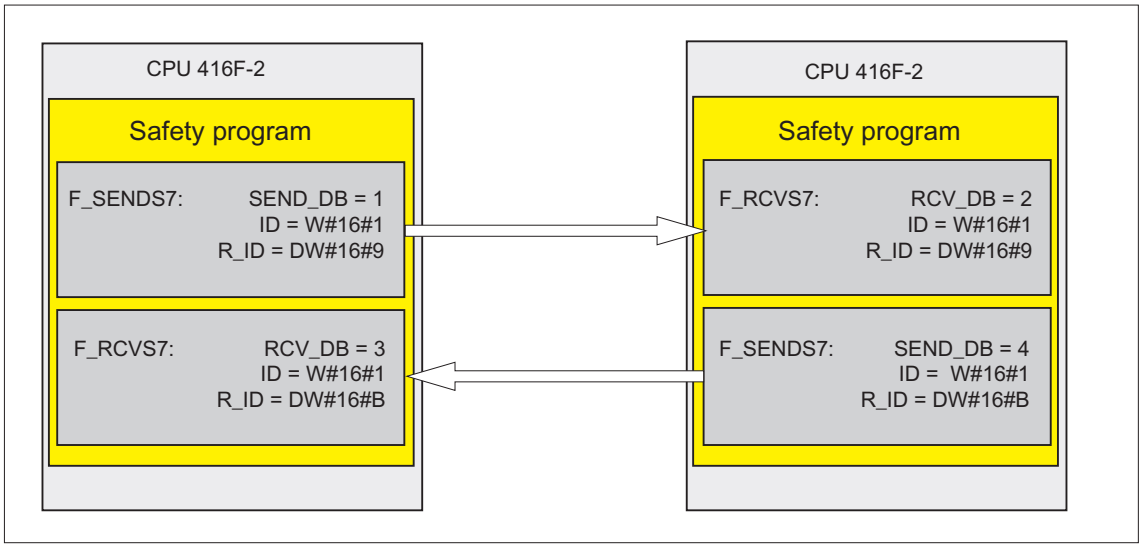
5.4.3 Programming Safety-Related CPU-CPU Communication via S7 Connections

Overview

This section describes how to program safety-related communication between safety programs on 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/receive data for communication are stored
- Call and assign parameters to F-application blocks for communication from the *Distributed Safety* F-library (V1) in the safety program

Communication using F_SENDS7 and F_RCVS7



You use the **F_SENDS7 and F_RCVS7** F-application blocks for sending and receiving data in a fail-safe manner 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 m 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, and the F_SENDS7 at the end of the F-PB.

For a detailed description of F-application blocks, refer to the section, *FB 225 "F_SENDS7",FB 226 F_RCVS7": Communication via S7 Connections*.

Note

In S7 Distributed Safety V 5.3, S7 connections are generally permitted over Industrial Ethernet only!

Safety-related communication via S7 connections to and from CPUs 416F-2 is only possible with **firmware version V 3.1.4** or later.

F-Communication DB

For each connection, send data are receive data are each stored in one F-DB (F-communication DBx and F-communication DBy, respectively).

The F-communication DB numbers are provided to the F_SENDS7 or F_RCVS7 as parameters.

Requirements for Programming

The following requirements must be met prior to programming:

- S7 connections between relevant F-CPUs must be configured in *NetPro* (see *Configuring Safety-Related Communication via S7 Connections*)
- Both CPUs must be configured as F-CPUs:
  - "CPU contains safety program" option must be activated
  - 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 the same way as other F-DBs in *SIMATIC Manager* (see *Creating and Editing an F-DB*).

Note the following when creating F-communication DBs:

Assign the "COM_DBS7" identifier in the "Family" field in the "General – part 2" tab of the F-DB object properties. 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 end must match the length and structure of the associated F-communication DB on the sender end.

If the F-communication DBs do not match, the F-CPU can go to STOP mode. One of the following diagnostics events will then be entered in the diagnostics 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 end 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 end, 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.
- The maximum permitted amount of data of data type BOOL is 128.
- 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 properties are not fulfilled, *S7 Distributed Safety* outputs an error message.

Assigning Fail-Safe Values

Fail-safe values are provided by the receiver end:

- While the connection between the communication partners is being established the first time after startup of F-systems
- Whenever a communication error occurs

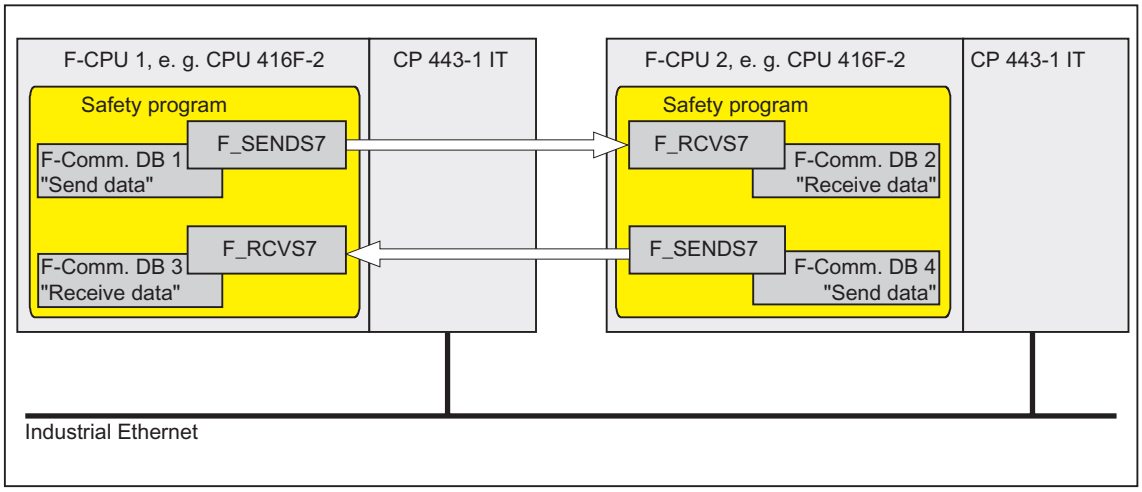
The values you specified in the F-communication DB on the receiver end are provided as fail-safe values (default of F-communication DB).

Programming Procedure

1. Supply the variables in the F-communication DB on the sender end 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 on the receiver end (receive signals) that you want to process 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 F-application block F_SENDS7 to send data at the end of the F-PB.
4. In the safety program from which data are to be received, call F-application block F_RCVS7 to receive data at the beginning of the F-PB.
5. Assign the applicable F-communication DB numbers to the inputs SEND_DB of F_SENDS7 and RCV_DB 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 *(*see *Configuring Safety-Related Communication via S7 Connections*).
7. Assign the local ID of the S7 connection (data type: WORD) configured in *HW Config* to the input ID of F_RCVS7 (see *Configuring Safety-Related Communication via S7 Connections).*
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 odd and unique from all other safety-related communication connections in the network. The value R_ID + 1 is internally assigned and must not be used.



Note

A separate instance DP must be used for each call of an F SENDS7 or F_RCVS7 block.

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 or another F_RCVS7 or F_RCVDP call. The F-CPU can go to STOP if this is not observed. One of the following diagnostics events will then be entered in the diagnostics 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 is recorded on the sender end and transferred to the receiver only if the signal is pending for at least as long as the configured monitoring time (TIMEOUT).
You will find information about the calculation of F-system monitoring times in the *Safety Engineering in SIMATIC S7* system description.

10. To reduce the bus load, you can temporarily disable 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 in its F-communication DB). If communication was already established between the partners, a communication error is detected.
11. Evaluate output ACK_REQ of the F_RCVS7, 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.

    Supply input ACK_REI of F_RCVS7 with the signal for acknowledgment for reintegration (see Implementing User Acknowledgment in Safety Program of F-CPU of DP Master or Implementing User Acknowledgment in Safety Program of F-CPU of I-Slave).
12. Evaluate output SUBS_ON of F_RCVS7 or F_SENDS7 to query whether the F_RCVS7 is outputting fail-safe values that you have specified as the default in the F-communication DB.
13. Evaluate output ERROR of the F_RCVS7 or F_SENDS7, for example, in the standard user program or on the operator control and monitoring system, to query or to indicate whether a communication error has occurred.
14. Evaluate output SENDMODE of the F_RCVS7, 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 by the 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 *Deactivating Safety Mode*).

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.

Please note that the SFBs 8 and 9 are internally called for each F_SENDS7 or each F_RCVS7 call and use connection resources in the F-CPU. This affects the maximum number of communication connections available. Similar to the standard program, information about the connection resources of an F-CPU is available in the "Module Status" dialog of the "Communication" tab.

5.5 Communication between Standard User Program and Safety Program

Data Transfer from Safety Program to Standard User Program

The standard user program can read out all of the data from the safety program, for example, through symbolic (fully qualified) access to the following:

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

Note

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

To find out the time at which the process input and output images are updated by the standard operating system, refer to "Process image of inputs/outputs" in the *online Help for STEP 7.* With the S7-400, bear in mind the update timing when using process image partitions. 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 timing
- 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 can therefore only access the process input image in the standard program after execution of an F-runtime group. In this case, you can also evaluate the QBAD variable in the relevant F-I/O DB in the standard user program to find out whether the process input image contains fail-safe values (0) or process data. When using process image partitions (S7-400 only), 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

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

- 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)
- Compile data 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 address of the variables are indicated in the printout of the safety program.

Note

Starting with *S7 Distributed Safety* V 5.2, the collective signature of the safety program ("F_PROG_SIG" variable) is output in the F-shared-DB as a double word.

If you previously used *S7 Distributed Safety* V 5.1 and read out the "F_PROG_SIG" variable in the safety program or by means of an operator control and monitoring system and you now convert to *S7 Distributed Safety* V 5.3, you must change the data type to DWORD for evaluation, if necessary.

Warning

Do not copy the F-shared DB from a safety program to another safety program (exception: copying the entire S7 program).

Memory Marker

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

Process Output Image

In the safety program, you can also write to the process output image (PIQ) of standard I/O, for display purposes, for example. These values cannot be read in the safety program, either (see also table of supported address areas in *Differences between the Programming Languages F-FBD/F-LAD and the Standard Languages FBD/LAD).*

Data Transfer from Standard User Program to Safety Program

Only fail-safe data or fail-safe signals from fail-safe I/O and other safety programs (in other F-CPUs) can be generally 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) of standard I/O in the safety program (see also table of supported address areas in *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 expressed 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

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

Programming Validity Checks

Examples:

- 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.
- Only allow a motor to be switched off, but not to be switched on, with unsafe signals from the standard user program, for example, 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 that Can Change during Runtime of an F-Runtime Group

You must use dedicated memory bits if, in the safety program, you want to read data from the standard user program (bit memory or PII of standard I/O) 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 bit memory** that you defined when configuring your F-CPU (in *HW Config,* in the Object Properties dialog box of the F-CPU) can change during runtime of the F-runtime group, since clock bit memory runs asynchronously to the F-CPU cycle.

Note

If you do not take into account the information provided above, the F-CPU can go to STOP mode. One of the following diagnostics events will then be entered in the diagnostics 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"

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

Overview

This section describes how to create a safety program in F-FBD/F-LAD using F-FBs/F-FCs/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 the *SIMATIC Manager* in *"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 regardless of the hardware used. However, keep in mind that F-addresses and the validity of F-I/O access are not checked in this case.

5.6.1 Creating and Editing an F-FB/F-FC

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

1. Go to the block container of the *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; see *Configuring 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.
   The block symbol will be displayed in the *SIMATIC Manager* with a yellow background.
3. The created F-block can then be opened and edited with the *FBD/LAD editor*: Double click the F-FB/F-FC in *SIMATIC Manager*.
4. Enter the password for the safety program (password protection queries will no longer be mentioned in operating procedures below (see *Access Protection*).
   The *FBD/LAD editor* will open.
   You should enable "Type test 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 (see *Differences between the F-FBD/F-LAD Programming Languages and the Standard FBD/LAD Languages*)
- 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-blocks

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

   When you edit, the data types are checked in incremental mode. Any errors detected are output in the *FBD/LAD editor,* as is the case for programming the standard user program.

Note

An F-FB/F-FC called in F-CALL (which causes it to become an F-PB) cannot have any parameters because they cannot be assigned (see *Defining F-Runtime Groups*).

Note

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

Note

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

Rule: there can be no STL networks in F-FBD representation. STL networks in F-FBD must be represented again as F-FBD when they are converted to F-FBD.

Warning

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

Note

Accessing static parameters of instance DBs of other F-FBs is not permitted.

Note

When using F-FCs, note that write access must be used when output parameters of F-FCs are accessed the first time. 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 diagnostics events will then be entered in the diagnostics 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 a formal parameter of an F-FC as the actual parameter to an address from the data area (data block), you need to use fully qualified DB access.

Note

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

Note

Note that you can only read the input parameters in an F-FB/F-FC and only write to its output parameters.

Use an input/output parameter if you wish to read and write.

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, preconfigured interconnections with edge memory bits and network access junctions) when you try to save the F-block. Such F-blocks cannot be saved. You must delete the STL network and replace the interconnections with your own networks, in which you direct the 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 the *SIMATIC Manager*, in the "Safety Program," and in the symbol table. Symbolic names are assigned in the same way as in the standard case.

"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 for a standard system

Note

The "Check block consistency" function is not sufficient for obtaining a consistent safety program. Rather, you must compile the safety program (see *Compiling Safety Program*).

5.6.2 Creating and Editing an F-DB

F-DBs

Similarly to F-FBs/F-FCs, you can also create and edit F-DBs (with the F-DB programming language), for which read and write access of parameters is possible within one F-runtime group of the safety program.

When you edit, the data types are checked in incremental mode. Any errors detected are output in the *FBD/LAD editor,* as is the case for programming the 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 *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 compiled.

Note

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

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

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

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

If you have selected one of these options, the selection is 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 one F-communication DB each on the sender and receiver ends. 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 *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 one "DB for F-runtime group communication" for each 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 *Defining F-Runtime Groups*.

"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 for a standard system

The "Check block consistency" function is not sufficient for obtaining a consistent safety program. Rather, you must compile the safety program (see *Compiling Safety Program*).

5.6.3 Know-How Protection for User-Created F-FBs and F-FCs

Know-How Protection

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

Starting with *S7 Distributed Safety*, V 5.2 + SP 1, you can provide user-created F-FBs and F-FCs with know-how protection.

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

From protected F-FBs/F-FCs, you can merely view the block properties and the variable declaration section with input parameters, output parameters and in/out parameters; the instruction section remains hidden.

The variable declaration table of the F-FB/F-FC displays the same variable declaration types as for standard blocks with know-how protection.

Using Know-How Protection

Use know-how protection when you want to protect the knowledge contained in an F-FB/F-FC and want to prevent unwanted manipulation of the F-FBs and F-FCs.

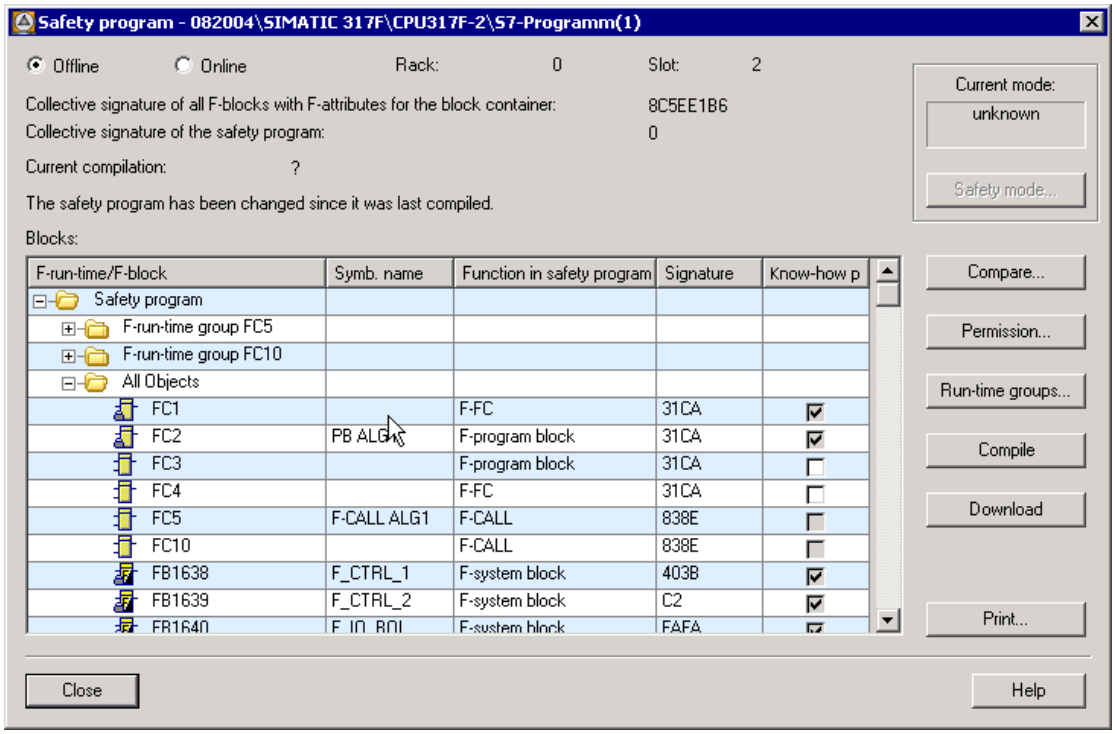
Setting Know-How Protection

Requirements:

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

Follow the steps outlined below:

1. Open the "Safety Program" dialog in the *SIMATIC Manager*.
2. You set know-how protection for F-FBs/F-FCs in the off-line safety program. Therefore, select "Offline".



3. Check the relevant check box in the "Know-how protection" column for the F-FBs and F-FCs.
   Result: A dialog for creating a backup copy opens automatically for every F-FB/F-FC you want to protect.
4. Remember the following when you save the backup copy:

Note

Assign a unique name for the backup copy so that you can assign the F-FB/F-FC to the protected F-FB/F-FC later (e.g., same name, comments regarding F-FB/F-FC).

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

If you want to save the backup copy in an F-library, make sure that the F-library is a user-created F-library in *S7 Distributed Safety* (see Section *User-Created F-Libraries*). The *FBD/LAD Editor* displays only F-libraries for *S7 Distributed Safety*.

5. Save the backup copy of the F-FB/F-FC.
   Result: The check box in the "Know-how protection" column of the "Safety Program" dialog is activated and can no longer be selected.
   The block icon in the "Block" column has a padlock. The F-FB or F-FC is protected.
6. Follow the same procedure until all the F-FBs/F-FCs you want to protect are protected.

Modifying Protected F-FBs/F-FCs

Note

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

If you want to modify a protected F-FB/F-FC, follow the steps below:

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

5.6.4 Defining F-Runtime Groups

Requirements

You must have programmed your safety program.

Rules for F-Runtime Groups of the Safety Program

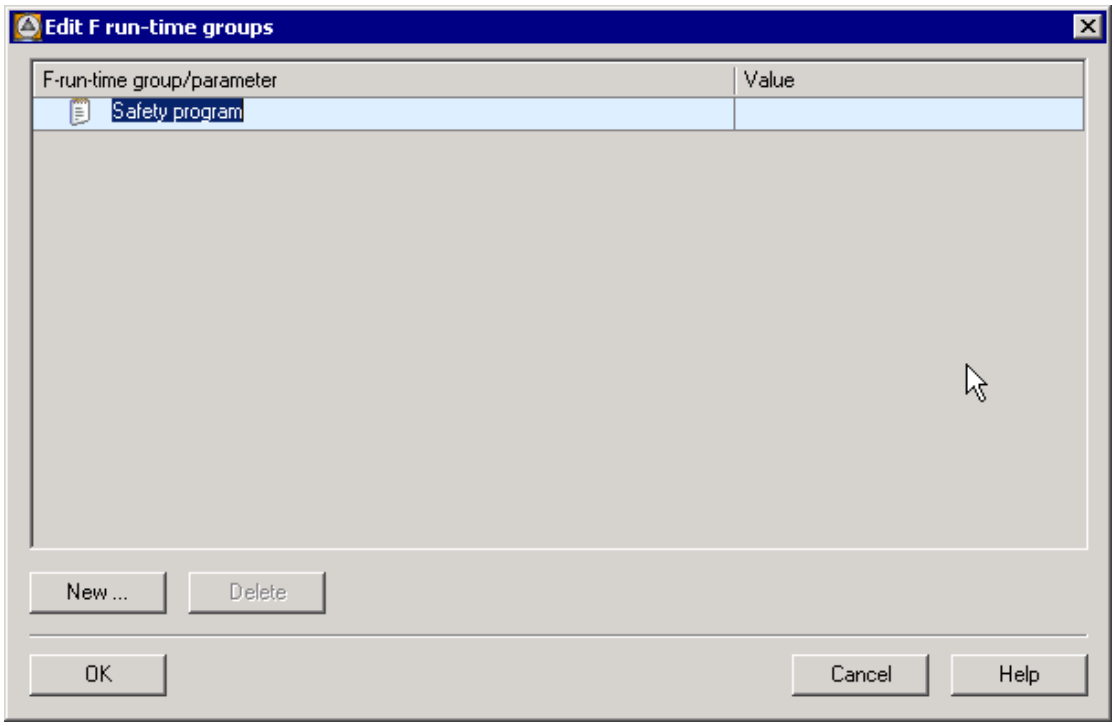
Warning

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

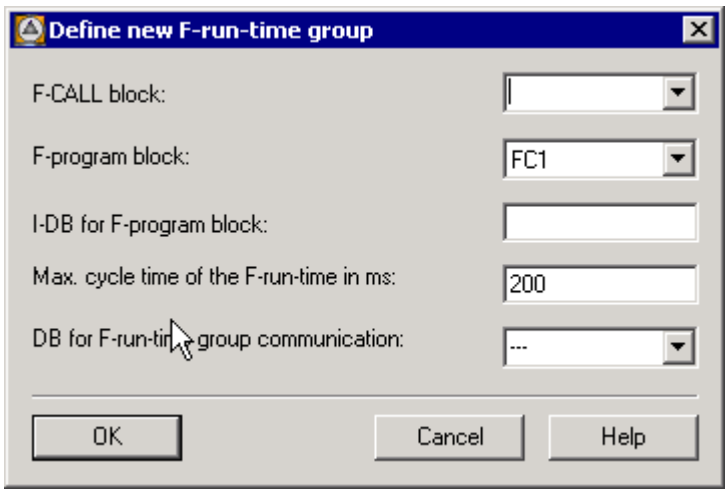
- F-blocks must not be called directly in an OB; rather, they must be inserted into one or two F-runtime groups.
- For optimal use of local data, you must call the F-CALL blocks (the F-runtime groups) directly in OBs (cyclic interrupt OBs, to the extent possible); you should not declare any additional local data in these cyclic interrupt OBs.
- Within a cyclic interrupt OB, the F-CALL (the F-runtime 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-runtime is always called in fixed time intervals, regardless of the length of execution of the standard user program.
- An F-CALL can only be called once. Multiple calls are illegal and can cause the F-CPU to go to STOP mode.
- The process image of inputs and outputs of standard I/O and memory bits can be accesses from several F-runtime groups.
- F-FCs can generally be called in more than one F-runtime group.

