SIEMENS

SIMATIC S7

Product Information

A5E00169432-01

Edition 12/2002

Manual S7 Distributed Safety, Configuring and Programming A5E00109537-01

Scope

This product information document supplements the *S7 Distributed Safety, Configuring and Programming* manual, A5E00109537-01, Edition 03/2002.

Documentation Package

The manual indicated above and this product information document are included in the documentation package *S7 Distributed Safety*, 6ES7 988-8FB10-8BA0.

Organization of Product Information Document

This product information document is organized in two parts. The first part contains all of the changes in Version 5.2 of the optional package *S7 Distributed Safety* compared to the previous version (V 5.1).

The second part presents corrections to the manual mentioned above that could not be made prior to publication. These corrections apply to Versions V 5.1 and V 5.2 of the optional package *S7 Distributed Safety*.

Cross-References in this Product Information

For sake of brevity, all cross-references to sections of the manual mentioned above are indicated without the manual name (for example, "see manual, Section *6.3*").

All cross-references that do not indicate a specific publication refer to this product information documentation (for example, "see Section *1.2.2*").

Siemens Aktiengesellschaft

Contents

Siemens AG
Automation & Drives
Industrial Automation Systems
Östliche Rheinbrückenstr. 50 76181 Karlsruhe,
Federal Republic of Germany

Siemens Aktiengesellschaft

A5E00169432-01
Printed in the Federal Republic of Germany

1 Changing from *S7 Distributed Safety*, V 5.1 to V 5.2

What's New in *S7 Distributed Safety* V 5.2?

The major innovations in *S7 Distributed Safety* V 5.2 are as follows:

- Support of F-CPUs IM151-F/CPU and CPU 416F-2
- Safety-related communication by means of PROFIBUS-DP (master-slave communication)
- STEP 7 operations JMP, JMPN, RET, and OV in F-FBD/F-LAD
- Length of signature of safety program = 32 bits (collective signature of all F-blocks with an F-attribute in the block container; collective signature of safety program; signatures of symbols)
- "Check block consistency" function
- F-application block F_SCA_I: scaling of values of data type INT
- Expanded functionality of "Compare Program" dialog box
- Expanded printout of safety program project data

Software Requirements for *S7 Distributed Safety* V 5.2

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

STEP 7, V 5.1 + Service Pack 6 or higher

Safety Note

Use of optional package *S7 Distributed Safety* V 5.2 with earlier versions of *STEP 7* is not permissible.

Software Requirements for Configuring F-CPUs

The following software requirements are applicable for configuring F-CPUs for use in *S7 Distributed Safety*:

F-CPU	**Order No.**	***S7 Distributed Safety***	***STEP 7***
IM151-F/CPU	6ES7 151-6FB00-0AB0	V 5.2	V 5.2
CPU 315F-2 DP	6ES7 315-6FF01-0AB0	V 5.2	V 5.1 + Service Pack 6
CPU 416F-2	6ES7 416-2FK02-0AB0	V 5.2	V 5.2

Calculation of Maximum Response Time of Your F-System

Use the Microsoft Excel file provided with *S7 Distributed Safety* V 5.**2** to calculate the maximum response time of your F-system.

Note

When *S7 Distributed Safety* V 5.2 is installed, the MS Excel file supplied with the optional package for V 5.1 (*...\Siemens\STEP7\S7Manual\s7fco\s7fcotib.xls)* is overwritten. If you made your response time calculation entries directly in this file rather than in a copy of the file that you created in a **different** directory, you must save the file from V 5.1 in another directory **before installing *S7 Distributed Safety* V 5.2**.

Otherwise your calculation entries in V 5.2 will be lost when *S7 Distributed Safety* V 5.1 is installed !

If you want to update the calculations for *S7 Distributed Safety* V 5.2, these entries must be manually transferred to the V 5.2 file.

Conversion of *S7 Distributed Safety* from V 5.1 to V 5.2

Reading a Safety Program with *S7 Distributed Safety* V 5.2:

If you would like to use *S7 Distributed Safety* V 5.2 to read, but not change, a safety program created with *S7 Distributed Safety* V 5.1, open the "Safety Program" dialog box with V 5.2. Do **not** compile the safety program.

Note

When opening the dialog "Safety Program", the status: "The safety program is consistent“ is output for a consistent safety program created with *Distributed Safety*, V 5.1, 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 5.2:

If you want to use *S7 Distributed Safety* V 5.2 to change a safety program created with V 5.1, proceed as follows:

1. Compile the safety program with *S7 Distributed Safety* V 5.2 prior to making changes.

   Result: All F-blocks of the *Distributed Safety* library that were used in the safety program and that have been updated in the *Distributed Safety* library of V 5.2 are **automatically** replaced following confirmation.
   The collective signature of all F-blocks and the signature of individual F-blocks change for the following reasons:

   - Length of collective signature has been changed from 16 to 32 bits
   - F-blocks of library have been 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* library through changes in F-block signatures. Modified signatures and initial value signatures of all F-application blocks and F-system blocks must conform to those in Annex 1 of the Report on the Certificate.
   - Furthermore, you can identify whether changes have been made in the safety program. If necessary, the safety program must undergo another acceptance test (see manual, Section *6.3*).

Conversion of *S7 Distributed Safety* from V 5.2 to V 5.1

Safety Note

A safety program compiled with *S7 Distributed Safety* V 5.2 must not be read or edited with *S7 Distributed Safety* V 5.1.

1.1 Configuration

1.1.1 Configuring Safety-Related Communication via PROFIBUS-DP (Master Slave Communication)

Overview

Safety-related communication by means of a DP/DP coupler (master-master communication) is described in the manual.

The following section describes the configuration of an additional safety-related communication option, that is, safety-related communication between safety programs in different F-CPUs. As in a standard system, safety-related communication between the safety program of the F-CPU of a DP master and the safety program(s) of the F-CPU(s) of one or more intelligent DP slaves takes place by means of master-slave connections.

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

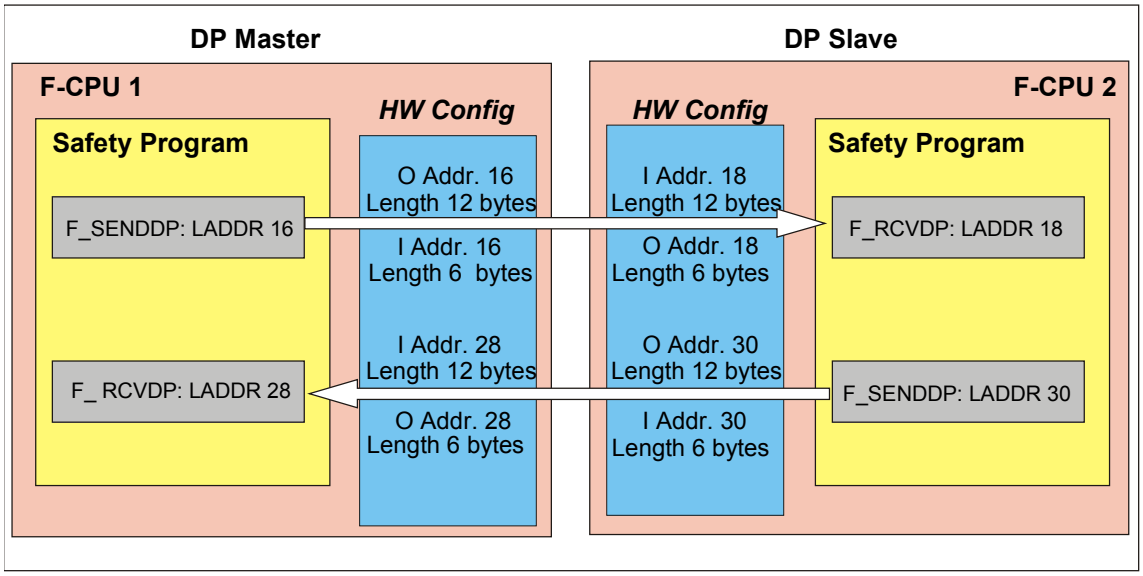
Communication by Means of F_SENDDP and F_RCVDP

Safety-related communication occurs by means of the F-application blocks F_SENDDP and F_RCVDP that you used in the safety programs of the F-CPUs.