Procedure for Defining an F-Runtime Group

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



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/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 defined. If the assigned block 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 match 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 1,020,000 ms); see *Safety Engineering in SIMATIC S7* system description.

Warning

The F-runtime group call interval is monitored for maximum value, that is, to verify whether the call is executed often enough, but not whether it is executed too often. For this reason, fail-safe timers must be implemented by means of F-application blocks from the *Distributed Safety* F-library (V1) and not by means of 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."
7. 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 created F-blocks.
8. Repeat steps 2 to 7 to create a second F-runtime group.
9. After 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.

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

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 an F-runtime group in an F-DB are read in another F-runtime group.

The "DB for F-Runtime Group Communication" is created in one of the following ways:

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

Note

The F-runtime group for which you furnished the F-DB can read and write to a DB for F-runtime group communication, while the "receiver" F-runtime group can only read this F-DB.

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 communication in *SIMATIC Manager* in the same way as other F-DBs (see *Creating and Editing an F-DB*).

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

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

Updating Data Read from Another F-Runtime Group

Note

The data read are as old as the point in time when the F-runtime groups that made the data available completed processing before the start of the reading F-runtime group.

If the data being made available have been changed several times during the processing of the F-runtime group making the data available, the reading F-runtime group is always given the most recent changed data.

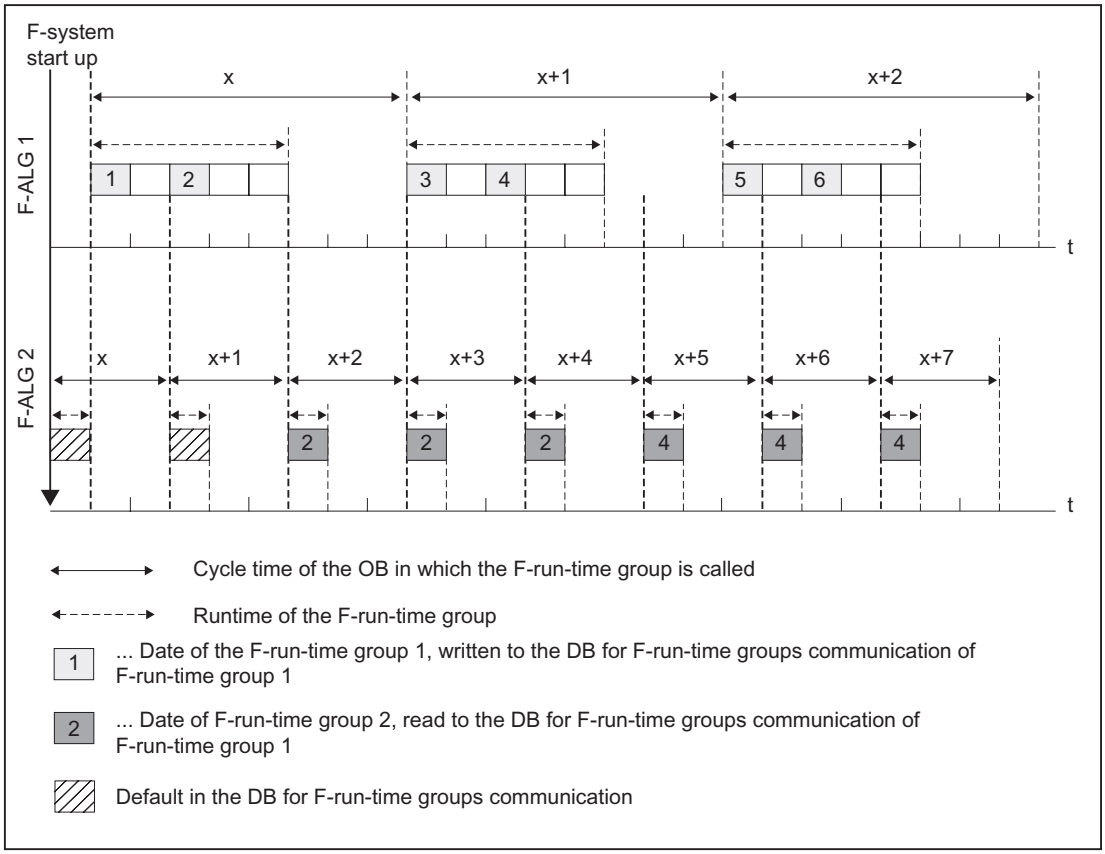
Assigning Fail-Safe Values

After the startup of the F-system, the F-runtime group reading data from the DB for F-runtime group communication of another F-runtime group (for example, F-runtime group 2), fail-safe values are made available. The values you specified in the F-runtime group communication DB of F-runtime group 1 are provided as fail-safe values (default of the F-runtime group communication DB).

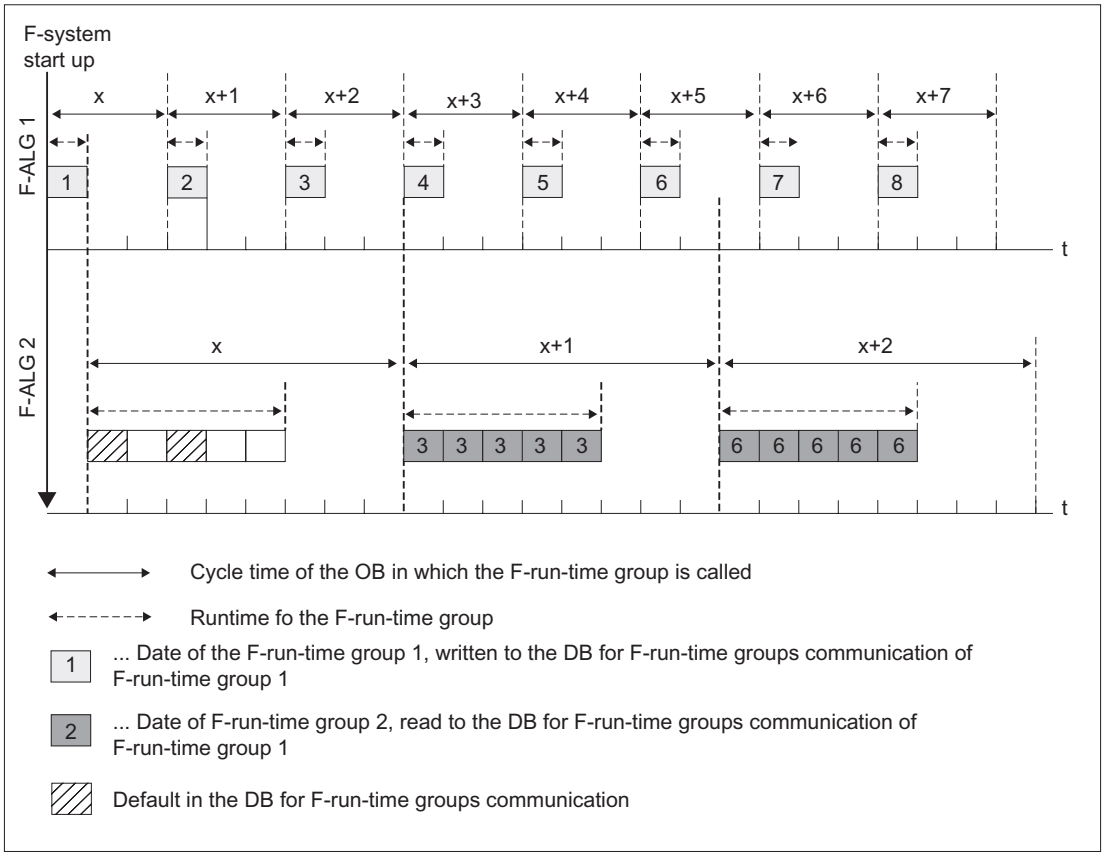
F-runtime group 2 reads the fail-safe values the first time it is initiated. The second time F-runtime group 2 is called, it reads the latest data if F-runtime group 1 has completed processing between the two calls for F-runtime group 2. If F-runtime group 1 has not completed processing, F-runtime group 2 continues to read the fail-safe values until F-runtime group 1 has completed processing.

The response 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



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



F-runtime group providing data is not processed

Note

When the F-runtime group from which the DB for F-runtime group communication data is being read is not processed (F-CALL of the F-runtime group is not called in an OB), *S7 Distributed Safety* outputs the following:

- The "Cycle time overrun" error message
- The number of the F-CALL for the F-runtime group not being processed and
- The "Current cycle time in ms: 0"

The F-CPU is set to STOP with SFC 46.

Deleting an F-Runtime Group

1. In the "Edit F-Runtime Groups" dialog, select the folder of the F-runtime group to be deleted.
2. Activate the "Delete" button.

The assignment of the F-blocks to an F-runtime 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*.

Changing F-Runtime Groups

You can make the following changes for each runtime group of your safety program in the "Edit F-Runtime 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-runtime group
- Define a different F-DB as the data block for F-runtime group communication (select an F-DB from the drop-down list or enter a new one)

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

5.7 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 F-attribute are initialized with the values from the load memory - as is the case with a cold start. 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 restart of the safety program with the values from the load memory. If the process does not allow this, a restart protection must be programmed as part of the safety program: Process data outputs must be blocked until manually released. These outputs must not be released until it is safe to do so and errors have been removed.

Example of Restart/Startup Protection Application

A requirement for applying restart/startup protection is that startup must be detected. To detect 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 (see *F-I/O DB*).

For manual release, reset this variable by user acknowledgment (see *Implementing User Acknowledgment in Safety Program of F-CPU of DP Master* and *Implementing User Acknowledgment in Safety Program of F-CPU of I-Slave*).

5.8 Compiling and Downloading the Safety Program

5.8.1 "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.

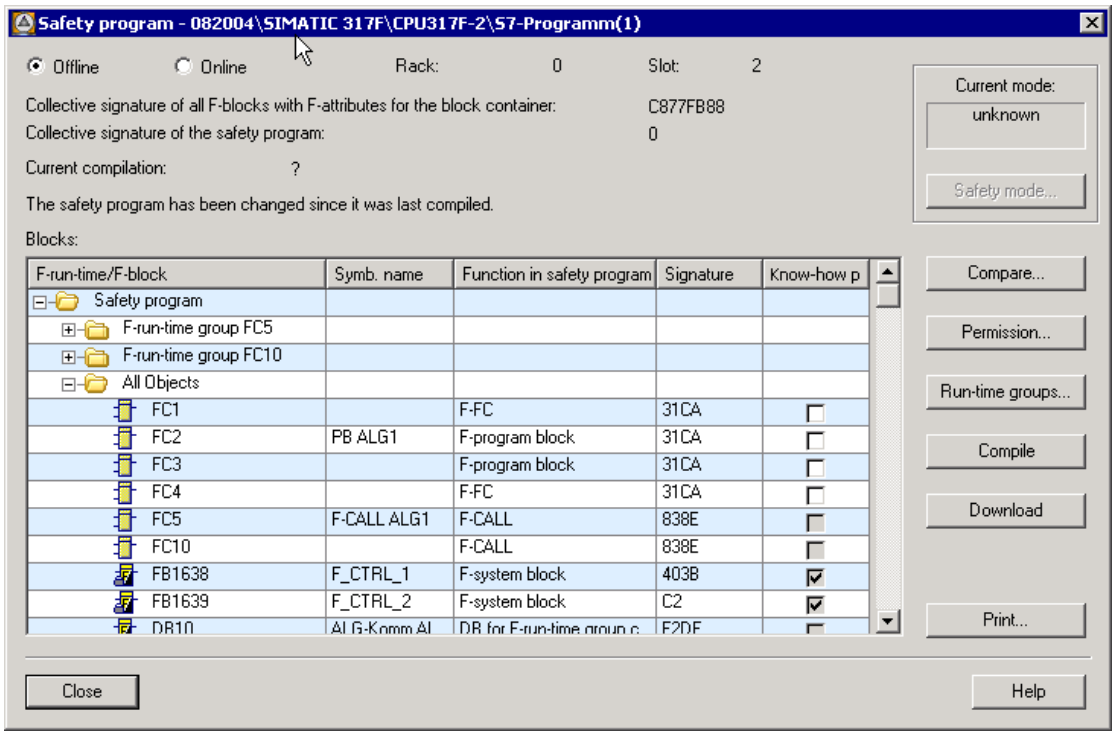
Note

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 and the S7 program assigned to it.
2. In *SIMATIC Manager,* select the **Options > Edit Safety Program** menu command.
   The "Safety Program" dialog will appear.



Contents of "Safety Program" Dialog

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" option buttons to choose whether the F-blocks of the offline or online block container are to be listed.

The "F-Run-Time 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 runtime group folders are formed as follows: "F-runtime group" + name of F-CALL of F-runtime group.

The "F-Run-Time Group ..." folder contains 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-Run-Time Group" folder also contains an "F-I/O DBs" folder. This folder contains all F-I/O DBs that are addressed from the F-run-time 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 safety program": value across all F-blocks called in the F-runtime group of the safety program (see *Safety Program States*)
- Information regarding the state of the safety program. There are three possible states (see *Safety Program States*):
  - Consistent
  - Inconsistent
  - Modified
- "Current mode:" contains information showing if:
  - 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
  - The "F-runtime group cannot be called": an F-Call for at least one F-runtime group was not called (because no call of the F-CALL has been programmed in an OB (OB 35), for example).

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 this 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.

To print the safety program, see *Printing Project Data of Safety Program*.

Aborted Connection to F-CPU

If the connection to the F-CPU is aborted while the "Safety Program" dialog or one of its follow-on dialogs is open, close the "Safety Program" dialog and all of its follow-on dialogs.

5.8.2 Safety Program 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 (see also *System 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.
While unused F-blocks with F-attribute produce an executable safety program, this program cannot pass acceptance tests due to inconsistent collective signatures. Therefore, before the safety program is downloaded to the F-CPU, a warning is displayed ("The ... safety program is inconsistent").

Modified

Before the safety program is downloaded to the F-CPU, a warning is displayed ("The safety program has been changed"). You then have the option of generating a consistent safety program (i.e., to compile the safety program).

5.8.3 Compiling the Safety Program

Note

Before you compile the safety program, close the *LAD/FBD Editor*, *Display S7 Reference Data*, and *Check Block Consistency* applications and the symbol table.

Procedure for Compiling the Safety Program

1. Select the correct F-CPU and the 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.

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 are not using 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 F-system blocks required additionally 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.

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, 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.

Compiling a Consistent Safety Program that Was Uploaded from the F-CPU to the Programming Device/PC

Note the following when compiling a consistent safety program that has been uploaded from the F-CPU to the programming device or PC:

- Prior to compiling, you must update the symbolic name for the F-shared DB by hand; symbolic name = name in the object properties for the block (header). Otherwise, the compile operation will be aborted with an error message.
- When compiling, you are prompted to replace the F-application blocks of the Distributed Safety F-library (V1) used in the safety program with the F-application blocks in the currently installed F-library. If you confirm the prompt, the associated instance DBs will be regenerated with default values. The actual values in the instance DBs at the time that the safety program was uploaded to the programming device or PC are overwritten with the default values.

5.8.4 Downloading the Safety Program

Introduction

Once you have compiled your safety program, you can download it to the F-CPU. You have two options:

- Download the entire safety program in the"Safety Program" dialog in STOP mode
  This is the recommended method for downloading a consistent safety program.
- Downloading Individual F-Blocks in *SIMATIC Manager* or *FBD/LAD Editor*

Procedure for Downloading in the "Safety Program" Dialog

1. Select the correct F-CPU and the 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.

Any F-blocks in the online block container are deleted before the F-blocks are downloaded.

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. In the F-CPU, all F-blocks without F-attribute are deleted. 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" option buttons 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).

Downloading to an S7-PLCSIM

Starting with S7 Distributed Safety V 5.3, you can test the safety program with the SP-PLCSIM function (hardware simulation) of STEP 7 (see *Testing the Safety Program*).

Requirements for Downloading to an S7-PLCSIM

- The option package S7-PLCSIM V 5.3 or later 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 and the 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 in the "Safety Program" dialog.
   All F-blocks with F-attribute belonging to the safety program are identified and downloaded to the 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 an S7-PLCSIM. If the target device is an S7-PLCSIM, special simulation blocks (F-system blocks) are automatically downloaded from the S7 Distributed Safety F-library (V1) to the S7-PLCSIM.

Your offline safety program is unchanged and consistent following the download operation to the S7-PLCSIM. In the S7-PLCSIM, the collective signature of all F-blocks with F-attribute no longer matches the collective signature.

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 an 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.

Downloading in *SIMATIC Manager* or *FBD/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. The F-CPU goes to STOP when an inconsistent safety program is encountered in active safety mode.

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.
- 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 made effective 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.

5.8.5 RAM Requirement for Safety Program

You can estimate the RAM requirement for the safety program as follows:

RAM Requirement for Safety Program:

26 Kbytes for F-system blocks F_CTRL_1, F_CTRL_2, and F_IO_BOI

+ 4.3 Kbytes for F-system block F_RTGCO2 (for F-runtime group communication only)
+ 4.5 x RAM requirement for all F-FB/F-FC/F-PB
+ 4.5 x RAM requirement for all utilized F-application blocks (except F_SENDDP, F_RCVDP, F_SENDS7 and F_RCVS7)
+ RAM requirement for utilized F-application blocks F_SENDDP and F_RCVDP (4.4 Kbytes each)
+ RAM requirement for utilized F-application blocks F_SENDS7 and F_RCVS7 (9.5 Kbytes each)

RAM Requirement for Data:

5 x RAM requirement for all F-DBs (including F-communication DB, but excluding DB for F-runtime group communication) and I-DBs for F-PB/F-FB

+ 24 x RAM requirement for all DBs for F-runtime group communication
+ 2.3 x RAM requirement for all I-DBs for F-application blocks (except F_SENDDP, F_RCVDP, F_SENDS7 and F_RCVS7)
+ RAM requirement for all I-DBs of 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)
+ 4 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).

5.8.6 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 a flash card inserted (e.g., CPU 416F-2)
- F-CPUs with MMC
  (e.g., CPU 317F-2 DP, CPU 315F-2 DP, or IM 151-7 F-CPU)

Warning

If the function of the safety program is not tested 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:

1. For F-CPUs with MMC: Download the safety program to the F-CPU in the "Safety Program" dialog.
   For F-CPUs with flash card inserted: Download the safety program to the F-CPU in the "Download User Program to Memory Card" dialog.
2. 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; refer to the section, *Comparing Safety Programs*).
3. Perform a memory reset of the F-CPU using the **mode selector** or via the **programming device or PC**. This causes the safety program to be transferred from the load memory (memory card, MMC for 3xxF F-CPUs and IM 151-7 F-CPU or flash card for 4xxF F-CPUs) to the RAM again after the RAM is cleared.

Warning

If **more than one F-CPU** is accessible over a network (such as MPI) from **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:

1. Perform a memory reset of the F-CPU using the **mode selector** or via the **programming device or PC**.
2. Download the configuration to the F-CPU in *HW Config*.
3. Download the safety program to the F-CPU in the "Safety Program" dialog.
4. 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; refer to the section, *Comparing Safety Programs*).

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 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 of flash card) to ensure that the F-CPU does not contain an "old" safety program:

1. 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.)
2. Remove the Micro Memory Card (MMC or flash card) with the old safety program from the F-CPU.
3. Insert the Micro Memory Card (MMC or flash card) with the new safety program in the F-CPU.
4. Switch on the F-CPU again. For F-CPUs with battery backup (e.g.,CPU 416F-2), reinsert the battery, if one was present.

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:

1. Download the safety program to the memory card (MMC or flash card).
2. 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, see section, *Comparing Safety Programs*).
3. Affix an appropriate label to the memory card (MMC or flash card).

The procedure outlined must be ensured through organizational measures.

5.9 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

As of S7 Distributed Safety V 5.3, you can test the safety program with the S7-PLCSIM function of STEP 7 as of V 5.3 (hardware simulation). 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.

5.9.1 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.

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 only 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 displayed.
  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/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. For safety-related CPU-CPU communication, you must take into account the following: If the F-CPU with F_SENDDP or F_SENDS7 is in deactivated safety mode, you cannot assume that the data sent by 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 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.

Requirement for Deactivating Safety Mode

The F-CPU is in RUN mode and safety mode is activated.

Procedure for Deactivating Safety Mode

1. Select the correct F-CPU and the 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 this 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 safety program.
6. If the password is not valid, safety mode is not deactivated and remains active.

   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.

   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.
7. 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 (see *Modifying the Safety Program in RUN 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 monitored operation and manual safety shutdown.

5.9.2 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 Safety Program Data for Testing and Commissioning

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 *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, in your safety program, you must always use at least one channel of the F-I/O that you would like to modify.

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 (see *Program Structure of Safety Program in S7 Distributed Safety*).

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 (see also *Process Data or Fail-Safe Values*). 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 the F-block offline and then download it (see *Modifying the Safety Program in RUN Mode*).

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 modified offline, you must do so by editing the actual value and compiling the safety program offline, as well (see *Modifying the Safety Program in RUN Mode*).

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
  - Errors in communicating with the F-I/O
  - Errors in 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:
  - Errors in communicating with the F-I/O
  - Errors in 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.

Procedure for Testing the Safety Program

The following procedure is used for testing:

1. Deactivate safety mode (see *Deactivating 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 *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 (see *Downloading the Safety Program*).

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. Safety mode can be deactivated only in the "Safety Program" dialog. (see *Deactivating Safety Mode*).

For a detailed description of the S7-PLCSIM function of *STEP 7*, refer to the *S7-PLCSIM V 5.x* user manual.

5.10 Modifying the Safety Program

5.10.1 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 *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 *Creating F-Blocks in F-FBD/F-LAD*.

Procedure for Modifying the Safety Program in RUN Mode

1. Modify and save the F-PB/F-FB and its associated instance-DB, F-FC, or F-DB in *FBD/LAD Editor*.
2. Download the modified F-block from *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 after inserting a new block call for F_SENDDP, F_SENDS7, F_RCVDP or F_RCVS7 you must always recompile the relevant safety program and download it in its entirety to the F-CPU in STOP mode (see *Compiling the Safety Program* and *Downloading the Safety Program*).

Restrictions on F-Runtime Group Communication

You cannot modify F-runtime group communication in RUN mode of the F-CPU.

After modifying F-runtime group communication in the STOP mode, you must recompile the safety program and perform a complete download to the F-CPU (see sections, *Compiling the Safety Program* and *Downloading the Safety Program*).

Restrictions on F-I/O Access

If during operation (in RUN mode), you insert an F-I/O access to an F-I/O in which no single channel or variable from the associated F-I/O DB was 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 (see *Compiling the Safety Program* and *Downloading the Safety Program*).

Modifications to the Standard User Program

You can download modifications to the standard user program in RUN mode of the F-CPU, 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. Consequently, an appropriate level of protection must be set (see *Configuring the F-CPU*).

Modifications to an F-Run-Time Group Call

If an OB (e.g., OB35) 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 (see *Compiling the Safety Program*).
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 (see *Downloading the Safety Program*).
3. Follow the steps described in *Safety Program Acceptance Test*.

5.10.2 Comparing Safety Programs

Criteria for Comparing Safety Programs

You can compare two versions of the safety program 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.

Comparison Options

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)

Procedure for Comparing Safety Programs

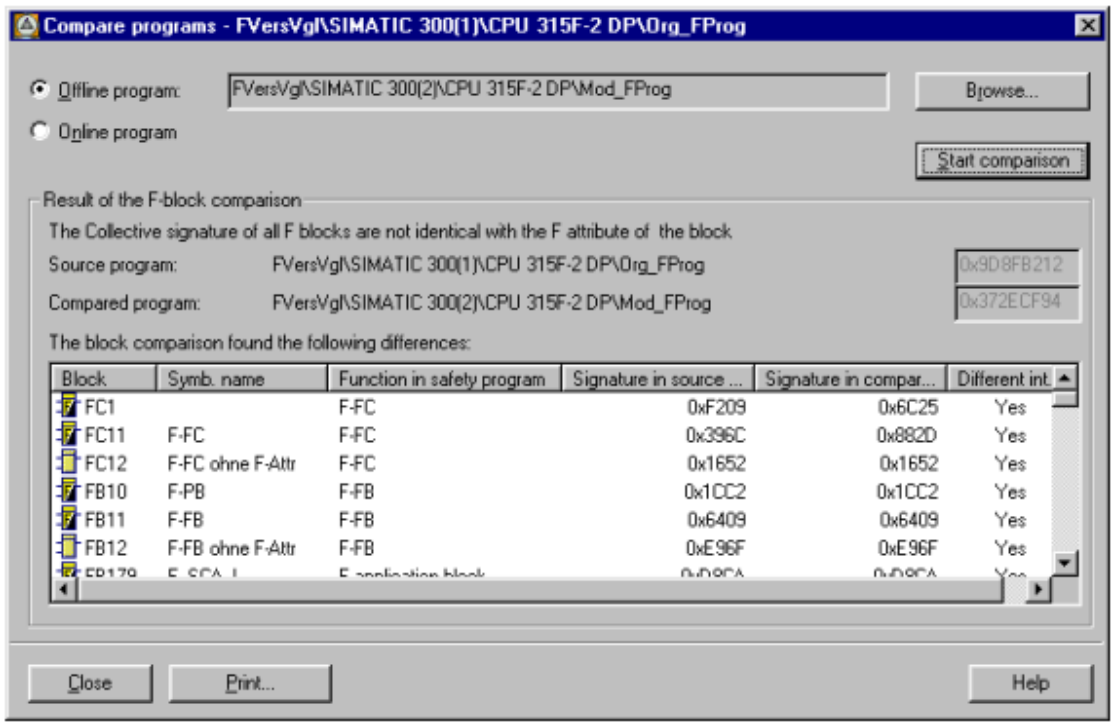
To compare two safety programs:

1. Select the correct F-CPU and the S7 program assigned to it.

   If you want to compare a safety program on a Memory Card (MMC or flash card), you must use the safety program on the Memory Card (MMC or flash card) as the source for the comparison (safety program 1). In *SIMATIC Manager,* select the **File > S7 Memory Card > Open** menu command.

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 program" dialog will appear.



4. Select the safety program you would like to compare (safety program 2). 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 comparison 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 in declaration table in called FBs/FCs or in F-PB/F-FB/ F-FC of utilized F-DBs • 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 to 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, there are differences in 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*.

5.10.3 Deleting the Safety Program

Use the following procedure to delete the safety program or individual components of the safety program.

Deleting Individual F-Blocks

To delete an F-block, follow the same procedure as for a standard program.

Deleting an F-Run-Time Group

1. In the "Edit F-Run-Time Groups" dialog, select the folder of the F-runtime group to be deleted.
2. Activate the "Delete" button.

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 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.

5.11 Printing the Project Data of the Safety Program

Note

Before you print the project data of the safety program, close the *HW Config* and *LAD/FBD Editor* applications and the symbol table.

Procedure for Printing All Important Project Data of the Safety Program

1. Select the correct F-CPU and the 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.
   You can now select the parts of the project to be printed:
   - "Function Block Diagram/Ladder Diagram": All F-blocks you created for the safety program (F-PB, F-FB, F-FC, F-DB) corresponding to 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 acceptance test (see *System Acceptance Test*)
   - "Hardware Configuration" with module parameters
   - "Symbol Table"

Printed Project Data for the Safety Program

The printout of the safety program ("Safety program" option button) also contains the collective signatures and the date of the last generation procedure, which are relevant to the onsite acceptance test of the safety program (e.g., by experts).

Two collective signatures are output in the printout:

- "F-blocks with F-attribute in the block container" in the footer and in the program information section (= "collective signature of all F-blocks with F-attribute in the block container" in the *"Safety Program" dialog*)
- *"Safety Program"* in the program information section (= "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 (see *Safety Program States*). This is also indicated in the footer.

In addition, the following information is printed out:

- The F-compiler version used to compile the safety program is printed out as an internal version identifier of *S7 Distributed Safety.*
- Time when safety program was compiled.
- A note indicating whether the amount of local data reserved for the safety program has been exceeded.
- List of all F-blocks contained in the block container. F-blocks without F-attribute are identified by square brackets around the block name and signature.

  The following information is indicated 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 F-application block
  - Numbers of instance DBs used to call the F-application block
  - Name of F-block in which the F-application is called
  - Network number of call
  - Name of F-runtime group (Name of F-CALL)
  - Parameter value
- List of data from the standard user program:
  - Address
  - Symbol
  - F-runtime group in which the data element is used
- List of data for data communication between the F-runtime groups
  - Number of the F-CALL of the "sender" F-runtime group
  - Number of the F-CALL for 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 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, F-shared DB and automatically generated F-blocks. F-blocks without F-attribute are identified by square brackets around the block name and signature.

  The following information is indicated 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
- The following information is indicated for 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
  - Initial address
  - Name/identifier of the F-I/O
  - Module type
  - F_Monitoring_Time
  - Cyclic redundancy check by means of parameter assignment (to enable rapid detection of changes in I/O)
  - PROFIsafe source and destination address

- 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 of the generation time
  - Absolute and symbolic address of the RLO 0
  - Absolute and symbolic address of the RLO 1
- The following information is displayed in the footer:
  - Collective signature of all F-blocks with F-attribute in the block container
  - Signature of symbols (only for printout of the offline safety program)
  - Internal version identifier of *S7 Distributed Safety*
  - Version of F-compiler used to create the printout
  - Depending on the safety program state: "Safety program changed", "Safety program not changed" or "Symbols not current"

Note

If "Symbols not current" 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 online Help for *STEP 7*). If necessary, you must recompile the safety program.

6 F-Libraries

6.1 Distributed Safety F-Library (V1)

6.1.1 Overview 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 Block\Blocks* block container

---

Note

You must not change the F-library name.

The *Distributed Safety* F-library (V1) may only contain F-blocks that were installed with the *S7 Distributed Safety* version.

6.1.2 F-Application Blocks

Overview of F-Application Blocks

Block Number	Block Name	Function
FB 179	F_SCA_I	Scale values of data type INT
FB 181	F_CTU	Count up
FB 182	F_CTD	Count down
FB 183	F_CTUD	Count up and down
FB 184	F_TP	Create pulse
FB 185	F_TON	Create ON-delay
FB 186	F_TOF	Create OFF-delay
FB 187	F_ACK_OP	Fail-safe acknowledgment
FB 188	F_2HAND	Two-hand monitoring
FB 189	F_MUTING	MUTING
FB 190	F_1oo2DI	1oo2 evaluation with discrepancy analysis
FB 211	F_2H_EN	Two-hand monitoring with enable
FB 212	F_MUT_P	Parallel muting
FB 215	F_ESTOP1	Emergency STOP up to Stop Category 1
FB 216	F_FDBBACK	Feedback monitoring
FB 217	F_SFDOOR	Safety door monitoring
FB 223	F_SENDDP	Send data (16 BOOL, 2 INT) via PROFIBUS DP
FB 224	F_RCVDP	Receive data (16 BOOL, 2 INT) via PROFIBUS DP
FB 225	F_SENDS7	For CPUs 4xxF: Send data (from F-DB) via S7 connections
FB 226	F_RCVS7	For CPUs 4xxF: Receive data (from F-DB) via S7 connections
FC 174	F_SHL_W	Shift left 16 bits
FC 175	F_SHR_W	Shift right 16 bits
FC 176	F_BO_W	Convert 16 data elements of data type BOOL to a data element of data type WORD
FC 177	F_W_BO	Convert a data element of data type WORD to 16 data elements of data type BOOL
FC 178	F_INT_WR	Write value of data type INT indirectly to an F-DB
FC 179	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.

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

When you call a block, enable input EN and 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

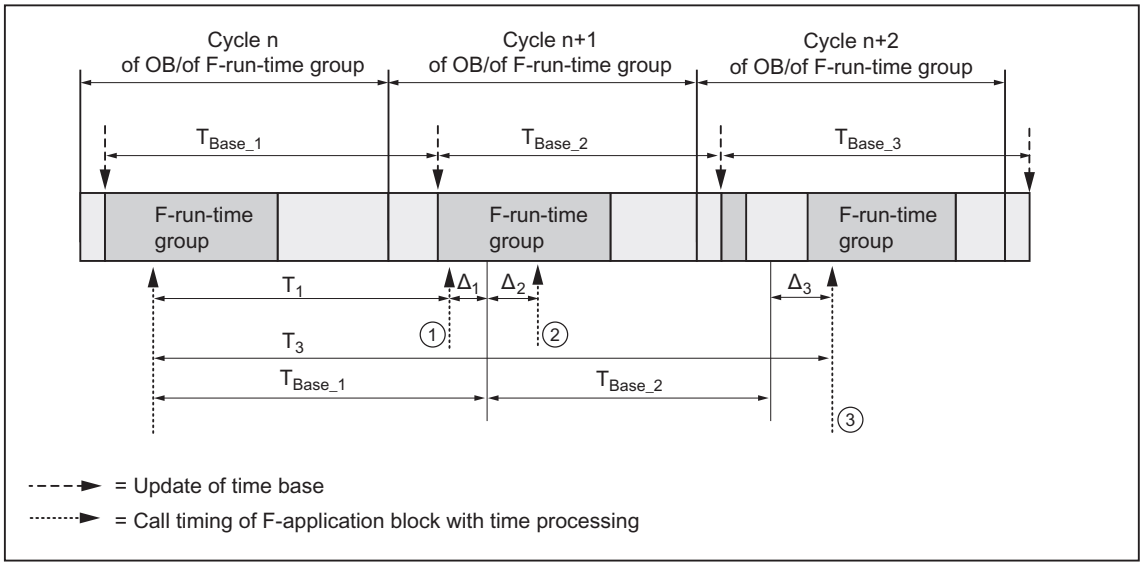
Warning

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 timing of the time base used in the F-application block (see *Timing Imprecision …* figure)
- 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 Timing of the Time Base Used in the F-Application Block



Explanation of Previous Figure:

(1) For the first call in cycle n+1, the call timing of the F-application block relative to the beginning 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 timing of the F-application block relative to the beginning 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 since the call in cycle n. This would also be the case if no call occurred in cycle n+1.

6.1.2.1 FB 179 "F_SCA_I": Scale Values of Data Type INT

Connections

	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 > 27,648: 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 7FFFH (for overflow) or 8000H (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 output OUT_HI or OUT_LO.

6.1.2.2 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 32,767)	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 (as compared to the previous 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 or not 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

6.1.2.3 FB 182 "F_CTD": Count Down

Connections

	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 on a rising edge (as compared to the previous 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

6.1.2.4 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: Q = 1, if CV >= PV Q = 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").

On a rising edge (as compared to the previous 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 in a cycle at both input CU and input CD, 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 or not 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 F_CTUD are reset in the first cycle following startup of the F-system, resulting in:

- CV = 0
- QU = 0
- QD = 0

6.1.2.5 FB 184 "F_TP": Create a 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 timing of the time base used in the F-application block (see *Timing Imprecision …* figure)
- 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

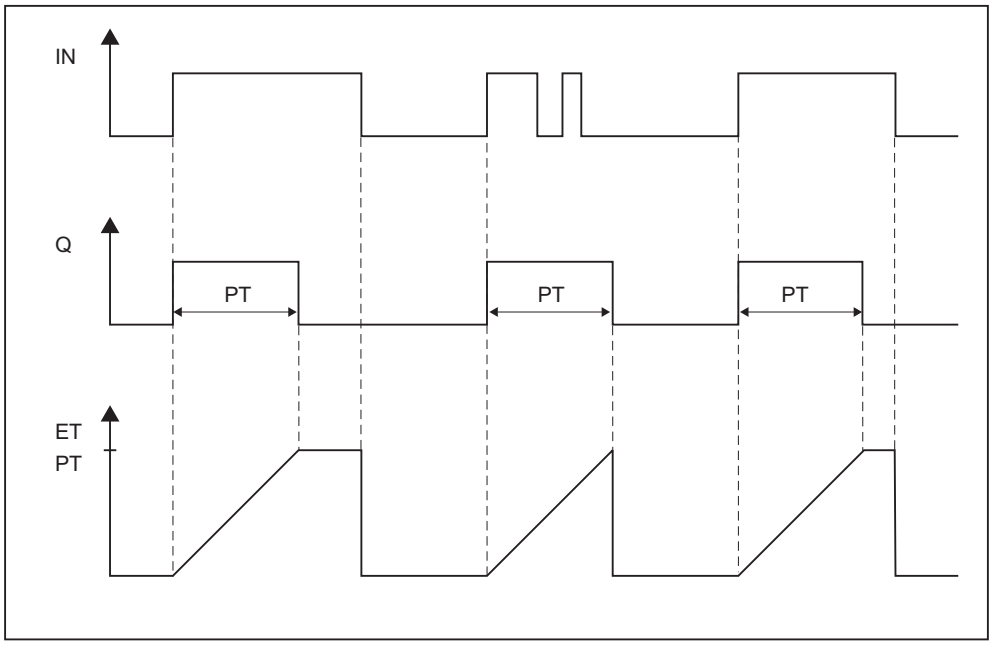
You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

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 startup of the F-system, resulting in:

- ET = 0
- Q = 0

6.1.2.6 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 *Timing Imprecision …* figure)
- 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

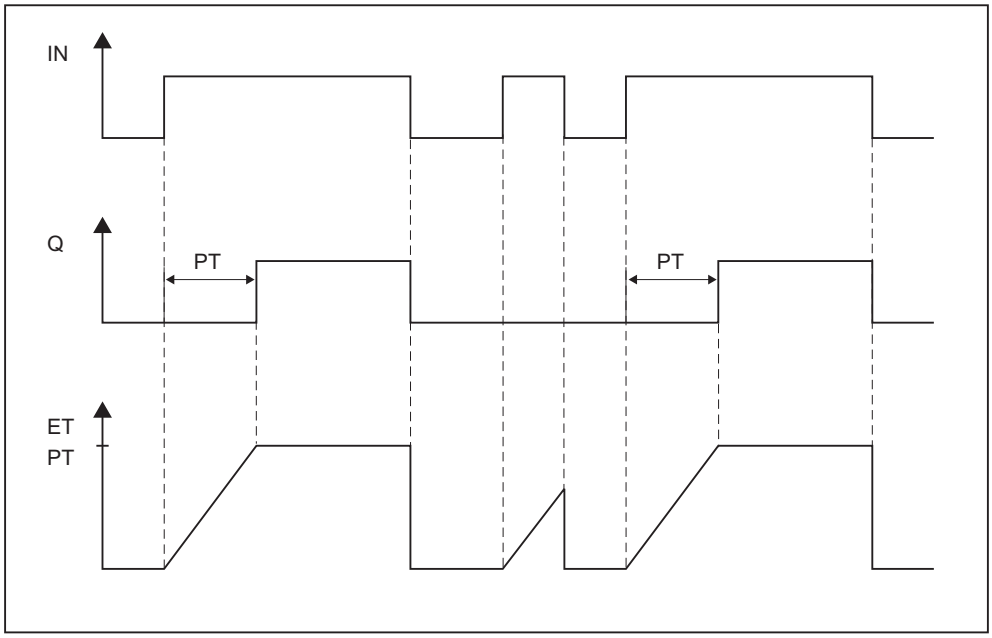
You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

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). This 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 startup of the F-system, resulting in:

- ET = 0
- Q = 0

6.1.2.7 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 *Timing Imprecision …* figure)
- 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

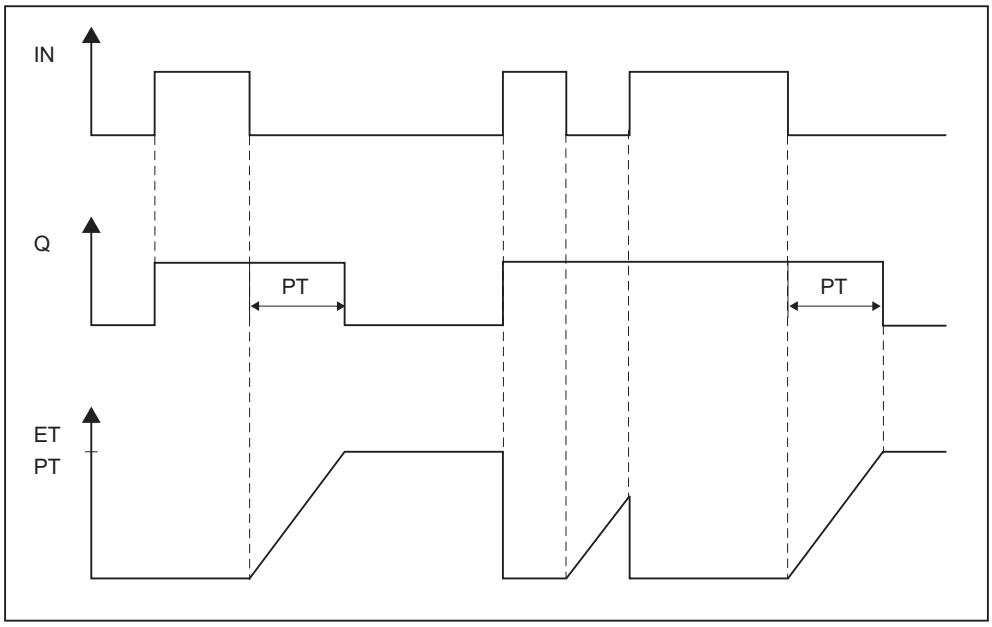
You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

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 startup of the F-system, resulting in:

- ET = 0
- Q = 0

6.1.2.8 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 it has changed to a value of 9 after 1 s, at the earliest, or 1 minute, at the latest. Output OUT (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 one minute or the change occurred before one second has elapsed, then in/out parameter IN is reset to 0, and both steps listed above must be repeated.