Configuring Input/Output Data Areas

You must configure both an output data area and an input data area in *HW Config* for each communication connection between two F-CPUs. In the figure below, each of the two F-CPUs is supposed to be able to send and receive data. You must therefore configure two output data areas and two input data areas for each F-CPU.

You assign the configured start addresses of the input and output data areas to the LADDR parameter of the corresponding F-application blocks F_SENDDP and F_RCVDP in the safety programs (see manual, Section *5.4*).



Rules for Defining the Data Areas

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

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

Procedure for Configuring Master-Slave Communication

The same procedure is used to configure safety-related communication between a DP-master and intelligent DP slaves as is used to configure master-slave communication in a standard system.

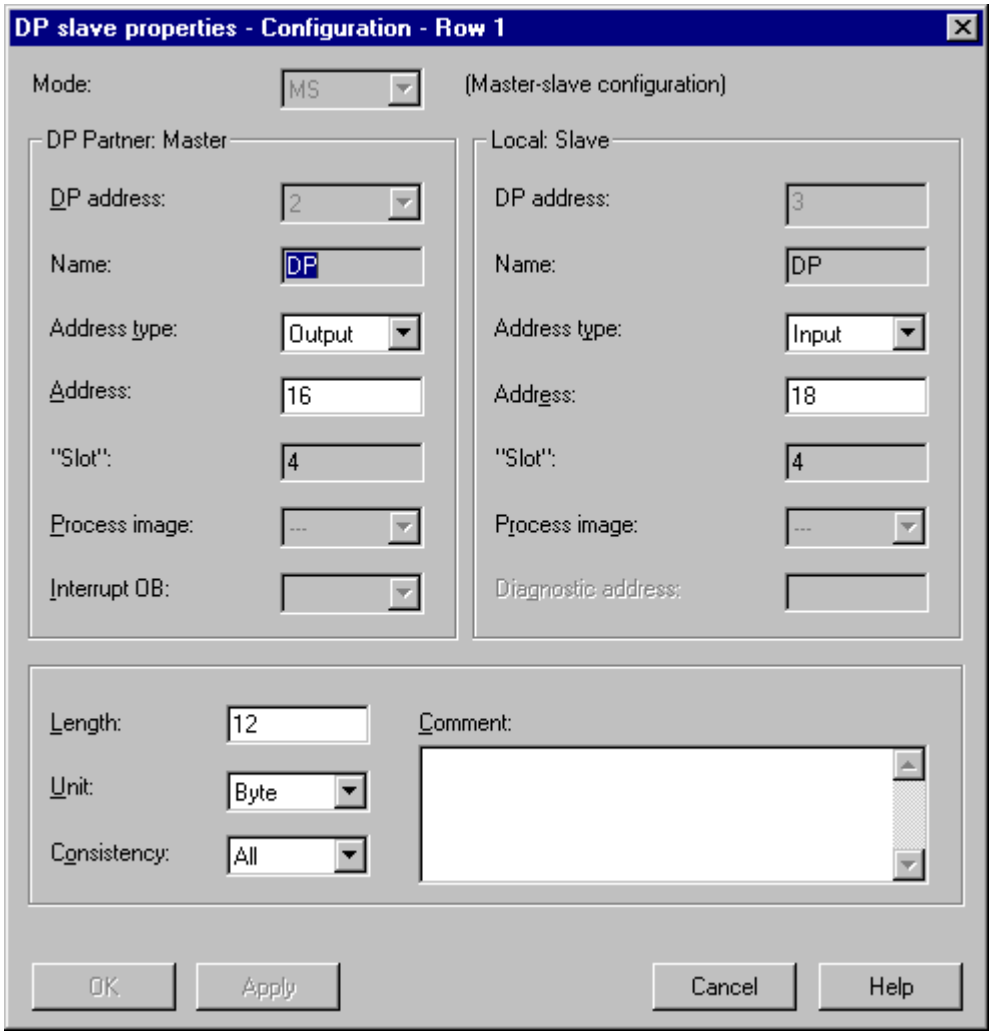
The figure below illustrates the configuration procedure for the address areas in the previous figure.

Requirement:

You have created a project in STEP 7.

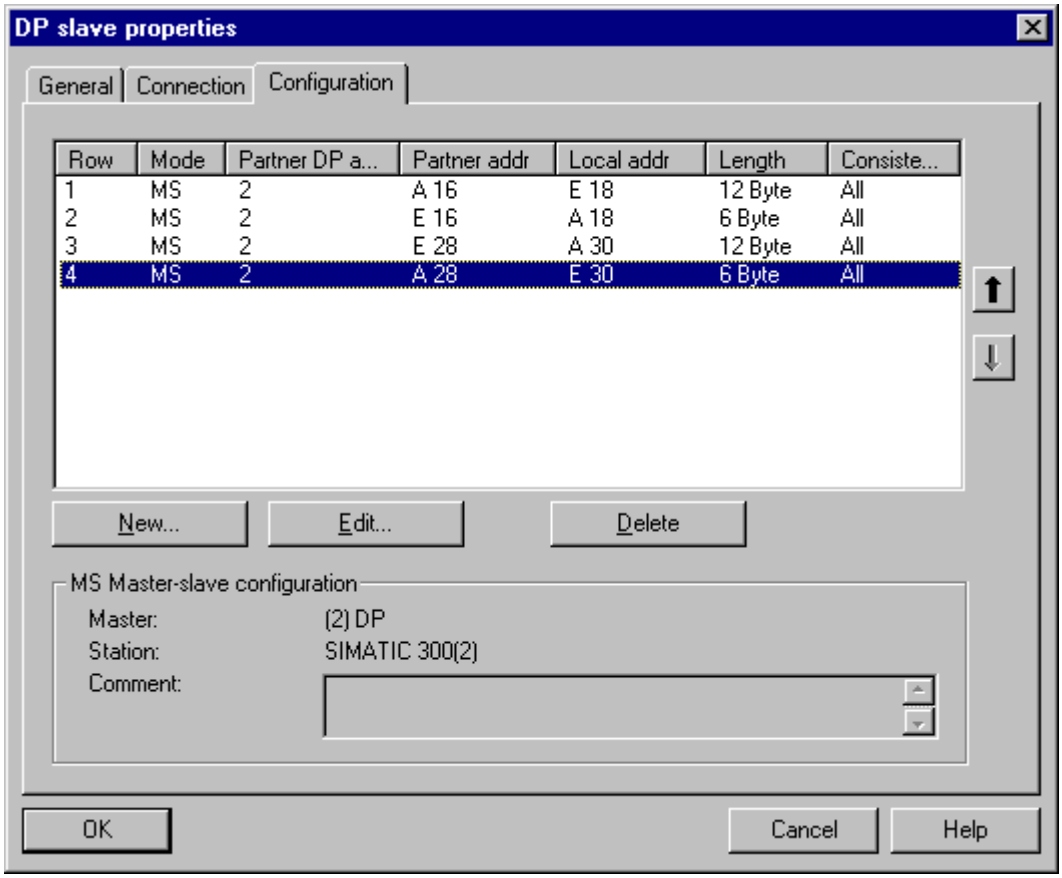
1. Create a station in your project (in *SIMATIC Manager,* for example, an S7-300 station).
2. Assign a CPU with fail-safe capability to this station (in *HW Config*, from the hardware catalog).
3. Configure this CPU as a DP slave (in *HW Config*, in the Object Properties of the DP interface of the CPU).
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 Object Properties of the DP interface of the CPU).
6. In the hardware catalog, under "Configured stations," select the station type of the intelligent DP slave ("CPU 31x" in this example) and position it on the DP master system.
7. Link the intelligent DP slave to the DP master in the Connection dialog box, which is displayed automatically.
   Now, you can specify the input and output data areas for the master-slave communication:
8. In the "Configuration" tab of the Object Properties of the intelligent DP slave, select "New."
9. Enter an output data area for the DP master and the associated input data area for the intelligent DP slave. For this example, make the following entries:
   - For "DP partner: Master", enter "Output" for Address type, "16" for Address, "12" for Length, "Byte" for Unit, and "Total" for Consistent.
   - For "Local: Slave", enter "Input" for Address type and "18" for Address.