During the time in which in/out parameter IN must change from 6 to 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 *Timing Imprecision …* figure)
- 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. Output Q cannot be accessed in the safety program.

You may 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!

Additional Information

For more information about fail-safe acknowledgment with the F-fail-safe application block F_ACK_OP, refer to the sections, Implementing User Acknowledgment in Safety Program of F-CPU of a DP Master and Implementing User Acknowledgment in Safety Program of F-CPU of an I-Slave.

6.1.2.9 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 $\leq$ 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), the 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 "Provide a value of 0" for the discrepancy behavior 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 is set to 1 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 *Timing Imprecision …* figure)
- 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 configuration in *Configuration*.

You will find additional information about the F-I/O DB in *F-I/O DB*.

6.1.2.10 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 signal of F-I/O DB of muting lamp	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.

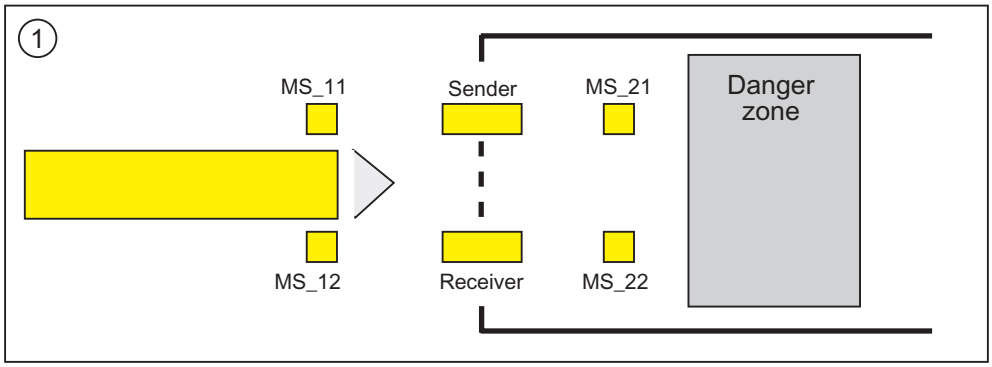
Warning

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 *Timing Imprecision …* figure)
- 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)

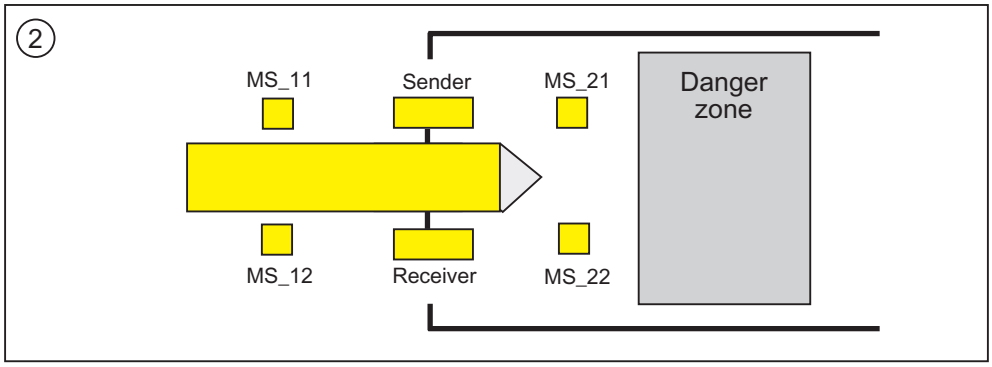


- If both muting sensors MS_11 and MS_12 are activated by the product within DISCTIM1 (apply signal state = 1), the F-application block starts the MUTING function. Enable signal Q remains 1, even when input FREE = 0 (light curtain interrupted by product). The MUTING output for setting the muting lamp switches to 1.

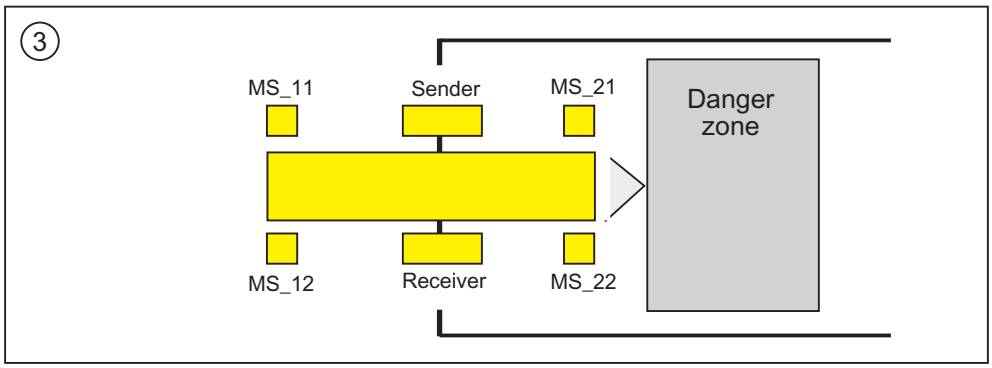
Warning

The muting lamp can be monitored using the QBAD_MUT input. To do this, you must wire the muting lamp to an output with wire break monitoring of an F-I/O and supply the QBAD_MUT input with the QBAD signal of the associated F-I/O DB. 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.

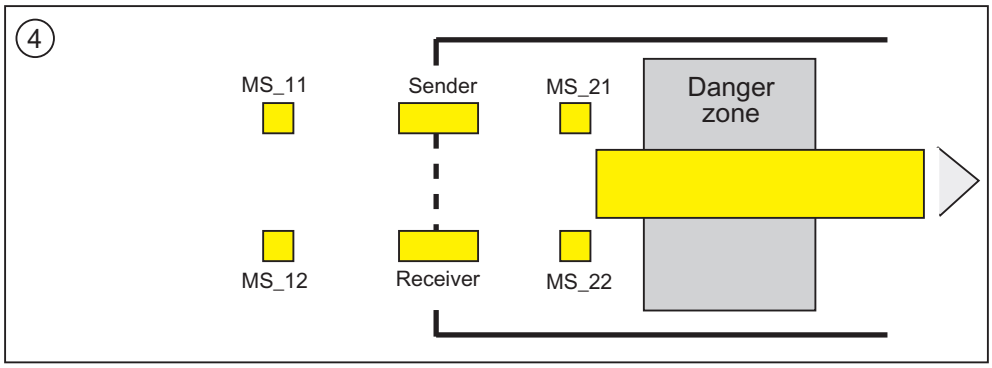
F-I/O that can promptly detect a wire break after activation of the muting operation must be used (see manual *for specific* F-I/O).



- 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).



- 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).

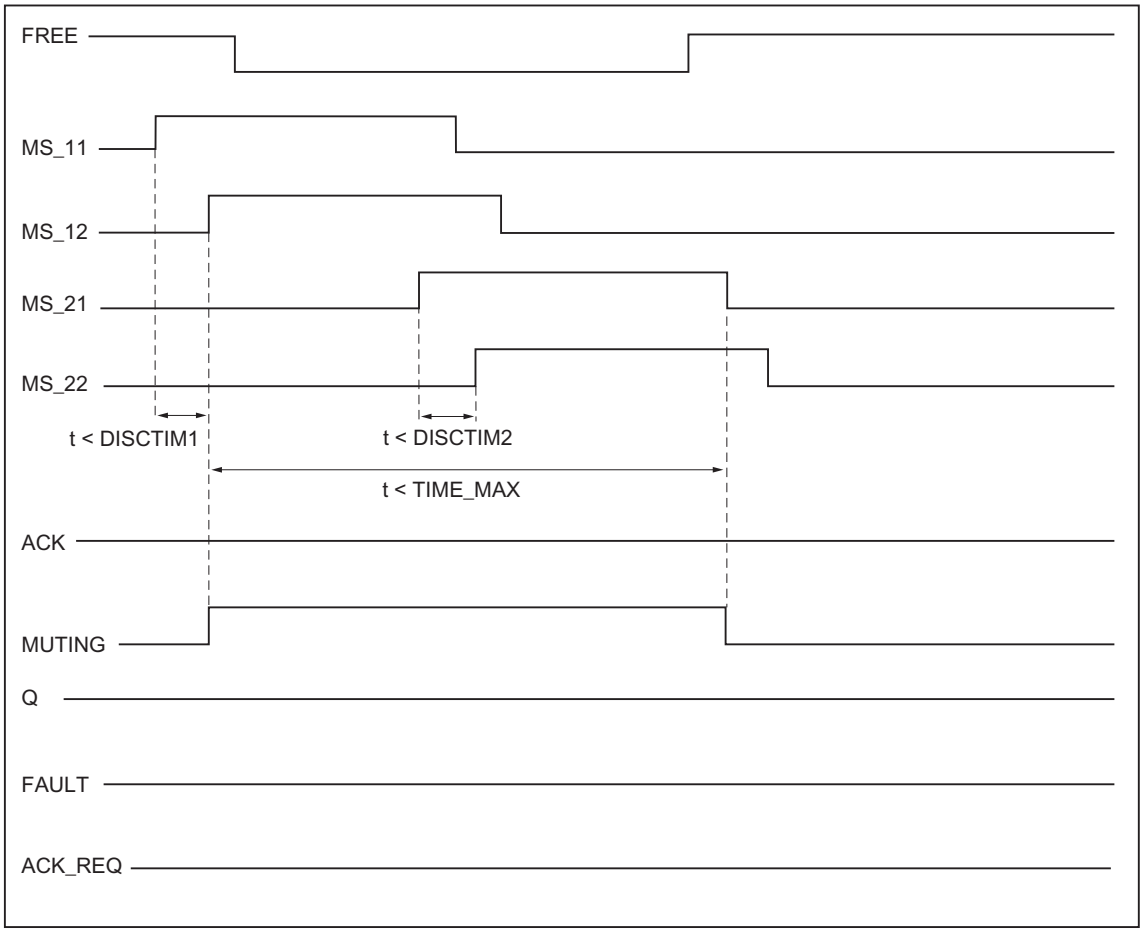


- Only if one of the two muting sensors MS_21 and MS_22 is switched to inactive (product enables sensors) is the MUTING function terminated (Q = 1, MUTING = 0). The maximum activation time for the MUTING function is the time set at input TIME_MAX.

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

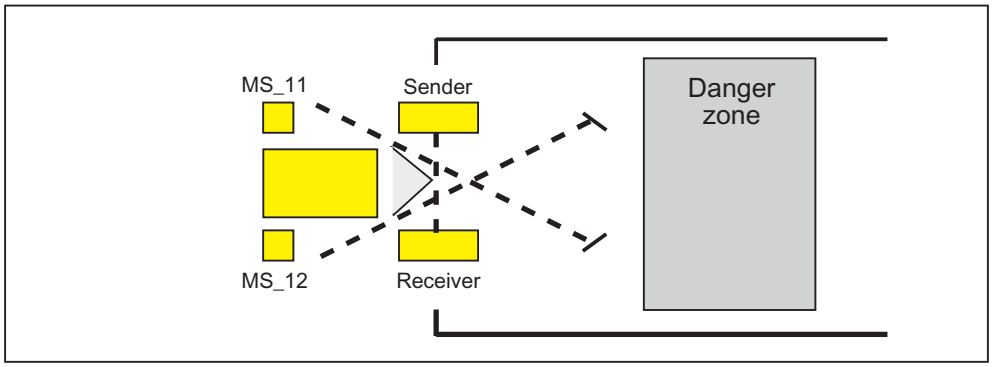


Schematic Sequence of Muting Procedure with Reflection Light Barriers

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.



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 within 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 less than 0 or greater than 3 s.
- Maximum muting time TIME_MAX has been set to a value less than 0 or greater than 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 are 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:

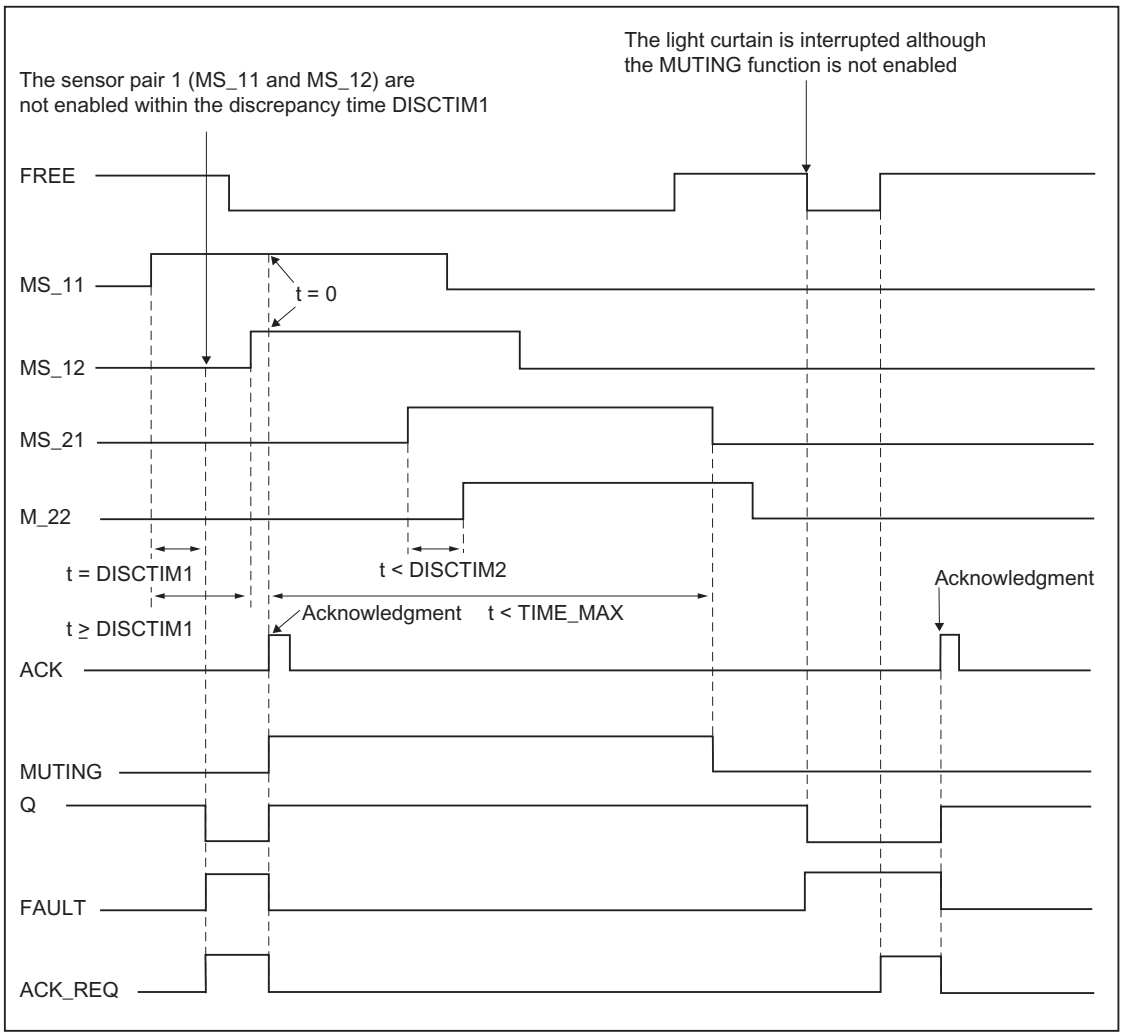
- The light curtain is no longer interrupted
- Errors, if present, are eliminated (see output DIAG)
  and
- A user acknowledgment with rising edge at input ACK occurs (see also *Implementing User Acknowledgment in Safety Program of F-CPU of DP Master* and *Implementing User Acknowledgment in Safety Program of F-CPU of I-Slave*).

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)



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 disabled. 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.	Meaning	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 solution, see DIAG variable, Bits 0 to 6 in *F-I/O DB*
		Discrepancy time setting is too low	If necessary, set a higher discrepancy time
		Discrepancy time setting < 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 no.	Meaning	Possible causes of problems	Remedies
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 < 0 s or > 10 min.	Set muting time in range between 0 s and 10 min.
Bit 3	Light curtain interrupted and muting not active	Light curtain is defective	Check light curtain
		Wiring fault	Check wiring of light curtain (FREE input)
		F-I/O fault, channel fault, or communication error, or passivation by means of PASS_On of F-I/O of light curtain (FREE input)	For solution, see DIAG variable, Bits 0 to 6 in *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 solution, see DIAG variable, Bits 0 to 6 in *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!

6.1.2.11 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 = Acknowledgment necessary	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 *Timing Imprecision …* figure)
- 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

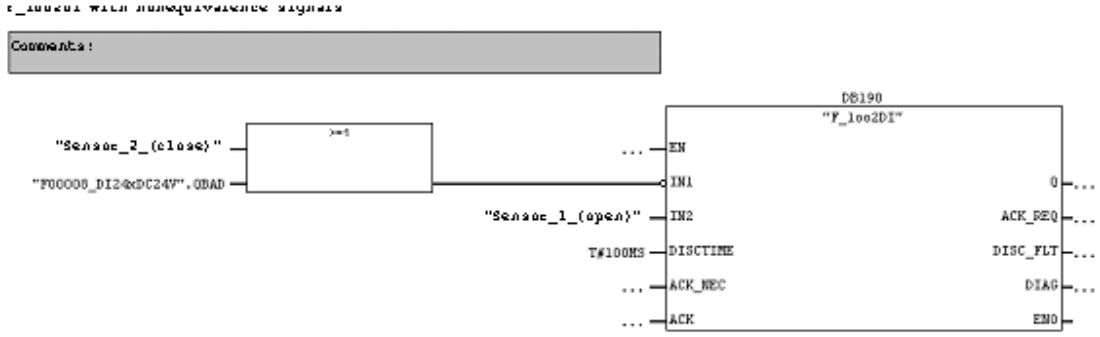
You must choose the interval between two call times of an F-application block with time processing so that the required response times are achieved, taking into account the possible timing imprecision.

Activating Inputs IN1 and IN2

Inputs IN1 and IN2 must both be activated in such a way that their safe state is 0.

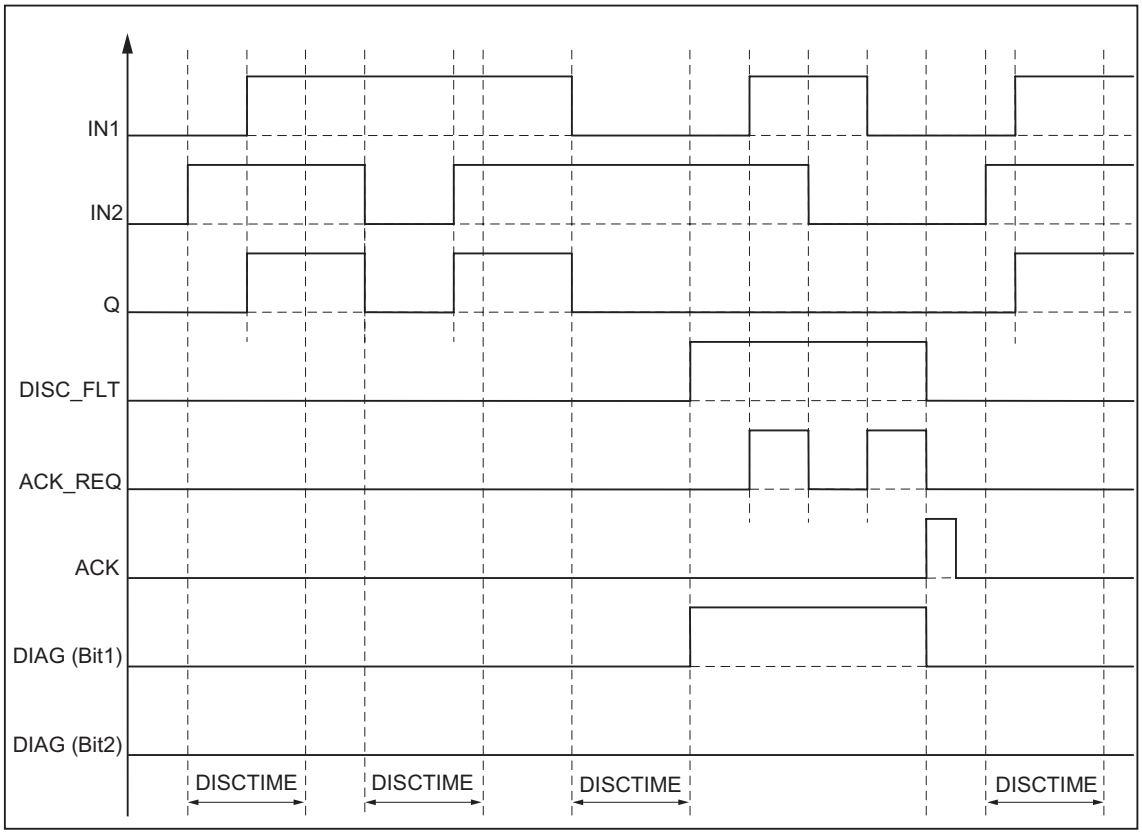
Example

For nonequivalent signals, you have to invert the input (IN1 or IN2), to which you have the assigned sensor signal with a safe sate of 1. You must also OR the sensor signal with the QBAD variable of the associated F-I/O DB, 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.	Meaning	Possible causes of problems	Remedies
Bit 0	Discrepancy error or incorrect discrepancy time setting (= state 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/P	For solution, see DIAG variable, Bits 0 to 6 in *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 between 0 s and 60 s.
Bit 1	If discrepancy error: Last signal state change was at input IN1	-	-
Bit 2	If discrepancy error: Last signal state change was at input IN2	-	-
Bit 3	Reserved	-	-
Bit 4	Reserved	-	-
Bit 5	If discrepancy error: 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!