   The dialog box has the following appearance:



10. Confirm your entries with "OK."

11. Continue following steps 8 and 9 until all output and input data areas are defined.
    This results in four configuration rows for this example:



Note

Be sure to use identical values for the start addresses of the output and input data areas.

A total of 12 bytes (consistent) must be configured for the output data area, and 12 bytes (consistent) for the input data area.

Always select the "Consistency: All" option for all input and output data areas.

Additional Information

Information on programming safety-related communication by means of PROFIBUS-DP (master-slave communication) can be found in Section *1.2.6*.

Additional information on communication by means of PROFIBUS-DP (master-slave communication) can be found in *STEP 7 Help*.

1.1.2 Configuring the F-CPU

Information on Local Data

The information below on defining local data for the safety program **completely replaces** the corresponding information in Section *3.3* of the manual.

"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. Use this parameter to define the amount of local data in bytes that are available for the automatically added F-blocks.

Note

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 generated anyway.

Safety Note

The calculated maximum runtime of the safety program using the MS Excel file in the directory (*...\Siemens\STEP7\S7Manual\s7fco\s7fcotib.xls)* 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 have configured for the max. cycle time of the F run-time group (F-monitoring time) to calculate the max. reaction times in the event of a fault and for any runtime of the standard system using the above-mentioned Excel file.

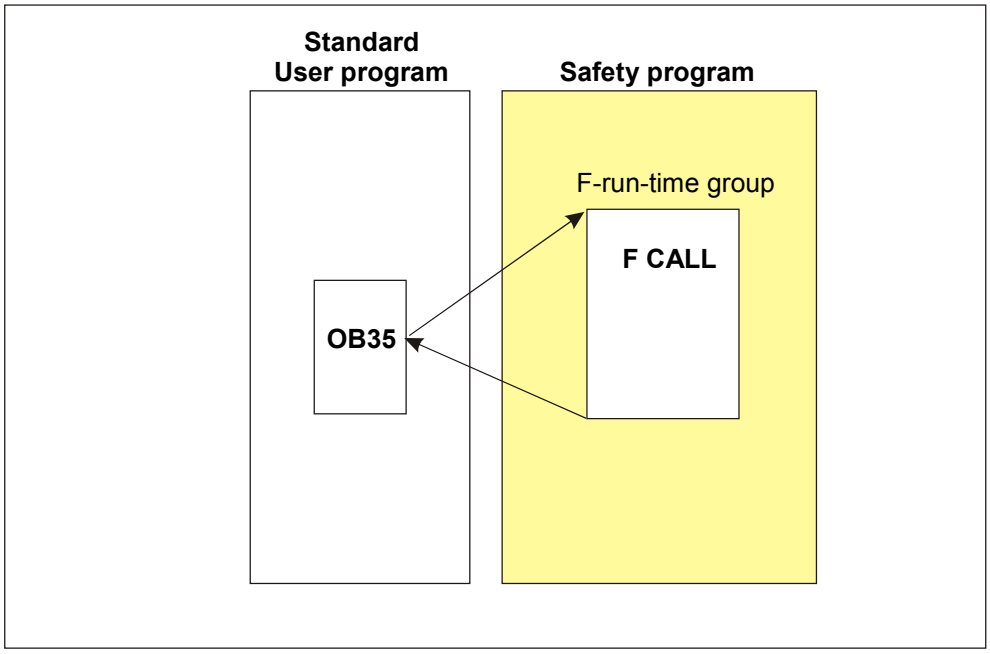
Note

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 block directly in the OB (cyclic interrupt OB 35, if possible) and not declare any additional local data in the cyclic interrupt OB.
- Maximum amount of local data of the utilized F-CPU (see *Technical Specifications in the Product Information*)
  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 class in which the safety program is called (for example, OB 35).

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

Case 1: F-CALL called directly in the OB

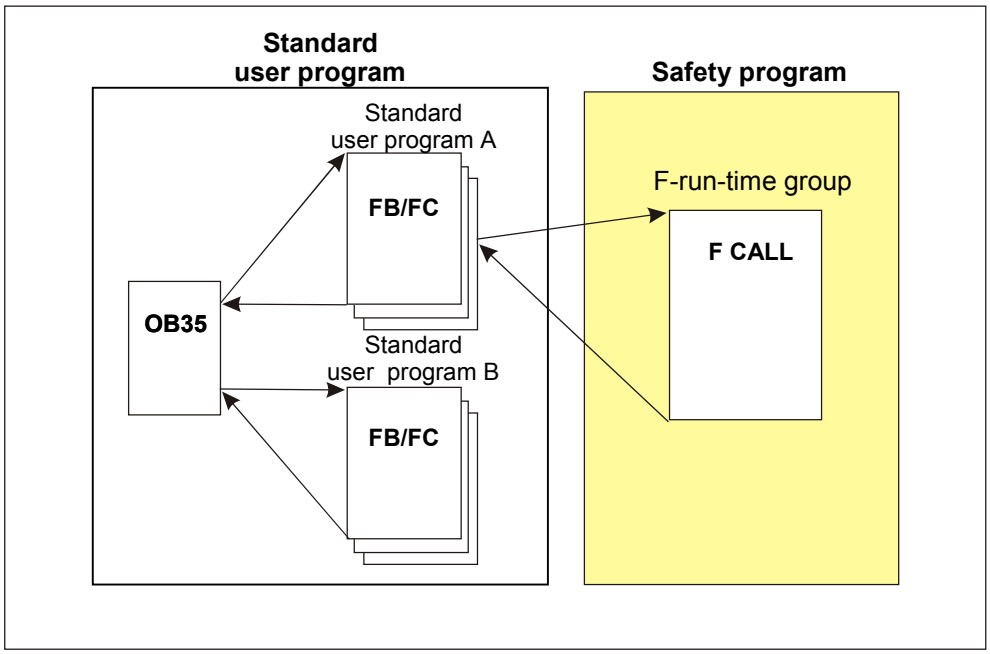


Set the "F-local data" parameter to the following:

Maximum amount of local data of the utilized F-CPU
Minus local data requirement of OB (at least 22 bytes).

Comment: You can derive the local data requirement of the OB 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 not called directly in the OB



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

Maximum amount of local data of the utilized F-CPU
Minus local data requirement of the OB (at least 22 bytes) and
Minus local data requirement of the standard user program A.

Comment: You can derive the local data requirement of the OB or 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:

Determine the local data requirement for each call hierarchy (path 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 in the path

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

\+ 22 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 is programmed

\+ 24 x number of nesting levels in which an F-FB, F-FC, or F-Application block is called with a parameter of data type TIME

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

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, you must observe the following limit (irrelevant for F-CPUs of the S7-400 range):

Local data requirement< max. size of local data per block
(see *Technical Specification about the F-CPU used in the Product Information*

Mean local data requirement in bytes =

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

+ 4 x amount of local data of F-FB/F-FC of data type INT

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

+ 12

+ 14 (if a fixed-point function is programmed)

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

+ 24 (if a called F-FB, F-FC or F-application block contains a parameter of data type TIME)

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.

1.2 Programming the Safety Program

1.2.1 Differences between F-Programming Languages and Standard Programming Languages

Access to Data Blocks

Data blocks should always be accessed with "fully qualified DB access" to ensure that the correct data block is opened.

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

Options for the data blocks "Unlinked" and "DB is write-protected in the AS"

Note

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

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

If you have set the above-mentioned option, this error will be corrected during the generation of the safety program.

Boolean Constants "0" and "1"

If your safety program requires the boolean constants "0" and "1" for initializing parameters for block calls, you can use fully qualified DB access to access the variables "VKE0" and "VKE1" in the F-shared-DB ("F_GLOBDB".VKE0 or "F_GLOBDB".VKE1).

1.2.2 FBD/LAD Operations

New Operations Supported

You can use the operations listed in the table below in the safety program.

Operation		Function	Description
F-FBD	**F-LAD**		
JMP	--(JMP)	Jump operation	Unconditional jump in block Jump in block if 1 (conditional)
JMPN	--(JMPN)	Jump operation	Jump in block if 0 (conditional)
RET	--(RET)	Programmed control	Return (exit block)
OV	OV --| |--	Status bit	Evaluate exception bit overflow (OV bit in status word)

Note

- An F_SENDDP call must not be programmed between a jump operation and its associated destination.
- For F-PB only: A RET operation must not be programmed before a F_SENDDP call.

Safety Note

You have to note the following if you call the following F application blocks in your safety program: F_TP, F_TON, F_TOF, F_ACK_OP, F_2HAND or F_MUTING and use the operations JMP, JMPN or RET:

- Every call of the F application blocks listed may only be edited in one F run group cycle. This means the calls of the F application blocks listed may not be edited several times (loop) by the JMP or JMPN operations.
- As soon as a time (PT, time in F_ACK_OP, DISCTIME, DISCTIM1/2 or TIME_MAX) is started and not yet terminated, a call of one of the listed application blocks has to be edited in every F run group cycle. This means the call of the listed F application blocks are not to be skipped by the JMP, JMPN or RET operations (nesting).

It is therefore recommended to precisely edit the calls of the listed application blocks in every F run group cycle once, otherwise the times will not be updated correctly if the mentioned restrictions are not taken into account.

Please also note the functionality and the time chart of the listed F application blocks (see manual, chapter *5.7.3*).

OV Bit Evaluation

By evaluating the OV bit, you can identify an overflow without the F-CPU going into STOP mode in the case of an overflow.
(If you do not evaluate the OV bit, an overflow causes the F-CPU to switch to STOP mode if the result/quotient in an output is fed to an F-I/O, or to a partner F-CPU by means of safety-related CPU-to-CPU communication.)

If you want to program an OV bit scan, observe the following conditions:

Note

An OV bit scan is only permitted in the network following the network with the operation that affected the OV bit.

The network with the OV bit scan must not be the destination of jump operation, that is, it must not contain a jump label.

If an OV bit scan is programmed in the network following the operation that affected the OV bit, the execution time of the operation affecting the OV bit is prolonged (see also *Excel file for response time calculation* in the *...\Siemens\STEP7\S7Manual\s7fco\s7fcotib.xls* directory).

Safety Note

If an OV bit scan is programmed in the network following the operation that affected the OV bit and the result of the operation affecting the OV bit (an ADD_I, SUB_I, MUL_I or NEG_I operation or the quotient of a DIV_I operation) is outside the permissible range for integers (16 bits), the F-CPU does **not** switch to STOP mode.

The result/quotient behaves like the analogous operation in a standard user program.

1.2.3 F- I/O DB

Defaults Settings for QBAD and PASS_OUT

Contrary to the description in Section *5.3.2* of the manual, the default setting for the variables QBAD and PASS_OUT of the F-I/O DB is "1".

The default setting of "1" has no effect on safety programs that were created with *S7 Distributed Safety* V 5.1.

1.2.4 Use of Fail-Safe Values

SM 336; AI 6 x 13-Bit: Fail-Safe Value Output

The F-system identifies 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 $7FFF_H$ (for overflow) or 8000_H (for underflow) in the PII for the safety program.

Use of Individual Fail-Safe Values

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 (see manual, Section *5.3.2*).

1.2.5 Implementation of User Acknowledgment

This section supplements Section *5.3.9* of the manual.

1.2.5.1 Implementation of User Acknowledgment in the Safety Program of the F-CPU of a DP Master

See manual, Section *5.3.9.*

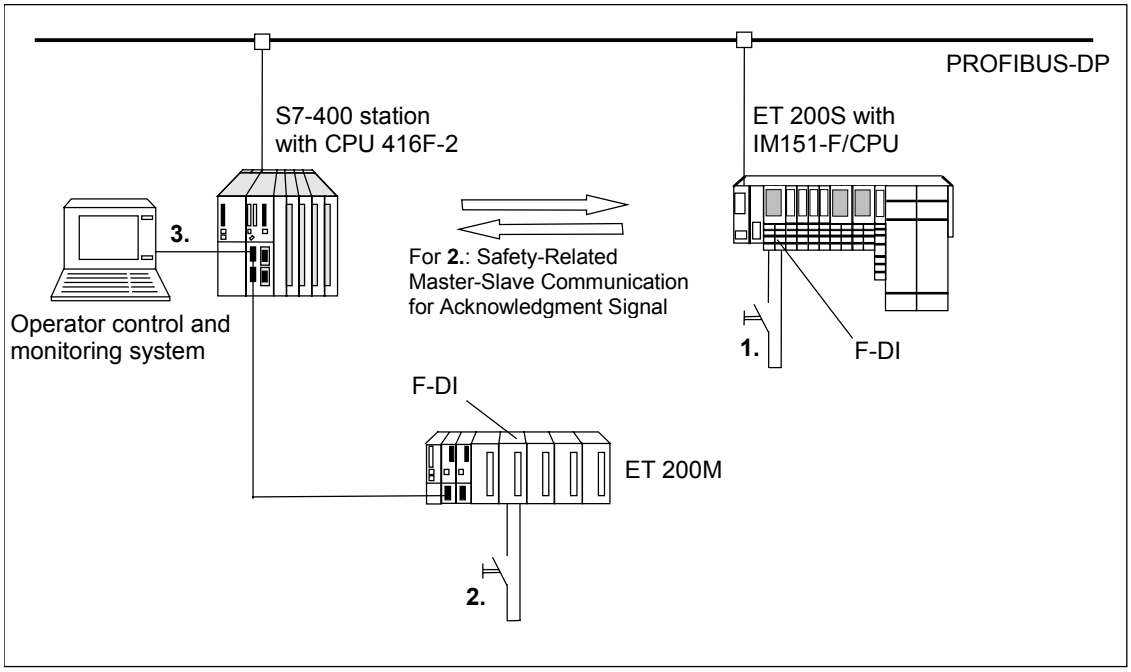
1.2.5.2 Implementation of User Acknowledgment in the Safety Program of the F-CPU of an Intelligent DP Slave

Options for User Acknowledgment

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

1. An acknowledgment key that you connect at an F-I/O with inputs that is assigned to the F-CPU of the intelligent DP slave
2. An acknowledgment key that you connect at an F-I/O with inputs that is assigned to the F-CPU of the DP master
3. An operator control and monitoring system 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 Acknowledgment Key at an F-I/O with Inputs That is Assigned to the F-CPU of the Intelligent DP Slave

Note

If a communication error, F-I/O fault, or channel fault occurs in the F-I/O to which the acknowledgment key is connected, 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 intelligent DP slave.