6.1.2.12 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 $\leq$ 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 at the F-application block. With suitable configuration (type of sensor interconnection: 2-channel, nonequivalent), the 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 "Provide a value of 0" for the discrepancy behavior 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 is set to 1 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 *Timing Imprecision …* figure)
- 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.	Meaning	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/P	For solution, see DIAG variable, Bits 0 to 6 in *F-I/O DB*
Bit 2	Reserved	-	-
Bit 3	Reserved	-	-
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!

6.1.2.13 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 signal of F-I/O DB of muting lamp	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
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.

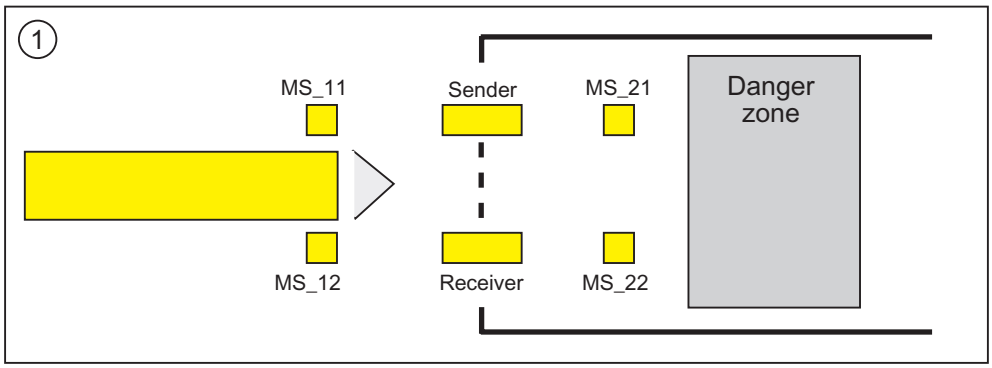
Warning

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 *Timing Imprecision …* figure)
- 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)

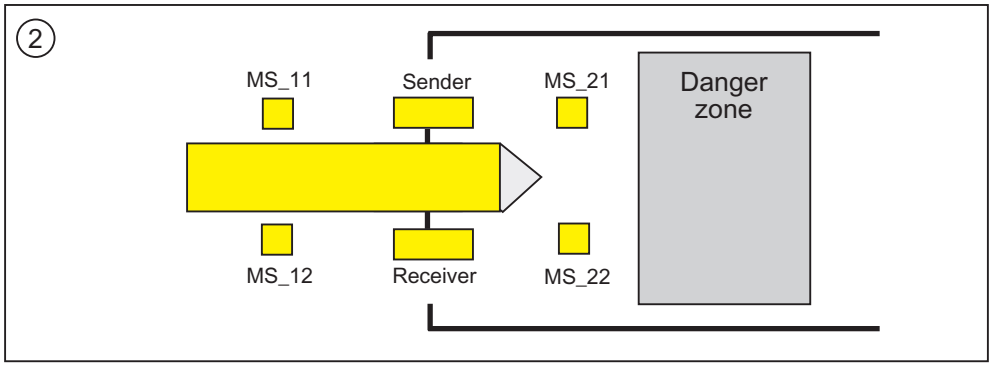


- If muting sensors MS_11 and MS_12 are both activated by the product within DISCTIM1 (apply signal state = 1) and MUTING is enabled by setting the ENABLE input to 1, the F-application block starts the MUTING function. Enable signal Q remains 1, even when input FREE = 0 (light curtain interrupted by product). The MUTING output for setting the muting lamp switches to 1.

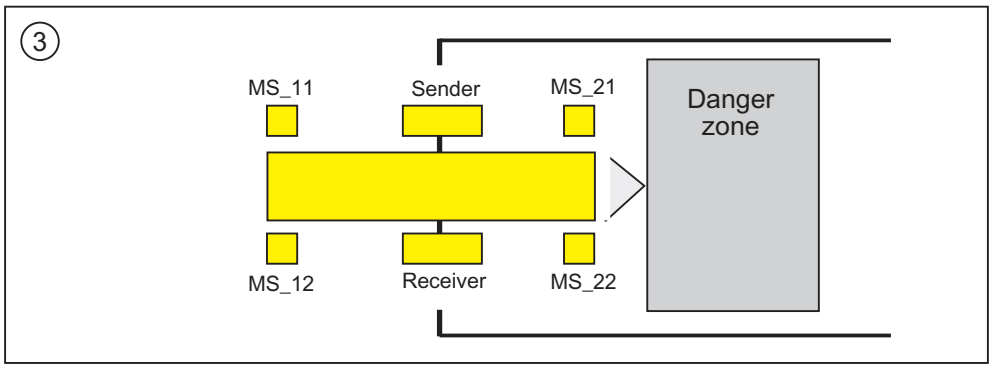
Note

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 signal of the associated F-I/O DB. 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.

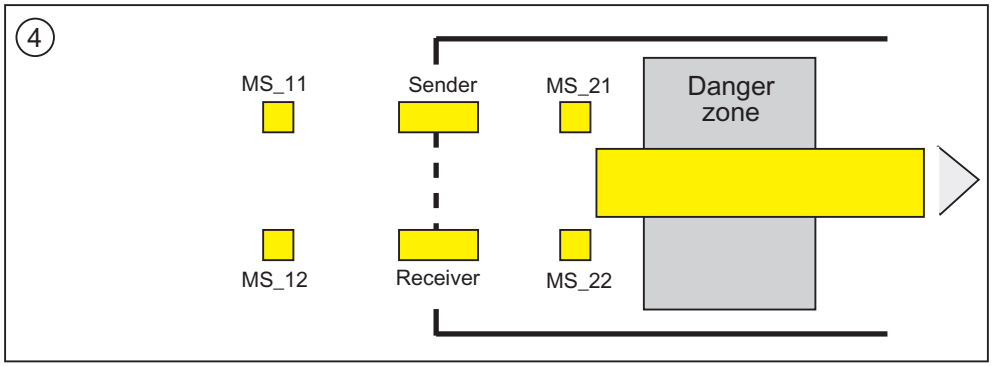
F-I/O that can promptly detect a wire break after activation of the muting operation must be used (see *manual for specific F-I/O*).



- 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).



- 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).

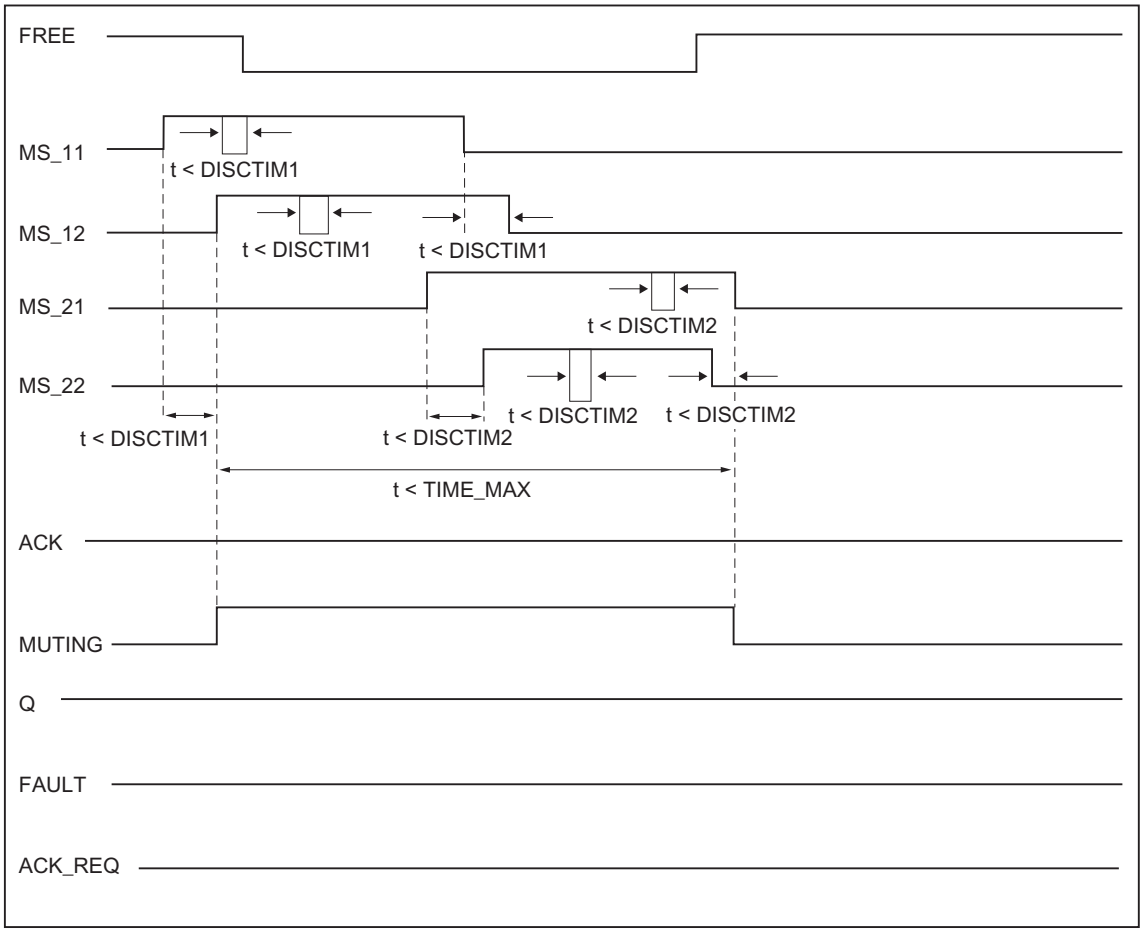


- Only if muting sensors MS_21 and MS_22 are both switched to inactive (product enables sensors) is the MUTING function terminated (Q = 1, MUTING = 0). The maximum activation time for the MUTING function is the time set at input TIME_MAX.

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

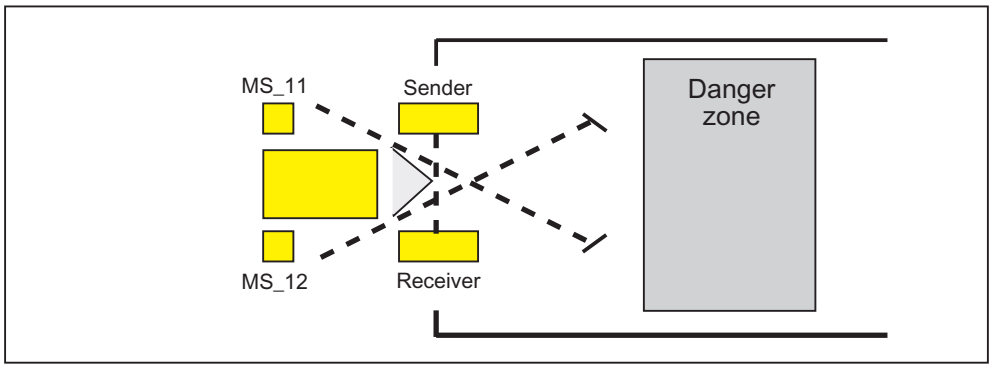


Schematic Sequence of Muting Procedure with Reflection Light Barriers

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 (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.
- 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 if 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 acknowledgment with rising edge at input ACK occurs (see also *Implementing User Acknowledgment in Safety Program of F-CPU of DP Master* and *Implementing User Acknowledgment in Safety Program of F-CPU of I-Slave*).

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 if 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

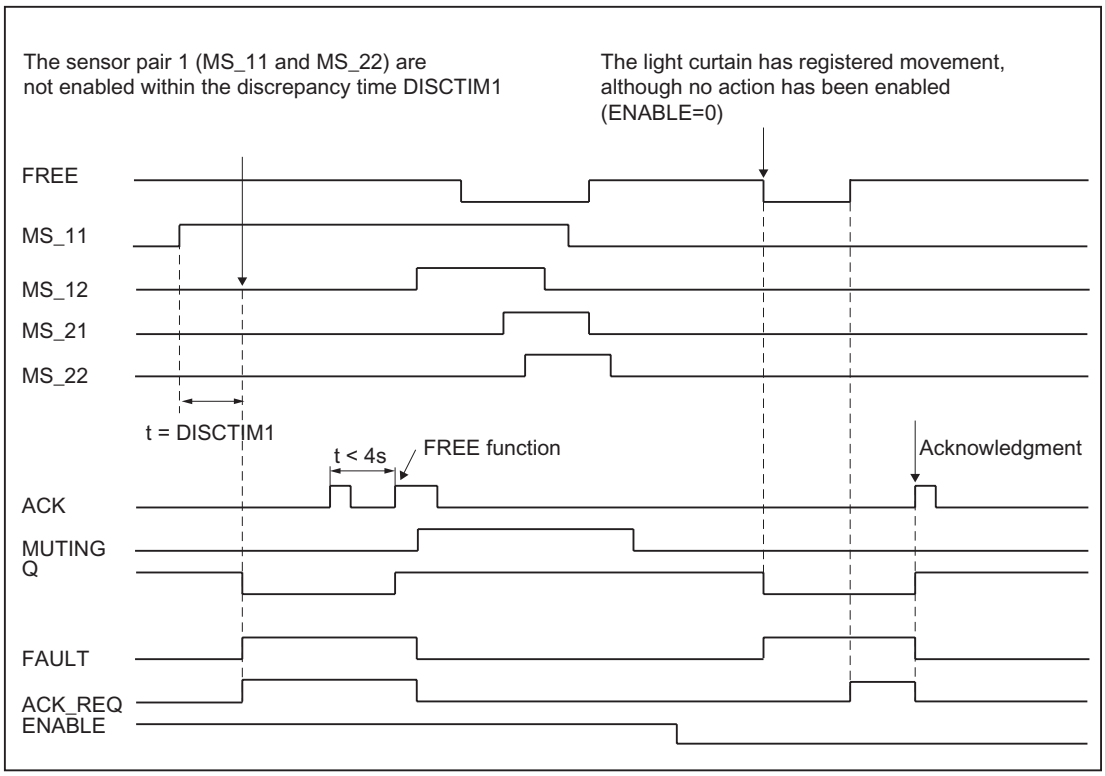
If an error cannot be corrected immediately, the FREE function can be used to free the muting range. Enable signal Q and output MUTING =1 temporarily. The FREE function can be used if:

- 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 (If MUTING Is Not Active)



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 disabled. 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.	Meaning	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 solution, see DIAG variable, Bits 0 to 6 in *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 between 0 s and 10 min.
Bit 3	Light curtain interrupted and muting not active	ENABLE = 0	Set ENABLE = 1
		Light curtain is defective	Check light curtain
		Wiring fault	Check wiring of light curtain (FREE input)
		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 solution, see DIAG variable, Bits 0 to 6 in *F-I/O DB*
		Startup of F-system	For FREE, see DIAG variable, Bit 5
		See other DIAG bits	

Bit no.	Meaning	Possible causes of problems	Remedies
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 solution, see DIAG variable, Bits 0 to 6 in *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!

6.1.2.14 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 *Timing Imprecision …* figure)
- 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 a value of 0" for the discrepancy behavior 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.	Meaning	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 solution, see DIAG variable, Bits 0 to 6 in *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 permanent signal state of 1	Acknowledgment button defective	Check acknowledgment button
		Wiring fault	Check wiring of acknowledgment button
Bit 6	Acknowledgment necessary (= state of ACK_REQ)	-	-
Bit 7	State of output Q	-	-

Note

Access to the DIAG output is not permitted in the safety program!

6.1.2.15 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 of F-I/O DB of output Q	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.

Output Q is set to 1 as soon as input ON = 1. In so doing, feedback input FEEDBACK = 1 and a feedback error cannot be saved.

Output Q is reset to 0, as soon as input ON = 0 or if a feedback error is detected.

Feedback error ERROR = 1 is detected if the signal state of feedback input FEEDBACK (for output Q) does not follow the signal state of input ON within the maximum tolerable feedback time FDB_TIME. The feedback error is saved.

If no discrepancy is detected between the ON and FEEDBACK inputs, the feedback 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 acknowledge the feedback error with a rising edge at input ACK.

The ACK_REQ = 1 output signals that a user acknowledgment is necessary at input ACK to acknowledge the feedback error. The block sets ACK_REQ = 1 as soon as discrepancy is no longer detected. 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 variable of the associated F-I/O DB.

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 F-application block F_FDBACK, 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_FDBACK, F-application block F_TOF must have number FB 186 and must not be renumbered!

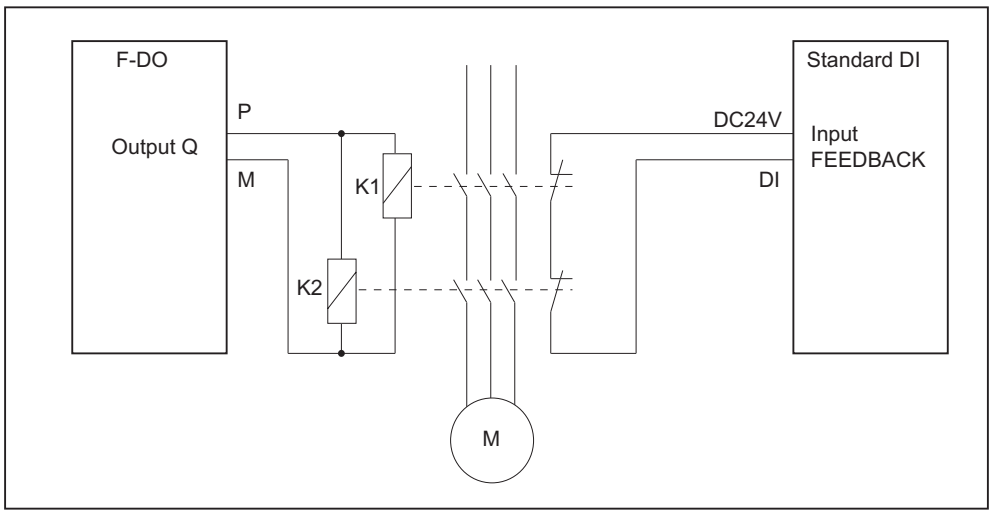
Warning

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 *Timing Imprecision …* figure)
- 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



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, you can evaluate it in your standard user program. DIAG bits 0, 2, and 5 are saved until acknowledgment at input ACK.

Structure of DIAG

Bit no.	Meaning	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 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 solution, see DIAG variable, Bits 0 to 6 in *F-I/O DB*
Bit 2	Following 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 solution, see DIAG variable, Bits 0 to 6 in *F-I/O DB*
Bit 3	Reserved	-	-
Bit 4	Reserved	-	-
Bit 5	If feedback error: input ACK has a permanent signal state of 1	Acknowledgment button defective	Check acknowledgment button
		Wiring fault	Check wiring of acknowledgment button
Bit 6	Acknowledgment necessary (= state of ACK_REQ)	-	-
Bit 7	State of output Q	-	-

Note

Access to the DIAG output is not permitted in the safety program!

6.1.2.16 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 of F-I/O DB of input IN1	0
	QBAD_IN2	BOOL	QBAD of F-I/O DB of input IN2	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 variable of the associated F-I/O DB(s). 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.

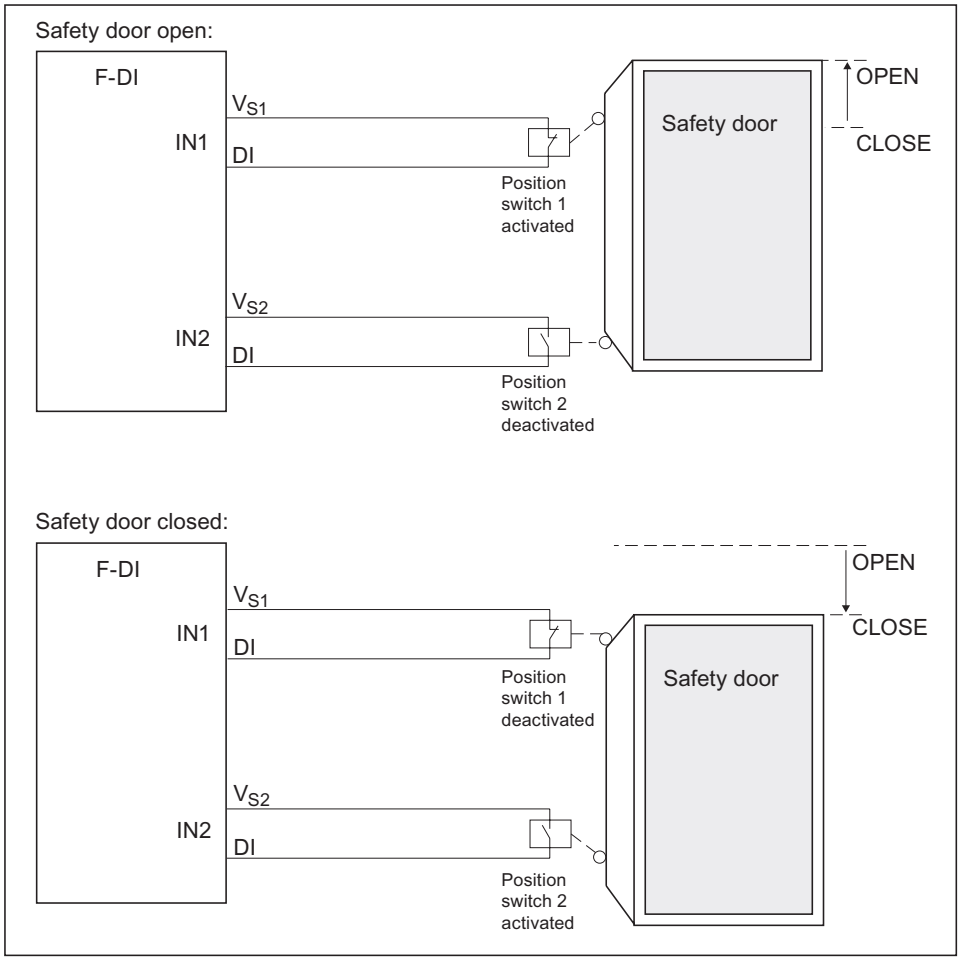
Warning

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.