For this reason, it is recommended that you provide for an additional acknowledgment by means of the F-CPU on the DP master to enable acknowledgment for reintegration of an F-I/O to which an acknowledgment key is connected (see manual, Section *5.3.9*).

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

If you want to use the acknowledgment key assigned to the F-CPU on the DP master for an additional user acknowledgment in the safety program of the F-CPU of an intelligent 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 intelligent DP slave by means of safety-related master-slave communication (see Section *1.1.1* and *1.2*.6).

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

1. Call the F-application block F_SENDDP in the safety program in the F-CPU of the DP master (see manual, Section *5.4*).
2. Call the F-application block F_RCVDP in the safety program in the F-CPU of the intelligent DP slave (see manual, Section *5.4*).
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 using 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 "VKE0," 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. VKE 0 is provided in the F-shared-DB. At input SUBBO_xx, enter fully qualified "F_GLOBDB".VKE0.

Note

If a communication error, F-I/O fault, or channel fault occurs in the F-I/O to which the acknowledgment key is connected, acknowledgment for reintegration of this F-I/O will also no longer possible.

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

For this reason, it is recommended that you provide for an additional acknowledgment by means of an operator control and monitoring system to enable acknowledgment for reintegration of the F-I/O to which an acknowledgment key is connected (see manual, Section *5.3.9*).

If a safety-related master-slave communication error occurs, the acknowledgment signal is not transmitted, and consequently, acknowledgment for reintegration of 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 intelligent DP slave.

For this reason, it is recommended that you provide for an additional acknowledgment by means of an acknowledgment key at an F-I/O with inputs on the intelligent DP slave to enable acknowledgment for reintegration of safety-related communication for transmission of the acknowledgment signal.

Then a STOP/RUN transition of the F-CPU of the intelligent DP slave is only necessary if a safety-related master-slave communication error is pending at the same time as a communication error, F-I/O fault, or channel fault of the F-I/O on the intelligent DP slave.

3. User Acknowledgment by Means of an Operator Control and Monitoring System Assigned to the F-CPU of the DP Master

The F-application block F_ACK_OP from the *Distributed Safety* library is required to implement a user acknowledgment by means of an operator control and monitoring system (see manual, Section *5.7.3.7*).

Procedure for Programming User Acknowledgment by Means of an Operator Control and Monitoring System Assigned to the F-CPU of the DP Master

1. Proceed initially as described in the manual, Section *5.3.9* under *Procedure for Programming User Acknowledgment by Means of an Operator Control and Monitoring System*.
   In so doing, note that the acknowledgment signal for evaluating user acknowledgments is provided at output **OUT** of F_ACK_OP **("Output Q" specified in the manual is incorrect!).**
   Also note the associated safety notes in the manual.
2. The acknowledgment signal at output OUT of F_ACK_OP is only set to 1 for one cycle of the safety program in the F-CPU of the DP master.
   Before the output signal is transferred to the safety program in the F-CPU of the intelligent DP slave, you must lengthen the signal with the F-application block "F_TP." Select the time value you assign at the TIMEOUT input for the duration at the PT input of the "F_TP."
3. Transfer the signal at output Q of the "F_TP" using the procedure for transferring the acknowledgment key, as described in Section "2. User Acknowledgment by Means of Acknowledgment Key at an F-I/O with Inputs that is Assigned to the F-CPU of the DP Master."

1.2.6 Programming the Safety-Related Communication by Means of PROFIBUS-DP (Master-Slave Communication)

Overview

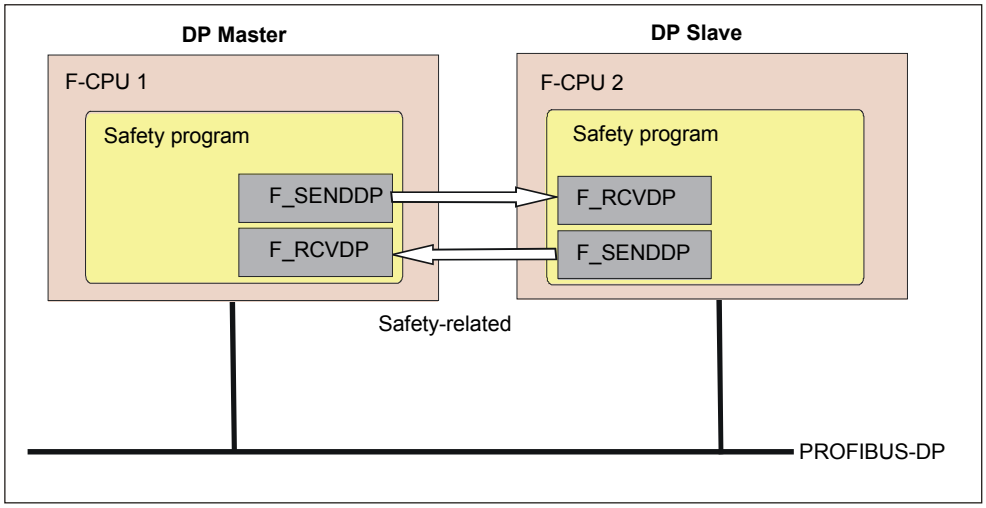
Safety-related communication by means of a DP/DP coupler (master-master communication) is described in the manual.

The procedure for programming safety-related master-slave communication is the same as for programming safety-related master-master communication by means of a DP/DP coupler. For this reason, only the differences are described in the following section.

Communication by Means of F_SENDDP and F_RCVDP

As in a standard system, safety-related communication between the safety program of the F-CPU of a DP master and the safety program(s) of the F-CPU(s) of one or more intelligent DP slaves takes place by means of master-slave connections.

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



Safety-related communication is carried out with F-application blocks F_SENDDP for sending and F_RCVDP for receiving. They can be used to transfer safely a *fixed* amount of fail-safe data of the data types BOOL and INT.

These library blocks are know-how protected. You can find them in the *F-Application Blocks* container in the *Distributed Safety* (V1) library. The F_RCVDP **must** be called at the start of the F-PB, and the F_SENDDP at the end of the F-PB.

A detailed description of the F-application blocks F_SENDDP and F_RCVDP can be found in the manual, in Section *5.7.3.10*.