Startup Characteristics

After an F-system startup, enable signal Q is reset to 0. The acknowledgment for the enable takes place according to the parameter assignment at inputs OPEN_NEC and ACK_NEC:

- When OPEN_NEC = 0, an automatic acknowledgment occurs **irrespective** of ACK_NEC as soon as inputs IN1 and IN2 both assume a signal state of 1 (safety door is closed) for the first time after reintegration of the associated F-I/O (see *Passivation and Reintegration of F-I/O after Startup of F-System*).
- 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.	Meaning	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 of IN1 and/or IN2	For solution, see DIAG variable, Bits 0 to 6 in *F-I/O DB*
Bit 4	Reserved	-	-
Bit 5	If enable is missing: Input ACK has permanent signal state of 1	Acknowledgment button defective	Check acknowledgment button
		Wiring fault	Check wiring of acknowledgment button
Bit 6	Acknowledgment necessary (= state of ACK_REQ)	-	-
Bit 7	State of output Q	-	-

Note

Access to the DIAG output is not permitted in the safety program!

6.1.2.17 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-oriented master to I-slave communication
- Safety-oriented I-slave to 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	Unique value network-wide 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 description)	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 (For a description of error codes, refer to the online Help for SFC 14.)	0
	RETVAL15	WORD	Error code of SFC 15 (For a description of error codes, refer to 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	Unique value network-wide 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 description)	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-application block F_RCVDP.

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"

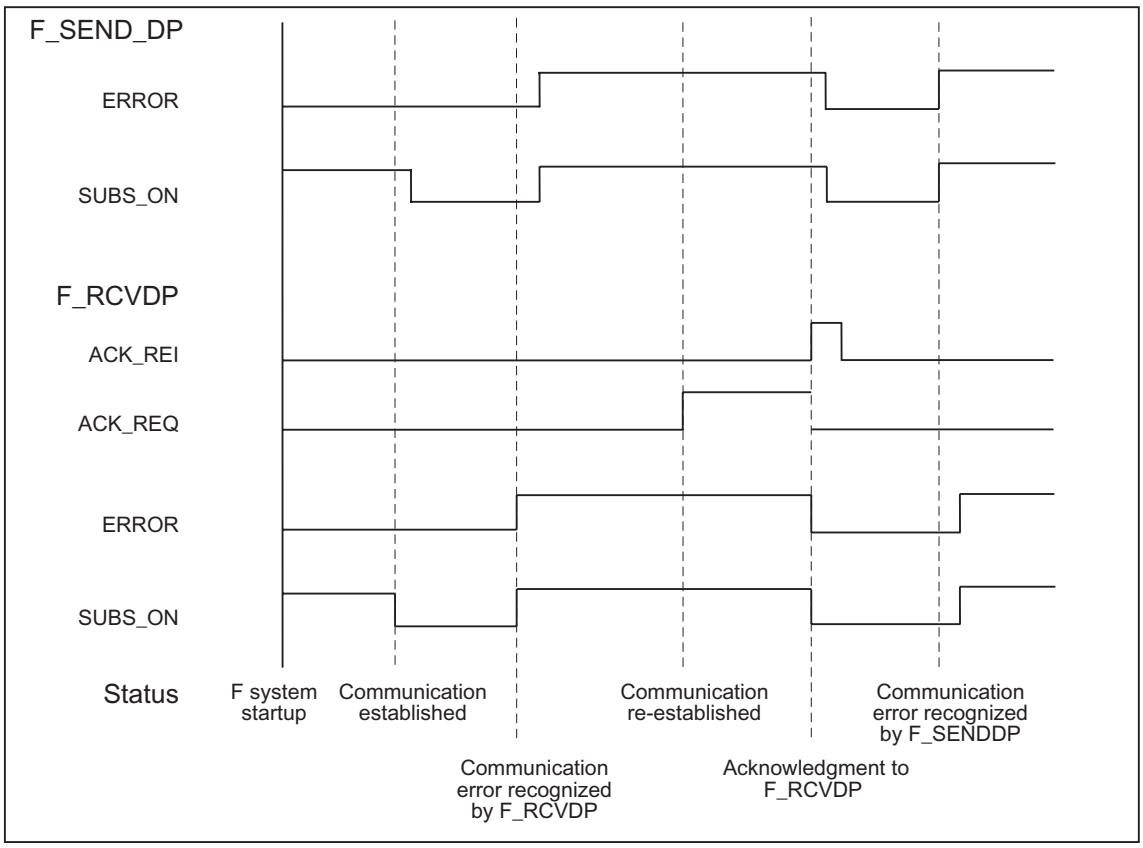
Startup Characteristics

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 the F_SENDDP pending at inputs SD_BO_xx and SUBI_xx are output again only if no more communication errors have been detected (ACK_REQ = 1) and you have used a rising edge to acknowledge at input ACK_REI (see also *Implementing User Acknowledgment in Safety Program of F-CPU of DP Master* and *Implementing a User Acknowledgment in Safety Program of F-CPU of I-Slave*).

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 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, from 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

For additional information on programming safety-related communication between safety programs on different F-CPUs, refer to *Programming Safety-Related CPU-CPU Communication* and *Configuring Safety-Related CPU-CPU Communication*.

6.1.2.18 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 V 5.3, S7 connections are generally permitted over Industrial Ethernet only!

Safety-related communication via S7 connections to and from CPUs 416F-2 is only possible with **firmware version V 3.1.4** or later.

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 description)	0 ms
	EN_SEND	BOOL	1= Send enable	1
	ID	WORD	Local ID of S7 connection (from *NetPro*)	0
	R_ID	DWORD	Unique value network-wide for address association between F_SENDS7 and F_RCVS7	0
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 (SFB 9/FB 9) (For a description of error codes, refer to online Help for SFB 9)	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 description)	0 ms
	ID	WORD	Local ID of S7 connection (from *NetPro*)	0
	R_ID	DWORD	Unique value network-wide for address association 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 (SFB 9/FB 9) (For a description of error codes, refer to online Help for SFB 9)	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 *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 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 disable 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 odd and 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 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 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"

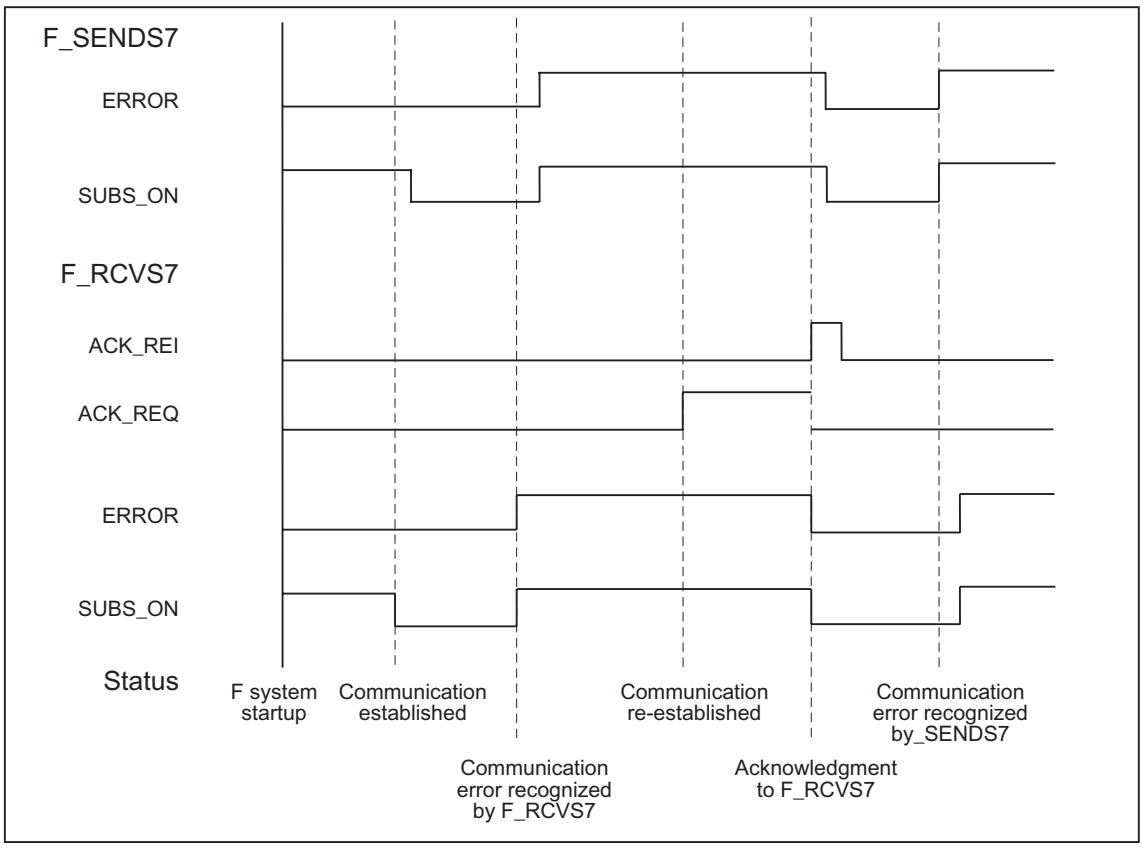
Startup Characteristics

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 located in the F-communication DB of the F_SENDS7 are output again only if no more communication errors are detected (ACK_REQ = 1) and you have used a rising edge to acknowledge at input ACK_REI of the F_RCVS7 (see also *Implementing User Acknowledgment in Safety Program of F-CPU of DP Master* and *Implementing User Acknowledgment in Safety Program of F-CPU of I-Slave*).

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_RECI 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 disabled 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 for Bit 4	See description for Bit 4
Bit 6	CRC error detected by F_SENDS7 and F_RCVS7	See description for Bit 4	See description for Bit 4
Bit 7	Reserved	-	-

Note

Access to outputs DIAG, STAT_RCV, and STAT_SND is not permitted in the safety program!

Additional Information

For more information on configuring safety-related communication via S7 connections, refer to Configuring Safety-Related CPU-CPU Communication via S7 Connections.

For more information on programming safety-related communication via S7 connections, refer to Programming Safety-Related CPU-CPU Communication via S7 Connections.

6.1.2.19 FC 174 "F_SHL_W": Shift Left 16 Bits

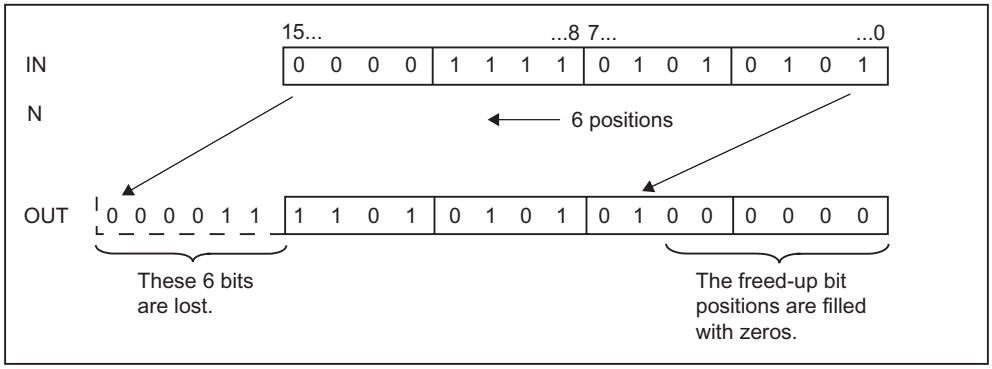
Connections

	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 $16 < N \leq 255$.

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.



6.1.2.20 FC 175 "F_SHR_W": Shift Right 16 Bits

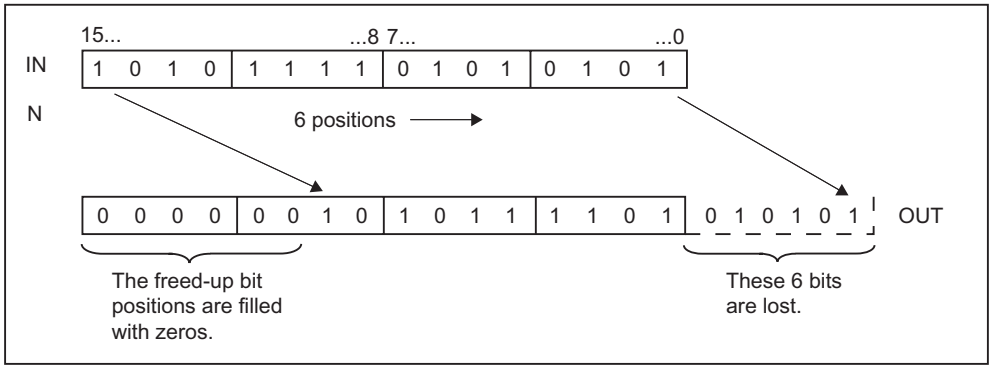
Connections

	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 $16 < N \leq 255$.

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.



6.1.2.21 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).

6.1.2.22 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).

6.1.2.23 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		

6.1.2.24 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. For examples for parameter assignment of ADDR_INT, END_INT, and OFFS_INT, refer to *FC 178 "F_INT_WR": Write Value of Data Type INT Indirectly to an F-DB.*

6.1.3 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
- FISCA_I
- FICTU
- FICTD
- FICTUD
- FITP
- FITON
- FITOF
- FI2HAND
- FIMUTING
- FI1oo2DI
- FI2H_EN
- FIMUT_P
- FIACK_OP
- FISHL_W
- FISHR_W
- FIBO_W
- FIW_BO
- FIINT_WR
- FIINT_RD

F-system blocks are automatically added when the safety program is compiled, and placed in the band of numbers you reserved for the "F-function blocks" (see *Configuration*). This enables an executable safety program to be generated from your safety program.

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* (V1) F-library or the block container of the user project.

6.1.4 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.

You can evaluate particular data from the safety program in the standard user program by means of the symbolic name F_GLOBDB (see *Communication between Standard User Program and Safety Program*).

Warning

Do not copy the F-shared DB from a safety program to another safety program (exception: copying the entire S7 program).

6.2 User-Created 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:

3. In *SIMATIC Manager,* select **File > New**.
4. In the "Libraries" tab, select "F-library" from the "Type" list.
5. Assign a name to the F-library.
6. Specify the "file path."
7. 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 yourself must check whether or not an existing user-created F-library is still up-to-date. If necessary, you must replace a user-created F-library with a newer 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 the 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 custom F-libraries are retained.

7 System Acceptance Test

7.1 Overview of System Acceptance Test

Introduction

During the system acceptance test, all relevant application-specific standards must be adhered to as well as the following procedures. This also applies to systems that are not "subject to acceptance testing."

7.2 Preliminary Acceptance Test for Configuration of F-I/O

Introduction

After you finish configuring the hardware and assigning parameters for the F-I/O, you can perform an initial acceptance test for the F-I/O configuration.

In order to do this, the hardware configuration data must be printed out, checked, and saved together with the overall *STEP 7* project.

Printing Hardware Configuration Data

1. Select the correct F-CPU and 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. In the resulting "Print safety program" dialog, select the "Hardware configuration" option.
4. Select "All" as the print area. The printout will then include the "Module description" and the "Address list." Select the "Including parameter description" option to include your parameter descriptions in the printout.

Checking Hardware Configuration Data

1. Check the safety-related parameters of the F-I/O in the printout.

   These safety-related parameters can be found in the printout for the respective F-I/O. The data are structured differently according to the F-I/O as follows:

   SM 326; DI 24 × 24 VDC (Order No. 6ES7326-1BK00-0AB0), SM 326; DI 8 × Namur, SM 326 DO 10 × 24 VDC/2 A and SM 336; AI 6 × 13 Bit

   - The PROFIsafe source address is always "1" and does not appear in the printout.
   - You determine the PROFIsafe destination address from the address value under "Addresses – Inputs – Start." Divide this address value by "8."
   - The safety-related parameters are found under "Parameters – Basic Settings" or "Parameters – In/Output x."

   ET 200S, ET 200eco Fail-Safe Signal Modules SM 326; DI 24 × 24 VDC (Order No. 6ES7326-1BK01-0AB0 and higher) and SM 326; DO 8 × 24 VDC/2 A PM

   - The PROFIsafe source address is found under "Parameters – F-Parameters – F_Source_Address."
   - The PROFIsafe destination address is found under "Parameters – F-Parameters – F_Destination_Address."
   - The safety-relevant parameters are found under "Parameters – F-Parameters" and "Parameters – Module parameters."

   Fail-Safe DP Standard Slaves

   - The PROFIsafe source address is found under "PROFIsafe – F_Source_Add."
   - The PROFIsafe destination address is found under "PROFIsafe – F_Dest_Add."
   - The safety-related parameters are found under "PROFIsafe."
     For information on handling of any process-related fail-safe parameters, refer to the documentation for the respective DP standard slave.

   Parameters of the F-CPU

   You must use the following settings for safety mode: Level of Protection "1" and "Removable with Password." In addition, you must select the "CPU Contains Safety Program" option. The following is displayed in the printout:

   "Protection level: No protection
   CPU contains safety program"

2. Once the safety-related parameters of an F-I/O module are checked, the parameter CRCs in the printout are sufficient as reference for further acceptance testing. These parameter CRCs have the following appearance (address/F-address = PROFIsafe address):

   S7-300 Fail-Safe Signal Modules (SM 326; DI 24 × 24 VDC, with Order No. 6ES7326-1BK00-0AB0; SM 326; DI 8 × NAMUR; SM 326; DO 10 × 24 VDC/2 A; SM 336; AI 6 × 13 Bit)

   - Parameter CRC (including address): 12345
   - Parameter CRC (excluding address): 54321

   ET 200S, ET 200eco Fail-Safe Modules and S7-300 Fail-Safe Signal Modules (SM 326; DI 24 × 24 VDC, Order No. 6ES7326-1BK01-0AB0 and higher; SM 326; DO 8 × 24 VDC/2 A PM)

   - Parameter CRC: 12345
   - Parameter CRC (excluding F-addresses): 54321

   Fail-Safe DP Standard Slaves

   - F_Par_CRC: 12345
   - F_Par_CRC (excluding F addresses):54321

   F-I/O that are to be assigned the same safety-relevant parameters can be copied during configuration. Then you no longer have to check all of the safety-related parameters individually: It is sufficient to compare every other CRC (for example, "Parameter CRC (excluding address)") of the copied F-I/O with the corresponding CRC of the previously checked F-I/O and to check the PROFIsafe source and destination addresses.

3. Check that the PROFIsafe addresses are unique from one another.

Warning

The switch setting on the address switch of the F-I/O, i.e., its PROFIsafe destination address, must be unique network-wide* and station-wide** (system-wide). You can assign a maximum of 1,022 PROFIsafe destination addresses in a system. That is, a maximum of 1,022 F-I/O can be addressed via PROFIsafe.
Exception: In different I-slaves, F-I/O can have the same PROFIsafe destination address since they are only addressed within the station, i.e., by the F-CPU in the I-slave.

The following restriction applies only to ET 200S F-modules or fail-safe DP standard slaves, in which the predefined PROFIsafe address settings **cannot be changed** in *HW Config*:

If a PROFIBUS network contains ET 200S F-modules or fail-safe DP standard slaves with PROFIsafe addresses that cannot be modified in *HW Config*, you can only operate **one DP master with F-CPU** in this network. Otherwise the system-wide uniqueness of the PROFIsafe addresses cannot be guaranteed.

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

** "Station-wide" means, for one station in *HW Config* (e.g., an S7-300 station or an I-slave)

To determine the PROFIsafe addresses of individual F-I/O, refer to step 1.

7.3 Safety Program Acceptance Test

General Procedure for Safety Program Acceptance Test

1. Create the HW configuration and the safety program. Compile the safety program and save the overall *STEP 7* project.
2. Print the safety program as follows:

- Select the correct F-CPU and S7 program assigned to it.
- In *SIMATIC Manager,* select the **Options > Edit Safety Program** menu command.
  The "Safety Program" dialog will appear.
- Activate the "Offline" button, since the signature of the symbols is included in the footer of the offline safety program printout only.
- Click the "Print" button and select the following options in the resulting "Print safety program" dialog:
  - "Hardware configuration"
  - "Function Block Diagram/Ladder Logic"
  - "Safety Program"
  - "Symbol table"
- Select "All" as the print area for the "Hardware configuration." The printout will then include the "Module description" and the "Address list." Select the "Including parameter description" option to include your parameter descriptions in the printout.

3. Check the printout. The following should be checked, in particular:

   The two collective signatures in the footer of the printout (collective signature of all F-blocks with an F-attribute in the block container and signature of the symbols) must match in all four printouts.

   Check the symbol information in the footer of the printout.

Note

If "Symbols not current" 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 online Help for *STEP 7*). If necessary, you must recompile the safety program.

In the printout of the safety program, the collective signature of all F-blocks with F-attribute in the block container and the collective signature of the safety program must match.

The signatures and initial value signatures of all F-application blocks and F-system blocks and the version of *S7 Distributed Safety* must correspond to those found in Annex 1 of the Certification Report.

Ensure that you have assigned a unique DP_DP_ID parameter throughout the network for all safety-related communication connections for safety-related master-to-master-, master-to-I-slave- and I-slave-to-I-slave communication .

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.

Check to determine whether a validity check was programmed for all data in the safety program from the standard user program.

Check the number of F-runtime groups in the safety program (maximum of 2) and the structure of the F-runtime groups (whether all necessary blocks are present in the F-runtime group).

You must check the following values in the "Safety Program" printout to make sure they correspond to the values you configured or programmed:

- Runtime group information for each F-runtime group:
  - Number of F-CALL
  - Number of called F-program block
  - Number of associated instance DB, if applicable
  - Maximum cycle time of the F-runtime group
  - Number of DB for F-runtime group communication, if applicable
- For each F-I/O module addressed in the F-runtime group:
  - The symbolic name used in the safety program and the number of the F-I/O DB must be part of the initial address of the correct module.
  - The value of F_Monitoring_Time must match the corresponding value of the F-I/O with the same initial address in the "Hardware configuration" printout. The corresponding parameter is called "Monitoring time" for S7-300 fail-safe signal modules and "F_WD_Time" for fail-safe DP standard slaves.

For a listing of all information provided in the printout, refer to *Printing Project Data of the Safety Program*

If these checks reveal any deviations or errors, recompile the safety program and restart the acceptance test at step 1.

4. Download the safety program to the F-CPU (if not yet downloaded). The "Safety Program" dialog must be used to download F-blocks the last time prior to the acceptance test (see *Downloading the Safety Program*).
   Once the safety program has been downloaded to the F-CPU, check the following:

   The online collective signature of all F-blocks with F-attribute in the block container must match those in the accepted offline printout, and no unused F-CALL may be present in the online safety program. The maximum F-CALL blocks in the F-CPU is 2.

   If this is not the case, check to determine whether you downloaded the safety program to the correct F-CPU and repeat this step, if necessary. If the problem persists, recompile the safety program and restart the acceptance test at step 1.
5. Perform a complete function test of the safety program (see *Testing the Safety Program*).
6. For recurring tests, determine whether the F-CPU contains the correct safety program by comparing the online signature of all F-blocks with F-attribute in the block container with those in the accepted offline printout.
   If a programming device or PC with *S7 Distributed Safety* V 5.3 is not available for recurring tests, you can read out the collective signature of the safety program from the F-shared DB by means of an operator control and monitoring system. You can obtain the address in the F-shared DB where the collective signature of the safety program ("F_PROG_SIG" variable) is found from the printout of the safety program. This option should only be used if you do not have to perform a manipulation.

Acceptance Test for Safety Program Changes

Use the same procedure to perform an acceptance test for safety program changes as is used for the safety program acceptance test, with the exception that only the changes must undergo a function test.

Procedure for Identifying Changes in the Safety Program

1. Compare the two collective signatures in the printout of the safety program to be tested with those in the printout of the accepted safety program to find out if there is a safety-related change.
2. If there is a safety-related change, compare the changed safety program offline with the saved accepted program by clicking the "Compare..." button in the "Safety Program" dialog (see *Comparing Safety Programs*). This enables you to identify which F-blocks were changed.

   You obtain detailed information on changes in the F-block by comparing the printouts.

   Changes in the parameters assigned to the F-I/O can be indirectly identified in the printout of the safety program in the "Parameter CRC" information for the associated F-I/O DB. Changes to the initial address can be indirectly identified in the printout in the "Initial address" information for the associated F-I/O DB.

Use of Software Packages with 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.

8 Operation and Maintenance

8.1 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 frames, based on PROFIsafe, for example, and make S7 Distributed Safety F-system available via the bus system (such as PROFIBUS DP or Industrial Ethernet), you have to ensure the safety of the F-system using organizational measures, for example, by monitoring the operation and manual safety switching.

If you use the STEP7 S7-PLCSIM function for the simulation of safety programs (see section, *Testing the Safety Program*), these measures are not necessary because the S7-PLCSIM cannot establish an online connection to real S7 components.

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 cannot be regarded as a safety condition (see also *Programming Startup Protection*).

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.

8.2 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 (see *F-I/O Access*). Otherwise, the F-I/O will remain passivated.

CPU Operating System Update

Checking the CPU operating system for F-acceptance: 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.

IM 151-1 High Feature (ET 200S), Operating System Update

You can use an IM 151-1 High Feature as an interface module for an ET 200S distributed I/O system with F-modules. When using a new operating system for this interface module (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. Following startup of the IM on the PROFIBUS DP, all of the ET 200S F-modules are passivated.
The ET 200S F-modules are 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 (see *F-I/O Access*).

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 more detailed information, refer to the Manuals for F-I/O. 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 *Installing/Removing S7 Distributed Safety V 5.3 Optional Package*.

F-system hardware is removed and disposed of in the same way as with standard S7-300 automation systems; refer to the appropriate *hardware manuals*.

8.3 Guide to Diagnostics

Introduction

This section 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 there is a communication error on the PROFIBUS DP. Illuminates when OB85 and OB121 are programmed and a programming error occurs (e.g., instance DB not loaded) • STP LED on the F-CPU: illuminates if F-CPU is in STOP mode • Fault LEDs on the F-I/O: e.g., SF-LED (group fault LED) illuminates if any fault occurs in the individual F-I/O	*Manuals for* F-CPU and *F-I/O*
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	*Online Help for STEP 7* and *manuals for* F-CPU and *F-I/O*
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	*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	Section on relevant F-application block

Evaluation of the Diagnostic Variable or Parameters of F-I/O DBs or Instance DBs

Note

The following diagnostic variable/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 contained in the previously identified variable/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_24VDC".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.

9 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 section references refer to this documentation.
- "*SD*" stands for the *Safety Engineering in SIMATIC S7* system description.
- "*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.

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	Section *Product Overview*; *SB*, Section 2.5, 3.4	
Selection of sensors and actuators	Requirements for actuators	*F-SMs Manual*, Section 6.5; *F-Modules Manual*, Section 6.5; *ET 200eco Manual*, Section 5.5	
Specification of required safety properties for individual components	• DIN V 19 250 • IEC 61508	*SB*, Section 5.7, 5.8	
Configuration			
Installing the optional package	Requirements for installation	Section *Installing/Removing...*	
Selection of S7 components	Rules for configuration	*SB*, Section 3.2.1, 3.3, 3.4; *F-SMs Manual*, Section 3; *F-Modules Manual*, Section 3; *ET 200eco Manual*, Section 2.2	
Configuration of hardware	• Rules for F-systems • Verification of utilized hardware components based on Annex 1 of Certification Report	Section *Overview of Configuration*, *Particularities for Configuring...*; Annex 1 of Certification Report	
Configuration of F-CPU	• Level of protection, "CPU contains safety program" • Password • Define/set F-specific parameters • Call time for the F-runtime group during which the safety program is to be executed, defined in accordance with the requirements and safety regulations - same as with standard system	Section *Configuring the F-CPU*; S7-300 standard; S7-400 standard; IM 151-7 CPU	

Phase	Requirement/Rule	Reference	Check
Configuration of F-I/O	• Settings for safety mode • Configure monitoring times • Define type of sensor interconnection/evaluation • Define diagnostic behavior • Assign symbolic names	Section *Configuring the F-I/O* and subsequent sections; *SB*, Section 9; *F-SMs Manual*, Section 4, 9, 10; *F-Modules Manual*, Section 4, 9; *ET 200eco Manual*, Section 3, 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	Section *Overview of Programming*, *Structure of the Safety Program...*, *Defining the Program Structure*; Annex 1 of Certification Report	
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 - Insert F-blocks from Distributed Safety F-library (V1) and user-created F-libraries - Safety-related CPU-CPU communication - Communication with the standard user program	Section *Creating F-Blocks in F-FBD/F-LAD*, *Distributed Safety F-Library (V1)* Section *User-Created F-Libraries*	
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	Section *Defining F-runtime Groups* *SB*, Section 9	
Compiling the safety program		Section *Compiling a Safety Program*	
Implementing call of safety program	Call of F-CALL blocks directly in OBs (e.g., OB35)	Section *Defining F-runtime Groups*	

Phase	Requirement/Rule	Reference	Check
Installation			
Hardware configuration	• Rules for mounting • Rules for wiring	Section *Overview of Configuration*, *Particularities for Configuring...*; *F-SMs Manual*, Section 5, 6; *F-Modules Manual*, Section 5, 6; *ET 200eco Manual*, Section 4, 5	
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	Section *Downloading the Safety Program* Section *Comparing Safety Programs*	
Testing safety program	• Rules for deactivating safety mode • Procedures for changing safety program data	Section *Testing the Safety Program*, *Deactivating Safety Mode*	
Changing the safety program	• Rules for deactivating safety mode • Rules for modifying the safety program	Section *Modifying the Safety Program in RUN Mode*, *Deactivating Safety Mode, Deleting the Safety Program*	
Testing the safety-related parameters	Rules for configuration	Section *Printing Out Project Data of the Safety Program*; *F-SMs Manual*, Section 4, 9, 10; *F-Modules Manual*, Section 4, 9; *ET 200eco Manual*, Section 3, 8	
Acceptance test	• Rules and notes on the acceptance test • Printouts	Section *System Acceptance Test*	
Operation, Maintenance			
General operation	Notes on operation	Section *Notes on Safety Mode...*	
Access protection		Section *Access Protection*	
Diagnostics	Responses to faults and events	Section *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 operating system update for the IM 151-1 • Notes on preventive maintenance	Section *Replacing Software and Hardware Components*, *F-I/O Access*; Online Help for *STEP 7*	
Removing, demounting	• Notes for removing software components • Notes for demounting modules	Section *Installing/Removing...*, *Replacing Software and Hardware Components*	

10 Glossary

A

Access Protection
-> Fail-safe systems must be protected from dangerous, unauthorized access. Access to F-systems is protected by assignment of two passwords (for the -> F-CPU and for the -> safety program).

Automatically Generated F-Blocks
These -> F-blocks are generated automatically when the safety program is compiled and can be called, if necessary, to generate an executable safety program from the user's safety program.

C

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
Collective signatures uniquely identify a particular state of the -> safety program. 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 in 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.

D

DB for F-Run-Time Group Communication
-> This F-DB is 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 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. Discrepancy analysis is initiated when different levels are detected for two associated input signals (for nonequivalence testing, when the same levels are detected). After a programmable time interval (so-called -> discrepancy time) has elapsed, a check is made to determine whether the difference has disappeared (for nonequivalence testing, whether the agreement has disappeared). If not, this means that a discrepancy error exists. A discrepancy analysis is carried out between the two input signals of the -> 1oo2 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

The DP/DP coupler is a device for coupling two PROFIBUS DP subnets, which are required for master-master communication between -> safety programs in different -> F-CPUs in S7 Distributed Safety.
At least two F-CPUs are involved in 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.

E

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).

F

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 generated in *HW Config* for each F-I/O during compilation. 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/channel faults
- If F-I/O must be passivated as a result of particular states of the safety program (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

Fail-Safe DP Standard Slaves

Fail-safe DP standard slaves are standard slaves that are operated on PROFIBUS with the DP protocol and the -> PROFIsafe bus profile. They must behave in accordance with IEC 61784-1:2002 Ed1 CP 3/1 and the PROFIsafe profile. A device database file (*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 are equipped with integrated -> safety functions. They behave according to IEC 61784-1:2002 Ed1 CP 3/1 and the PROFIsafe bus profile.

Fail-Safe Modules

Fail-safe modules are ET 200S modules that can be used for safety-related operation (in -> safety mode) in the ET 200S distributed I/O system. These modules are equipped with integrated -> safety functions. They behave according to IEC 61784-1:2002 Ed1 CP 3/1 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 of *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. 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 have an F-attribute (identified in the "Safety Program" dialog box by an "F" in the F-block symbol). Once the -> safety program has been compiled successfully, only the blocks of the -> safety program have the F-attribute.

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 represented 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 out of the -> standard user program. It contains a call for the -> F-PB and calls for the automatically added F-blocks (-> F-SBs, -> automatically generated F-blocks, -> F-shared DB) of the F-runtime group.

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

F-DBs are optional fail-safe data blocks that can be read and written to within the entire -> 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. 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 DP standard slaves

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-Run-Time 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 that 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 that are automatically inserted and called when the -> safety program is compiled in order to create 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

i-Parameter Individual parameter of -> fail-safe DP standard slaves

M

MSR Instrumentation and control technology

N

Nonequivalent Sensor A nonequivalent 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).

P

Passivation For an -> F-I/O module with inputs, when passivation occurs 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.

For an F-I/O module with outputs, when passivation occurs 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 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 the *HW Config* application 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.

R

Reintegration

Switching from fail-safe values (0) to process data (reintegration of an F-I/O module) occurs automatically, or alternatively, following user acknowledgment in the F-I/O DB. The reintegration method depends on the following:

- Reason for -> passivation 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. For an F-I/O module with outputs, the output values provided in the safety program in the PIQ are again transferred by the F-system to the fail-safe outputs.

S

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 an -> F-system is the existence of a safe state for all process variables. For digital -> F-I/O, the value is always "0."

Safety Class

Safety level (AK) in accordance with DIN V 19250 (DIN V VDE 0801):

Safety classes are a way to categorize safety requirements for avoiding and controlling faults.

S7 Distributed Safety can be used in safety mode up to requirement class AK6.

Safety Frame

In -> safety mode, the data are transferred in a safety frame between the -> F-CPU and -> F-I/O, or in safety-related CPU-CPU communication between the F-CPUs.

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 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 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: The sensor signal is read once.
- 1oo2 evaluation: The sensor signal is read twice by the same -> F-I/O and compared internally for the purpose of increasing availability.

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 the Fail-Safe 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 up, all data blocks with -> F-attribute are initialized with values from the load memory (as with a cold restart). This means that saved error information is lost.

The -> F-system automatically performs -> reintegration of the -> F-I/O.

U

User Safety Function

The -> safety function for the process can be provided through a user safety function or a fault reaction function. The user only has to program the user safety function. In the event of an error, if the -> F-system can no longer execute its actual user safety function, it executes the fault reaction function; for example, the associated outputs are deactivated, and the -> F-CPU switches to STOP mode, if necessary.

User-Created F-Libraries

User-created F-libraries are F-libraries created by the user containing F-FBs, F-FCs, and application templates (network templates).

V

Voltage Group

In the ET 200S distributed I/O system: A voltage group is a group of electronic modules that are supplied by one power module.

Index

"PROFIsafe" tab 3-18
"Safety Program" dialog 5-89, 5-90, 5-92
  calling 5-90
  contents 5-90, 5-91

1

1oo2 evaluation with discrepancy analysis ... 6-31

A

Acceptance test for safety program changes . 7-8
Access permission 4-3, 4-4, 4-6, 4-7
  revoking for F-CPU 4-6
  revoking for safety program 4-3
  setting for F-CPU 4-6
  setting for safety program 4-3
Access protection 4-2, 10-1
  overview 4-1
Accessing variables of F-I/O DB 5-31
ACK_NEC 5-27, 5-28
ACK_REI 5-28, 5-29
ACK_REQ 5-28, 5-29
Acknowledgment 5-5
Address area for master-I-slave communication
  assigning 3-27
  specifying 3-27
Address areas 5-7, 5-8, 5-11, 5-12
Address areas for I-slave-I-slave communication 3-32
  assignment 3-32
  specifying 3-32
Address areas for master-I-slave communication 3-27
Address areas for master-master communication 3-23
  specifying 3-23
Address setting 3-15
  PROFIsafe 3-14, 3-15
Applying modifications to safety program ... 5-114
Approvals 1-1
Automatically generated F-blocks 5-96, 10-4, 10-5

B

  Block size 5-96
Band of numbers
  F-data blocks 3-6
  F-function blocks 3-4
Basic knowledge 1-1
  required 1-3
Basic procedure for creating the safety program 5-19
Behavior after communication errors 5-33
Behavior after F-I/O faults and channel faults 5-36
Behavior after startup 5-32
Bidirectional connections 3-25
Bit memory 5-8, 5-70
BOOL 5-8, 5-9

C

Category 10-1
Changes in safety program 7-5
Changing an F-runtime group 5-80
Changing BOOL to WORD 6-75
Changing WORD to BOOL 6-76
Channel faults 10-1
Checking block consistency 5-72, 5-75
Checklist 9-1
Collective signatures 10-1
Communication between standard user program and safety program 5-67
Communication error 5-34, 6-64
  F_SENDDP/F_RCVDP 6-61
Communication via S7 connections 3-38
  configuring 3-38, 3-39
Comparing safety programs 5-116
Compiling the safety program 5-94
Complete function test of safety program .. 5-102
Configuration
  level of protection of F-CPU 3-4
Configuring 3-27, 3-32, 3-38
  connection between two F-CPUs via DP/DP coupler 3-24
  connection via DP/DP coupler 3-23
  fail-safe DP standard slaves 3-17, 3-18

F-I/O 3-13, 3-14, 3-15
F-parameters of F-CPU 3-4
group diagnostics 3-13
overview 3-1
particularities 3-3
PROFIsafe address setting 3-13
safety-related communication via S7 connections 3-38
safety-related I-slave-I-slave communication 3-32
safety-related master-I-slave communication 3-27
safety-related master-master communication 3-25
same as for standard system 3-3, 3-13
symbolic names 3-20
with device database file (*.GSD file) 3-17
Configuring communication via S7 connections 3-38
Configuring I-slave-I-slave communication .. 3-32
Configuring master-I-slave communication .. 3-27
Connection between two F-CPUs
via DP/DP coupler 3-23
configuring 3-24
programming 3-23
Connection table 3-39
Connection via DP/DP coupler 3-23
configuring 3-23
programming 3-27
Connections
F_1oo2DI 6-31
F_2H_EN 6-36
F_2HAND 6-19
F_ACK_OP 6-17
F_BO_W 6-75
F_CTD 6-8
F_CTU 6-7
F_CTUD 6-9
F_ESTOP1 6-50
F_FDBACK 6-53
F_INT_RD 6-79
F_INT_WR 6-77
F_MUT_P 6-39
F_MUTING 6-21
F_RCVDP 6-61
F_RCVS7 ..... 6-67, 6-68, 6-69, 6-70, 6-71, 6-72
F_SCA_I 6-5
F_SENDDP 6-61
F_SENDS7 6-67
F_SFDOOR 6-57
F_SHR_W 6-74
F_TOF 6-15
F_TON 6-13
F_TP 6-12
F_W_BO 6-76
Consistent 5-93
Conventions 1-1
Conversion 2-6, 2-7
to another version of S7 Distributed Safety 2-5
Conveyor system
stop 6-21, 6-29
Count down 6-8
Count up 6-7
Count up and down 6-9
Counters and timers 5-5
CPU operating system update 8-3
CPU-CPU communication 3-2, 3-3, 3-22, 3-23, 5-49, 5-54, 5-60
options for safety-related 3-1
overview of safety-related 3-22, 3-23
safety-related 3-3, 5-49, 5-50, 5-51, 5-52, 5-54, 5-55, 5-56, 5-57, 5-59, 5-60, 5-61, 5-63
CRC 10-1
CRC signature 10-1
Create a pulse 6-11
Create OFF Delay 6-15
Create ON delay 6-13
Creating
F-DB 5-75
F-FB/F-FC 5-72, 5-73, 5-74
Creating a network template 5-71
Creating and editing an F-DB 5-75
Creating and editing an F-FB/F-FC 5-72
Creating F-blocks in F-FBD/F-LAD 5-71
without assignment to an F-CPU 5-71
Creating network templates 5-71
Creating the safety program 5-19
Cycle time 5-83, 5-87
for F-runtime group .. 5-80, 5-83, 5-84, 5-85, 5-86, 5-87