Procedure for Assigning DP Master/DP Slave to F_SENDDP/F_RCVDP

The DP master/intelligent DP slave can be assigned to an F_SENDDP/F_RCVDP as follows:

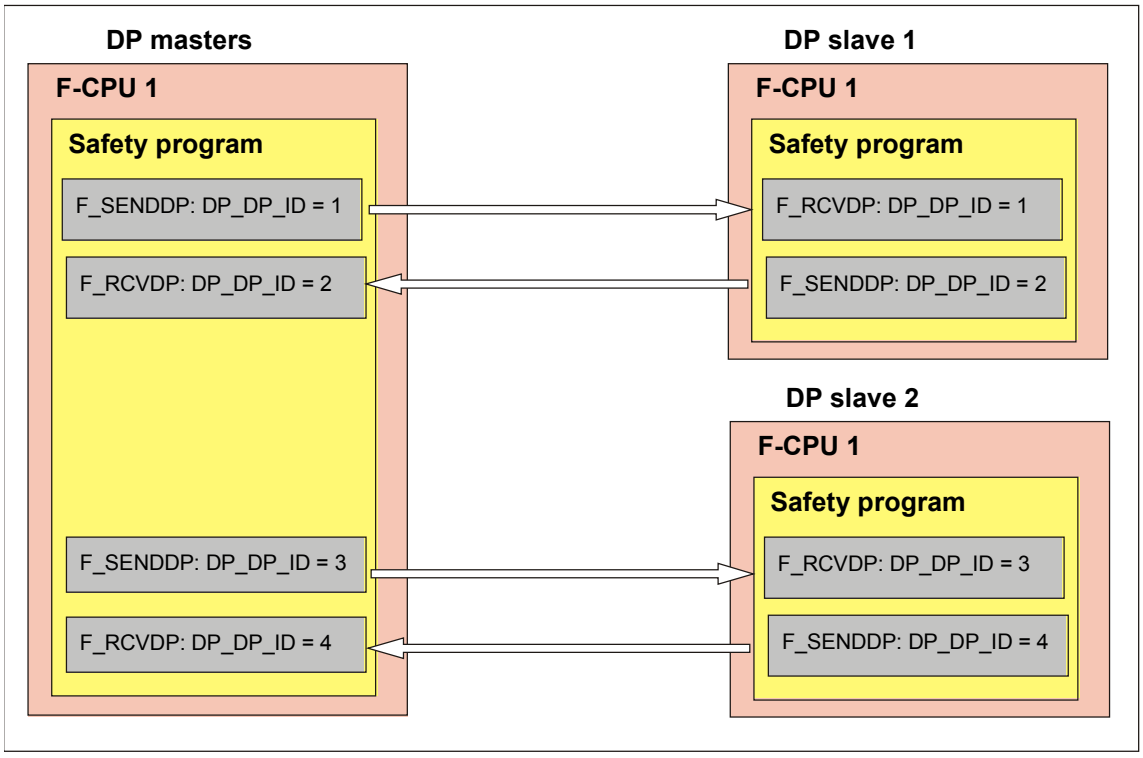
- Configure the input and output data areas for the DP master/intelligent DP slave in *HW Config* (see Section *1.1.1*)
- Assign start addresses of the input and output data areas of the DP master/ intelligent DP slave at input parameter LADDR and at F_SENDDP and F_RCVDP

You must carry out this assignment for every communication connection for each of the F-CPUs involved.

Programming Procedure

The procedure for programming safety-related master-slave communication is the same as for programming safety-related communication by means of a DP/DP coupler (master-master communication). Refer to Section *5.4* of the manual.

The example from the manual for specifying address associations at the inputs of the F-application blocks F_SENDDP and F_RCVDP has been adapted for the master-slave communication in the figure below.



Safety Note

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

If the data quantities to be transferred exceed the capacity of the F_SENDDP/F_RCVDP block pair, a second (or third) F_SENDDP/F_RCVDP block pair can be used. This requires an additional communication connection to be configured. Note that the maximum amount of input data and output data is 244 bytes, respectively; a maximum of 256 bytes in total can be used for transmission between a DP master and a DP slave.

Note

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

User Acknowledgment

To implement a user acknowledgment in the safety program of the F-CPU of an intelligent DP slave, refer to the description in Section 1.2.5.

1.2.7 F-Shared DB

F-Shared DB

In addition to the items specified in Section *5.5* of the manual, you can read out the following in the standard user program or by means of an operator control and monitoring system:

- Compile data of the safety program (variable "F_PROG_DAT", data type DATE_AND_TIME)

Note

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

If you used *S7 Distributed Safety* V 5.1 previously and read out the variable "F_PROG_SIG" in the safety program or by means of an operator control and monitoring system, and you are now converting to *S7 Distributed Safety* V 5.2, you must change the data type to DWORD for evaluation, if necessary.

The following can be read out in the safety program in the F-shared-DB:

- Constants 0 and 1 (variables "VKE0" and "VKE1", data type BOOL)

These variables are accessed with fully qualified access (for example, "F_GLOBDB".VKE0). 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.

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

"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. 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 manual, Section *5.8.3*).

1.2.9 Distributed Safety Library (V1)

Introduction

This section supplements Section 5.7 of the manual and describes the changes to the *Distributed Safety* library (V1) in *S7 Distributed Safety* V 5.2.

1.2.9.1 Changes

Changing F-Application Block Numbers

Note

Contrary to what is stated in the manual, you are permitted to change the F-application block numbers.

Note, too, that the symbolic name of an F-application block in the symbol table must match the name in the object properties of the block (header).

F_SENDDP and F_RCVDP

The following information supplements Section *5.7.3.10* of the manual.

The F-application blocks F_SENDDP and F_RCVDP are used as follows:

- For safety-related communication between DP masters (by means of a DP/DP coupler)
- For safety-related communication between the DP master and intelligent DP slaves

The information in Section *5.7.3.10* of the manual also applies. In addition, note the modified description for the input LADDR given below. This description applies to F_SENDDP and F_RCVDP:

	Parameter	Data Type	Description	Default
Input:	LADDR	INT	Start address of the output and input data area of the following: • DP/DP coupler for master-master communication • Respective address area for master-slave communication	0

1.2.9.2 F_SCA_I: Scaling Values of Data Type INT

Connectors

	Parameter	Data Type	Description	Default
Inputs	IN	INT	Input value to be scaled in physical units	0
	HI_LIM	INT	Upper limit value in physical units	0
	LO_LIM	INT	Lower limit value in physical units	0
Outputs	OUT	INT	Result of scaling	0
	OUT_HI	BOOL	1 = input value > 27648: OUT = HI_LIM	0
	OUT_LO	BOOL	1 = input value < 0: OUT = LO_LIM	0

Mode 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 27648. The scaling result is provided at output OUT.

The F-application block operates 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 27648, the 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, the output OUT is linked to LO_LIM, and OUT_LO is set to 1.

For reverse scaling, you must assign LO_LIM > HI_LIM. With reverse scaling, the output value at output OUT decreases while the input value at input IN increases.

Performance in the Event of Overflow or Underflow of Analog Values and Fail-Safe Value Output

Note

If inputs from the PII of an SM 336; AI 6 x 13 bit are used as input values, you must bear in mind that the F-system detects an overflow or underflow of a channel of this F-SM as an F-I/O fault or channel fault. The fail-safe value 0 is provided in place of $7FFF_H$ (for overflow) or 8000_H (for underflow) in the PII for the safety program.

If other fail-safe values are to be output in this case, you must evaluate the QBAD variable in the F-I/O DB (branch to output of an individual fail-safe value).

If the value in the PII of the F-SM is within the overrange or underrange, but is greater than 27648 or less than 0, you can likewise branch to the output of an individual fail-safe value by evaluating the outputs OUT_HI or OUT_LO.

1.2.10 Compiling the Safety Programm

Note

Before compiling the safety programs, close the *applications LAD/STL/FDB Editor*, *Display S7 Reference Data* and *Check Block Consitency*.

1.2.11 Complete Function Test of Safety Program or Protection through Program Identification

Introduction

The following two sections replace the corresponding sections in the *Section 5.8.5* of the manual.