D

Data and parameter types 5-7
Data block 5-16
access 5-8, 5-9, 5-10, 5-11, 5-12, 5-14, 5-16, 5-17
Data transfer
from safety program to standard user program 5-67
Data transfer from standard user program to safety program 5-69
DB for F-runtime group
communication 5-75, 5-80, 5-85, 5-86

defining 5-82
Deactivated safety mode 10-2
Deactivating safety mode 5-106
Defining F-runtime groups 5-80
Defining the program structure 5-21
Deleting safety program 5-119
Depassivation 10-1
Device database file (*.GSD file) 3-17, 3-18, 3-19
configuring 3-17, 3-19
parameters 3-18, 3-19
DIAG
F_1oo2DI 6-31
F_2H_EN 6-36
F_ESTOP1 6-50
F_FDBACK 6-53
F_MUT_P 6-39
F_MUTING 6-21
F_RCVS7 6-67
F_SENDDP/F_RCVDP 6-61
F_SENDS7 6-67
F_SFDOOR 6-57
F-I/O DB 5-26, 5-28
Diagnostic options 8-5
steps for evaluation 8-5
Diagnostic parameter 8-5
evaluation 8-6
Diagnostic variable 8-5, 8-6
evaluation 8-5
Diagnostics 8-5
guide 8-5
Difference between F-FBD/F-LAD programming languages and standard FBD/LAD languages 5-7
Discrepancy analysis 10-2
Discrepancy error at sensor pair 1 6-21
timing diagrams 6-25, 6-28
Discrepancy time 10-2
Distributed Safety F-library V1 6-1
directory 5-1
F-blocks 5-5
overview 6-1
Documentation 1-1, 1-2, 1-3, 1-4, 1-5, 1-7
additional 1-1
scope 1-1
Downloading 5-96
in "Safety program" dialog 5-96
in SIMATIC Manager or FBD/LAD Editor 5-96
safety program 5-96, 5-97, 5-98, 5-99
Downloading in SIMATIC Manager or FBD/LAD Editor
rules 5-99
Downloading to an S7-PLCSIM 5-96
DP/DP coupler 10-2
configuring safety-related master-master communication 3-23
Programming safety-related master-master communication 5-49

E

Editing
F-DB 5-75
F-FB/F-FC 5-72
Emergency STOP up to Stop Category 1 6-50
EN 5-13, 5-18
Enable input 5-18
Enable output 5-18
ENO 5-18
Entering/changing/revoking the password for the safety program 4-3
Evaluation 8-5
diagnostic variable or parameter 8-5
Expert 10-2

F

F_1oo2DI 6-31
F_2H_EN 6-36
F_2HAND 6-19
F_ACK_OP 6-17
F_BO_W 6-75
F_Check_SeqNr 3-17
F_CRC_Length 3-17
F_CTD 6-8
F_CTU 6-7
F_CTUD 6-9
F_Dest_Add 3-17
F_ESTOP1 6-50
F_FDBACK 6-53
F_GLOBDB 5-67, 6-81
F_INT_RD 6-79
F_INT_WR 6-77
F_MUT_P 6-39
F_MUTING 6-21
structure of DIAG 6-21
F_MUTING parallel 6-39
F_Par_Version 3-17
F_RCVDP 6-61, 6-63, 6-64, 6-65, 6-66
behavior in event of communication errors 6-61
programming of safety-related master-I-slave communication 5-54
programming safety-related I-slave-I-slave communication 5-54

programming safety-related master-master communication 5-49
receiving data 6-61
structure of DIAG 6-61
timing diagrams 6-65
F_RCVS7 ..... 5-60, 5-61, 5-63, 5-64, 5-65, 5-66, 6-67
Programming safety-related communication via S7 connections 5-60
F_SCA_I 6-5
F_SENDDP 6-61
behavior in event of communication errors 6-61
programming of safety-related I-slave-I-slave communication 5-54
programming safety-related master-I-slave communication 5-54
programming safety-related master-master communication 5-49
Sending data 6-61
structure of DIAG 6-66
Timing diagrams 6-61
F_SENDS7 ..... 6-67, 6-68, 6-69, 6-70, 6-71, 6-72
Programming safety-related communication via S7 connections 5-60
F_SFDOOR 6-57
F_SHL_W 6-73
F_SHR_W 6-74
F_SIL 3-17
F_Source_Add 3-17
F_TOF 6-16
F_TON 6-13
F_TP 6-11
F_W_BO 6-76
F_WD_Time 3-17
Fail-safe acknowledgment 6-17, 6-18
Fail-safe blocks 5-5
Fail-safe communication 10-1
Fail-safe DP standard slaves 10-3
Fail-safe inputs/outputs of F-I/O 3-3
assigning symbols 3-3
Fail-safe modules 10-3, 10-5
Fail-safe or process data 5-25
Fail-safe outputs
passivating over longer time period 8-3
Fail-safe systems 10-1, 10-3, 10-6, 10-7
Fail-safe value output for F-I/O 5-25
F-application blocks 5-6, 6-2, 6-3, 10-3, 10-5
overview 6-2
F-attribute 10-1, 10-3, 10-8
Fault reaction function 2-2, 10-9
example 2-1
Fault reaction time 10-2, 10-4
FB 179 6-5
FB 181 6-7
FB 182 6-8
FB 183 6-9
FB 184 6-11
FB 185 6-13
FB 186 6-15
FB 187 6-17
FB 188 6-19
FB 189 6-21
FB 190 6-31
FB 211 6-36
FB 212 6-39
FB 215 6-50
FB 216 6-53
FB 217 6-57
FB 223 6-61
FB 225 6-67
FB 226 6-67
F-blocks 5-5, 10-1, 10-3, 10-4, 10-5
deleting 5-119, 5-120
F-runtime group 5-5
FC 174 6-73
FC 175 6-74
FC 176 6-75
FC 177 6-76
FC 178 6-77
FC 179 6-79
F-CALL 5-5, 5-22, 5-81, 5-83, 5-86, 10-4, 10-5
defining 5-80
F-components 3-1
configuration 3-1, 3-2
F-CPU 3-1, 4-6, 4-7, 10-1, 10-2, 10-4, 10-7, 10-8, 10-9
changing existing password for F-CPU 4-6
setting access permission 4-6
F-DBs 10-4, 10-5
Feedback monitoring 6-53
F-FBD 5-7, 5-8, 10-4, 10-5
F-FBD and F-LAD programming languages 5-7
F-FBs 5-77, 5-78, 10-3, 10-4, 10-9
setting know-how protection 5-77
F-FCs 5-77, 5-78, 10-3, 10-4, 10-5, 10-9
setting know-how protection 5-77
F-I/O 3-1, 3-2, 8-3, 8-4, 10-3, 10-5, 10-6, 10-7, 10-8
removal and insertion during operation 8-3
F-I/O access 5-23
by means of process image 5-23
during operation 5-114, 5-115, 5-116
F-I/O DB 3-20, 3-21, 5-26, 8-5, 8-6, 10-3, 10-7
configuration of DIAG 5-26

evaluation of diagnostic variable or parameter....8-5
symbolic names....3-20
F-I/O fault....10-5
F-I/O faults and channel faults....5-36
F-I/O with inputs....5-25
F-I/O with outputs....5-25
F-LAD....5-7, 5-8, 10-4, 10-5
Flash card....5-102, 5-103, 5-105
F-libraries....6-82
user-created....6-82
F-local data....3-4
maximum possible amount....3-8, 3-9
F-modules....10-1
F-parameters of F-CPU....3-4
Basis for PROFIsafe addresses....3-4
configuring....3-4
F-data blocks....3-4
F-function blocks....3-7
F-local data....3-8, 3-9
F-PB....10-3, 10-4, 10-5
F-program block....5-5, 5-80, 5-83, 5-87
defining....5-80
F-relevant tab....3-3
F-runtime group....5-4, 5-5, 5-22, 10-4, 10-5
defining F-runtime groups
Safety-related communication between F-runtime groups....5-84
F-blocks....5-5, 5-6
rules for F-runtime groups....5-80
F-runtime group communication....5-76, 5-84, 5-87
F-SBs....10-4, 10-5, 10-6
F-shared DB....5-6, 5-68, 10-5
F-SMs....10-5
F-system blocks....5-5, 5-6, 6-80, 6-81, 10-1
overview....6-80
F-systems....10-1
Fully qualified DB access....5-7, 5-31
Function test of safety program....5-102

G

Glossary....10-1
Group diagnostics....3-16
for S7-300 F-SMs....3-14
Group passivation....5-39
Guide....1-1

H

Hardware components....2-3
Hardware configuration....3-3
Saving and compiling....3-3
Hardware configuration data....7-1
checking....7-2
printing....7-1
Hardware simulation....5-96

I

Identifying changes in safety program....7-5
Illegal address areas....5-7
Illegal data and parameter types....5-7
Illegal instructions....5-7
IM 151-1 High Feature (ET 200S)....8-3
Implementing a user acknowledgment....5-41, 5-45
in safety program of F-CPU of DP master....5-41
in safety program of F-CPU of intelligent DP slave....5-44
Inconsistent....5-93
Industrial Ethernet....3-23
safety-related communication via....3-23
Information landscape....1-1
position....1-3
Installing....2-5, 2-6, 2-8
readme file....2-5
S7 Distributed Safety....2-5, 2-7, 2-8
Instance DB....8-5, 8-6
evaluation of diagnostic variable or parameter....8-5
Instance-DB....5-7
access....5-7
Instructions....5-12, 5-13, 5-14, 5-18
INT....5-8, 5-9, 5-17
Integrated Help....2-5
Internet....1-7
Service & Support....1-1
SIMATIC documentation....1-1
Interruption of light curtain....6-21
IPAR_EN....5-28, 5-29
IPAR_OK....5-26
i-Parameter....10-6
I-slave-I-slave communication....3-32
configuring....3-33, 3-34

K

Know-how protection....5-77, 5-78, 5-79
for user-created F-FBs and F-FCs....5-77

L

Level of protection of F-CPU....3-4
Configuring....3-4, 3-5

Life cycle of fail-safe automation systems..... 9-1
Light curtain 6-21, 6-24, 6-26, 6-27, 6-28, 6-30
Local data 5-10, 5-11, 5-12
Local ID..... 3-39
of S7 connection 3-38

M

Master-I-slave communication ...3-27, 3-29, 3-32
Configuring..... 3-27, 3-28
Master-master communication..... 3-23
configuring 3-23
Memory requirement..... 5-96
Memory reset..... 5-102, 5-111
Micro Memory Card 5-105
MMC 5-102, 5-105
Modifications to safety program
in RUN mode 5-114
Modifications to standard user program..... 5-114
Modified 5-93
Modifying data in safety program..... 5-110
Modifying values in F-DBs 5-110
Monitoring/modifying variables..... 5-110
MSR..... 10-1
Muting with 4 muting sensors..... 6-21
Muting with reflection light barriers..... 6-21

N

Nonequivalent sensor 10-6

O

Opening F-blocks..... 5-110
Operating system update..... 8-3
Operational safety of system 2-1
preservation of 2-1
Order number..... 1-1
S7 Distributed Safety 1-1, 1-2, 1-3, 1-4

P

Parameters
device database file (*.GSD file) 3-17
F-local data 3-4
F-parameters of F-CPU..... 3-4
Partner ID 3-39
of S7 connection 3-38
PASS_ON.....5-27, 5-28, 5-29
PASS_OUT/QBAD..... 5-26
Passivation 10-6, 10-7
Passivation and reintegration of F-I/O
after communication errors 5-33
after F-I/O faults and channel faults.....5-36
after F-system startup.....5-32
Password
assigning new password for
safety program.....4-3
assignment 4-2
changing existing password for safety
program 4-3
F-CPU 4-6
prompt 4-1
safety program..... 4-3, 4-4, 4-5
validity 4-2
Preface 1-1
Preliminary acceptance test for configuration of
F-I/O 7-1
Preventive maintenance (proof test).....8-3
Principle of operation..... 6-5, 6-31, 6-36, 6-39,
6-50, 6-53, 6-57, 6-73, 6-74, 6-75,
6-76, 6-77, 6-79
F_1oo2DI.....6-31
F_2H_EN.....6-36
F_2HAND 6-19
F_ACK_OP.....6-18
F_BO_W.....6-75
F_CTD.....6-8
F_CTU.....6-7
F_CTUD 6-10
F_ESTOP1 6-50
F_FDBACK.....6-54
F_INT_RD 6-79
F_INT_WR 6-77
F_MUT_P 6-39
F_MUTING 6-21
F_RCVDP.....6-61
F_RCVS7 6-67
F_SCA_I.....6-5
F_SENDDP 6-61
F_SENDS7.....6-67
F_SFDOOR.....6-57
F_SHL_W.....6-73
F_SHR_W 6-74
F_TOF 6-15
F_TON.....6-13
F_TP.....6-11
F_W_BO.....6-76
Principles of safety functions in
S7 Distributed Safety.....2-1
Printing
hardware configuration data 7-1
safety program.....5-121
Printing project data 5-121
Process data or fail-safe values 5-25
Process image.....5-23

Process input image 5-8, 5-9
Process output image 5-9, 5-67, 5-68
Product overview 2-1
PROFIsafe 10-3, 10-6
PROFIsafe address 10-6
PROFIsafe address setting 3-13
Program identification 5-103, 5-105
Program signature 10-1
Programming 5-48, 5-49, 5-54, 5-63
  group passivation 5-39
  overview 5-1
  safety-related CPU-CPU communication 5-48
  safety-related CPU-CPU communication via S7 connections 5-60
  safety-related I-slave-I-slave communication 5-54
  safety-related master-I-slave communication 5-54
  safety-related master-master communication 5-49
  startup protection 5-88
  validity checks 5-69
Project data for safety program 5-121
Proof test 8-4
Proof-test interval 10-6
Protection 5-77
  of know-how of F-FBs/F-FCs 5-77
Protection through program identification ... 5-102
Purpose of documentation 1-1

Q

QBAD 5-31

R

RAM requirement 5-96
  of safety program 5-96
RAM requirement for safety program 5-100
Read INT indirectly from an F-DB 6-79
Reading data from the standard user program
  data that can change during the runtime of an F-runtime group 5-67
readme file 2-5
Ready-made F-functions 5-5
Reasons to access an F-I/O DB 5-26
Reflection light barriers 6-25
Reintegration 10-3, 10-7, 10-8
Reintegration of F-I/O 5-25, 5-32, 5-33, 5-35, 5-37, 5-38, 5-44
  after communication errors 5-33
  after F-I/O faults and channel faults 5-36
  after F-system startup 5-32
  for group passivation 5-39
  programming a user acknowledgment 5-41, 5-44
Removing 2-6
  S7 Distributed Safety 2-5, 8-3, 8-4
Removing S7 Distributed Safety 6-82
Requirement class 10-1
Response time 2-8
  calculation 2-8
Restart characteristics 6-39
  F_MUT_P 6-39
Restart inhibit 6-26, 6-27
  upon interruption of light curtain 6-21
Restart inhibit upon interruption of light curtain 6-45
  F_MUT_P 6-39
Restart protection 5-88
RETVAL14 ..... 8-6, 8-7
RETVAL15 ..... 8-6, 8-7
Reusing created F-blocks 5-71
Rules for downloading F-blocks in SIMATIC Manager or FBD/LAD Editor 5-96
rules for F-runtime groups 5-80
Rules for testing 5-111
Rules for the program structure 5-21

S

S7 connections 3-23, 5-60, 5-61
  Programming safety-related communication 5-60
  safety-related communication via 3-22
S7 Distributed Safety 2-3, 2-4, 2-5, 8-3
  configuring and programming software 2-3
  installing 2-5
  principle of safety functions 2-1
  product overview 2-1
  removing 2-5, 8-3, 8-4
  software requirements 2-5
  start 2-5
  steps for creating program 5-19
S7 Distributed Safety fail-safe system 2-1
  hardware and software components 2-3
  S7 Distributed Safety optional package 2-3
  safety program 2-3
S7-PLCSIM 5-97, 5-98, 5-106, 10-7
  downloading to 5-97, 5-98
Safe state 10-3, 10-7
Safety class 10-1
Safety door monitoring 6-57
Safety frame 10-7, 10-8

Safety function 10-7, 10-9
Safety mode 5-106, 5-107, 5-108, 5-109, 8-1, 10-1, 10-2, 10-7, 10-8
deactivating 5-107, 5-108
of safety program 8-1
Safety program 2-4, 4-3, 4-6, 4-7, 5-19, 5-20, 5-21, 5-22, 5-93, 5-94, 5-95, 5-96, 5-106, 5-116, 5-117, 5-118, 5-119, 5-121, 5-122, 5-123, 5-124, 8-1, 10-1, 10-2, 10-3, 10-4, 10-5, 10-6, 10-7, 10-8
basic procedure for creating 5-19
comparing 5-116, 5-117, 5-118
compiling 5-94, 5-95
deleting 5-119
downloading 5-96, 5-97, 5-98, 5-99
notes on safety mode 8-1
password 4-3, 4-4, 4-5
printing 5-121
rules for the program structure 5-21
setting access permission 4-3
states 5-93
steps for creating program 5-19
structure 5-3, 5-4
testing 5-106
transferring to more than one F-CPU 4-6
Safety protocol 10-1
Safety requirements 2-1
achievable 2-2
Safety-related communication 5-84, 10-1, 10-8
Safety-related communication via S7 connections 3-38, 3-39
configuring 3-38
Programming 5-60
Safety-related CPU-CPU communication 3-1, 3-3, 3-22, 5-5, 5-49, 5-54, 5-60, 5-115, 6-68, 6-72
configuring new 5-114
F_RCVDP 6-61
F_SENDDP 6-61, 6-62, 6-63, 6-64, 6-65, 6-66
options 3-2
overview 3-22, 3-23
programming 5-48
Safety-related I-slave-I-slave communication 3-37, 3-38, 5-54
configuring 3-32
programming 5-54
Safety-related master-I-slave communication 3-31, 3-32
configuring 3-27
programming 5-54, 5-57
Safety-related master-master communication
configuring 3-23
programming 5-50
Safety-relevant parameters 3-3
changing 3-3
Scale INT 6-5
Schematic design 5-1
STEP 7 project 5-1
sending and receiving data via S7 connections 6-67
Sensor evaluation 10-1
Service & Support 1-1
Automation and Drives 1-6
Setting access permission for F-CPU 4-6
SFC46"STP" 8-1
F-CPU transition to STOP mode 8-1
Shift left 16 bits 6-73
Shift right 16 bits 6-74
Siemens Intranet 1-1
SIMATIC documentation 1-1
Signal chart for passivation and reintegration of F-I/O
after F-I/O faults and channel faults 5-36
after F-system startup 5-32
for group passivation 5-39
Signal chart for passivation and reintegration of F-I/O after communication errors 5-33
Signature 10-1
Simulation 5-98
of hardware 5-96
Simulation devices 8-1
use 8-1
Size 5-96
of automatically generated F-blocks 5-96
Software component 2-3, 8-3
replacement 8-3, 8-4
Software packages 7-8
Use in parallel with safety program 7-5
Software requirements 2-5
Standard communication 10-8
Standard mode 10-8
Standard user program 10-4, 10-8
Startup characteristics 6-34, 6-50, 6-53, 6-57
F_1oo2DI 6-31
F_CTD 6-8
F_CTU 6-7
F_CTUD 6-9
F_ESTOP1 6-51
F_FDBACK 6-53
F_RCVDP 6-61
F_SENDDP 6-61
F_SFDOOR 6-57
F_TON 6-14

F_TP 6-11
F_TOF 6-15
Startup of F-system 5-32, 5-88, 10-1
Startup protection 5-88
Startup response
F_RCVS7 6-67
F_SENDS7 6-67
States of safety program 5-93
STEP 7 instructions 5-7
STL 5-73, 5-74
STOP mode 8-1
F-CPU transition to STOP mode with SFC46 "STP" 8-1
with communication function 8-1
with mode selector 8-1
with programming device or PC operator control 8-1
Structure of safety program in S7 Distributed Safety 5-3
Support 1-2, 1-4, 1-6, 1-7
additional 1-5
Supported address areas 5-7
Supported data and parameter types 5-7
Supported instructions 5-7
SW Redundancy 7-5
Symbolic name of F-I/O DB 5-31
Symbolic names 3-20, 3-21
assignment 3-20
for F-I/O DBs 3-20
System acceptance test 7-1
overview 7-1
System test for safety program 7-5

T

Test options 5-106
Testing the safety program 5-113
Testing with S7-PLCSIM 5-110
TIME 5-8, 5-15
Timing diagrams 6-11, 6-13, 6-15, 6-21, 6-33, 6-43, 6-47, 6-61
F_1oo2DI 6-31
F_MUT_P 6-39
F_MUTING 6-21
F_RCVDP 6-61
F_SENDDP 6-61
F_TOF 6-15
F_TON 6-13
F_TP 6-11
Training center 1-5
Transferring safety program to F-CPU 5-102
with Micro Memory Card (MMC) 5-102
with programming device or PC 5-102
Transferring the safety program to more than one F-CPU 4-6
Two-hand monitoring 6-19
Two-hand monitoring with enable 6-36

U

Unidirectional connections 3-23
Universal module 3-25
Unlinked 5-7
DB 5-10, 5-11, 5-13, 5-14, 5-16, 5-17
User acknowledgment 5-44, 5-45, 5-46
by means of acknowledgment key 5-41, 5-46
by means of operator control and monitoring system 5-41, 5-44
for reintegration of F-I/O 5-41, 5-47
upon interruption of light curtain 6-21
User safety function 2-2, 10-9
example 2-1, 2-2
User-created F-libraries 6-82, 10-5, 10-9

V

Validity check 5-69
Variables of an F-I/O DB 5-26
Voltage group 10-9

W

Wiring test 5-111
with flash card 5-102
WORD 5-8, 5-9, 5-10, 5-13, 5-17
Write INT indirectly to an F-DB 6-77

SIEMENS

Siemens AG

A&D AS SM ID
Postfach 1963
D-92209 Amberg

Telefax: +49(9621)80-3103
mailto:doku@ad.siemens.de

Your Address:

Name:
Company:
Position:
Street:
Postal code / Place:
Email:
Phone:
Fax:

Your Feedback as regards the S7 Distributed Safety (Version 10/2004)

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**

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