1.2.11.1 Transferring the Safety Program to the F-CPU with a Programming Device/PC

CPU 416F-2 with Flash Card; CPU 315F-2 DP and IM151-F/CPU

The following safety notes apply when the safety program is transferred from a programming device/PC to:

- CPU 416F-2 **with Flash Card**
- CPU 315F-2 DP
- IM151-F/CPU

Safety Note

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/PC** to ensure that the F-CPU does not contain an "old" safety program:

1. For CPU 315F-2 DP and IM151-F/CPU: Download the safety program to the F-CPU in the "Safety Program" dialog box.
   For CPU 416F-2 with Flash-Card: Download the safety program to the F-CPU in the "Save to Memory Card" dialog box.
2. Perform a program identification (that is, check to determine whether the collective signatures of all F-blocks with an F-attribute in the block container match online and offline; refer to Sections *5.8.4* and *5.10.2* of the manual).
3. Perform a general reset of the F-CPU using the **mode selector** or by means of the **programming device/PC.** The safety program is thereby transferred again from the load memory (memory card, MMC for CPU 315F-2 DP and IM151-F/CPU, or Flash Card for CPU 416F-2) to the RAM once the RAM is reset.

Safety Note

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 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:

- A point-to-point connection must be used the first time a password is assigned to an F-CPU (this also applies 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.

CPU 416F-2 without a Flash Card

The following safety notes apply when the safety program is transferred from a programming device/PC to:

- CPU 416F-2 **without Flash Card**

Safety Note

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/PC** to ensure that the F-CPU does not contain an "old" safety program:

1. Perform a general reset of the F-CPU using the **mode selector** or by means of the **programming device/PC.**
2. Download the configuration in *HW Config* to the F-CPU.
3. Download the safety program to the F-CPU in the "Safety Program" dialog box.
4. Perform a program identification (that is, check to determine whether the collective signatures of all F-blocks with an F-attribute in the block container match online and offline; refer to Sections *5.8.4* and *5.10.2* of the manual).

Safety Note

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 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:

- A point-to-point connection must be used the first time a password is assigned to an F-CPU (this also applies 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.

1.2.11.2 Transferring the Safety Program to the F-CPU Using a Memory Card

Use of MMC or Flash Card

The following safety note applies to use of the following:

- Flash Card for CPU 416F-2
- MMC for CPU 315F-2 DP
- MMC for IM151-F/CPU

Safety Note

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 using a memory card** (MMC of Flash Card) to ensure that the F-CPU does not contain an "old" safety program:

1. Switch off the F-CPU, and remove the battery, if present, in the case of CPU 416F-2.
2. Remove the memory card (MMC or Flash Card) containing the old safety program from the F-CPU.
3. Insert the memory card (MMC or Flash Card) containing the new safety program in the F-CPU.
4. Switch on the F-CPU again, and if applicable, place the battery back in the CPU 416F-2.

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 use the following procedure:

1. Download the safety program to the memory card (MMC or Flash Card).
2. Please execute a program identification. This means that you should check if the collective signatures of all F-blocks with the F-attribute of the block container offline match those on the Memory Card (MMC or Flash Card). (See manual, Chapter *5.10.2*).
3. Label the Memory Card (MMC or Flash Card) unambiguously.

The procedure outlined must be ensured through organizational measures.

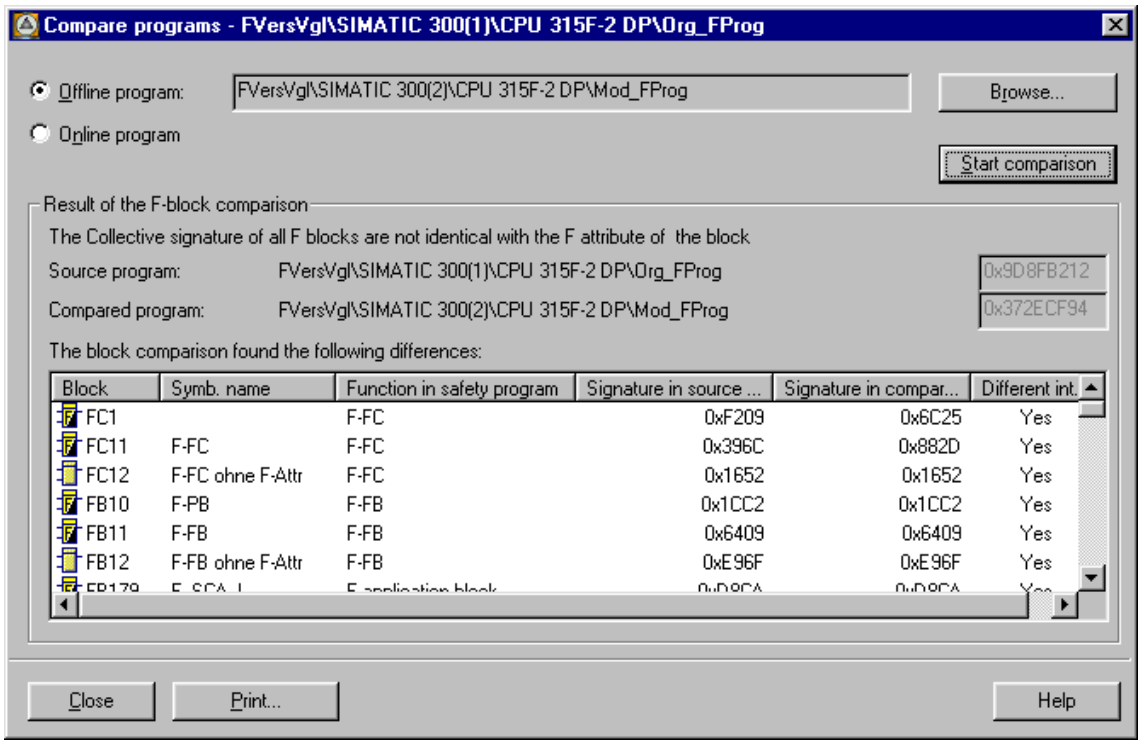
1.2.12 Comparing Safety Programs

"Compare Program" Dialog Box

The following information supplements Section *5.10.2* of the manual.

The "Compare Program" dialog box has been expanded to include the following two columns:

- "Function in the Safety Program"
- "Interface Different"



The "Function in the Safety Program" column displays the function of the F-blocks in the safety program.

The **"Interface Different"** column displays whether or not changes have resulted in the declaration table of F-blocks.

1.2.13 Printing Out Project Data of the Safety Program

Printout of Additional Project Data with *S7 Distributed Safety* V 5.2

In addition to the project data indicated in Section *5.11* of the manual, the following project data are printed out for the safety program:

- The version of the F-compiler that was used to compile the safety program is printed out as an internal version ID.
- For F_GLOBDB: absolute and symbolic address of the time of compilation and of VKE 0 and VKE 1.
- The time of compilation of the safety program is printed out.
- The version of the F-compiler used to create the printout is indicated in the footer.
- A note is printed out if the amount of local data reserved for the safety program has been exceeded.

2 Corrections to the *S7 Distributed Safety, Configuring and Programming* Manual, A5E00109537-01, Edition 03/2002

Introduction

This section presents corrections to the above-indicated edition of the manual that could not be made prior to publication. The corrections are associated with the corresponding sections of the manual.

The corrections apply to versions V 5.1 and V 5.2 of the optional package *S7 Distributed Safety*.

Use of Software Packages with Standard User Program

For software packages that can be used in parallel for the standard program and safety program (for example, SW Redundancy), general conditions may apply that must be observed:

Note

If the safety program assigns 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 when the software package is subsequently used. Changes made to the safety program must undergo a new acceptance test (see manual, Section *6.3*).

Section 3.3, 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.

Section 3.7, Configuring the safety-related CPU-CPU-Communication via DP/DP-Coupler (Master-Master-Communication)

Note

If you intend to pass **more than 2 bi-directional** or **more than 4 unidirectional** communication connections through a DP/DP coupler, then you must use:

- 4 universal modules for each F-CPU per bi-directional communication connection
- 2 universal modules for each F-CPU per unidirectional communication connection

In the object properties of the DP/DP coupler, select "Output" and "Input" individually as I/O type instead of "Input/Output". Enter the values for the output data area in the universal module with I/O type "Output" and the values for the input area in the universal module with I/O type "Input".

If you proceed as described in the manual, section *3.2*, you will be able to pass only a maximum of 2 bi-directional or 4 unidirectional communication connections through a DP-DP coupler.

Section 5.1.3, Supported Data and Parameter Types

Note

Block parameters of data type TIME of a called F-FB/F-FC must not be initialized with in/out parameters of the called F-FB/F-FC.
In/out parameters of data type TIME of a called F-FB/F-FC must not be interconnected with block parameters of a called F-FB/F-FC.

Section 5.3.2, F-I/O DB

The Description of the Variables Function IPAR_EN and IPAR_OK in the F-I/O DB has changed as follows:

	Variable	Data Type	Function	Default
Variables that Can or Must be Described	IPAR_EN	BOOL	Variable for reassigning fail-safe DP standard slave parameters	0
Variables that Can Be Evaluated:	IPAR_OK	BOOL	Variable for reassigning fail-safe DP standard slave parameters	0

IPAR_EN

The variable IPAR_EN corresponds to the variable iPar_EN_C in the PROFIsafe bus profile, V 1.2.

Refer to the PROFIsafe bus profile, V 1.2 and the documentation on fail-safe DP standard slaves to know when you have to set/reset these variables during a fail-safe DP standard slave parameter reassignment.

Safety Note

Please note that the affected F-I/O is **no** longer **passivated** as of *S7 Distributed Safety*, V 5.2 for IPAR_EN = 1.

If passivation should be continued for IPAR_EN = 1, you must set the variable PASS_ON = 1 in addition.

IPAR_OK

The variable IPAR_EN corresponds to the variable iPar_OK_S in the PROFIsafe bus profile, V 1.2.

Refer to the PROFIsafe bus profile, V 1.2 and the documentation on fail-safe DP standard slaves for how to evaluate these variables during a fail-safe DP standard slave parameter reassignment.

Section 5.3.4 to 5.3.7, Signal Chart Figures

The signal charts presented in the "Signal Chart ..." figures in Sections *5.3.4* to *5.3.7* of the manual 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:

- Type of F-I/O 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, or fail-safe DP standard slaves, version of PROFIsafe bus profile for the F-I/O)
- Cycle time of OB 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.

Section 5.3.8: No Longer Applicable

Section 5.4, Programming of Safety-Related Communication by Means of DP/DP Coupler

Note

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

Section 5.6, Safety Program Memory Requirement

You can roughly estimate the memory requirements of the safety program as follows:

Memory Requirement for Safety Program:

26 Kbytes for F system blocks F_CTRL_1, F_CTRL_2 and F_IO_BOI

+ 4.5 x memory requirement of all F-FB/F-FC/F-PB
+ 4.5 x memory requirement of all F application blocks used (except F_SENDDP and F_RCVDP)
+ Memory requirement of the F application blocks used F_SENDDP and F_RCVDP (4.4 Kbytes each)

Memory Requirement for Data:

5 x memory requirement of all F-DBs and I-DBs for F-PB/F-FB

+ 2.3 x memory requirement of all I-DBs for F application blocks (except F_SENDDP and F_RCVDP)
+ memory requirement of all I-DBs of the F application blocks used F_SENDDP and F_RCVDP
+ 0.7 Kbytes per F I/O (for F I/O DBs)

Block size of automatically generated F blocks

Note the following to stop the automatically generated F blocks from exceeding the maximum size possible in the respective F CPU:

- The size of an F-FB/F-FC/F-PB should not be more than a quarter of the maximum size of the FBs or FCs (see *technical specifications in the manual of the F-CPU used*).
- The following must apply for F-FBs (including F-PB):

  36

  + 2 x the number of all parameters or statistical data of data type BOOL

  + 4 x the number of all parameters or statistical data of data type INT

  + 6 x the number of all parameters or statistical data of data type TIME

  < maximum size of the data blocks in bytes
  (see *technical specifications in the manual of the F-CPU used*)
- The following must apply for F-DBs:

  36

  + 2 x the number of all F-DBs variables of data type BOOL

  + 4 x the number of all F-DBs variables of data type INT

  + 6 x the number of all F-DBs variables of data type TIME

  < maximum size of the data blocks in bytes
  (see *technical specifications in the manual of the F-CPU used*))

If you receive the message "Block x could not be copied" while downloading your safety program to the F-CPU, check whether conditions are fulfilled and if necessary:

- The size of the F-FB/F-FC/F-PB and
- The number of the parameters and the F-FBs/F-PBs statistical data and
- The number of the F-DBs variables.

Section 5.6.1, Use of Storage Elements in an F-FC

Note

For edge memory bits of Scan Edge (N, P) or Scan Signal Edge (NEG, POS) operations and addresses of Flip-Flop (SR, RS) operations, formal parameters of an F-FC may only be used if the following are assigned to these parameters as current addresses when the F-FC is called:

- F-DB data
- Instance DB data of an F-FB
- A parameter or static data of an F-FB calling the F-FC

Section 5.6.3, Inserting Application Templates

Prior to inserting the application template F_ESTOP (FBD) or F_ESTOP (LAD), you must copy the F-application block F_TOF from the block container F-Application Blocks\Blocks of Distributed Safety Library (V1) to the block container of your S7 program, if it is not already present.

Kap. 5.7.3, F-Application Blocks

Safety Note

You have to use a separate instance DB every time you call one of the following F application blocks:

F_TP, F_TON, F_TOF, F_ACK_OP, F_2HAND or F_MUTING

Every call of the listed F application blocks may only be edited in an F run group cycle once.

Section 5.7.3.9, F_MUTING

Safety Note

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

Only F-I/Os which execute the wire break monitoring within the max. reaction time specified in the technical specification for the F-I/O are suitable (for example SM 326; 10 x DC 24V/2A). The F module PM-E F DC24V PROFIsafe and 4 F-DO DC24V/2A PROFIsafe are not suitable.

Section 5.9.2, Permanent Modification Remains Active in the F-CPU

In addition to the cases specified in Section *5.9.2* of the manual, permanent modification remains active when you close a variable table with a permanent Modify request. Even though you are prompted for confirmation, the application is always closed with active Modify requests.

Remedy: Do not completely close the "Variable Tables" application. Instead, always close the active window first within the application, or install a *STEP 7* Version V 5.1 + Service Pack 3 + Hot Fix 1 or higher.

Section 5.9.2, More rules for testing the safety programs

In addition to the information in the manual, section *5.9.2,* setting breaking points in the standard user program could cause the following faults in the safety program:

- internal CPU fault

Section 6.3, Display of Incorrect Signatures for a Safety Program on an MMC in the Programming Device

This problem can be identified, for example, by the presence of signatures of F-blocks that do not match those in Annex 1.

Remedy: Install a *STEP 7* Version V 5.1 + Service Pack 3 + Hot Fix 1 or higher, or insert the MMC in an F-CPU and use the online view in the Safety Program dialog box to check the signatures.