SIEMENS

SIMATIC
Fail-safe function blocks for storage and retrieval machines

Distributed Safety/ TIA Safety Advanced

Manual

06/2014 Edition
Version v1.0

Safety instructions

This manual contains information which you should observe in order to ensure your own personal safety, as well as to avoid material damage. These notices are highlighted in the manual by a warning triangle and marked as follows according to the level of danger:

Safety notes and instructions
is important information, which is of significance for the acceptance and the safety-related use of the product.

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

Caution
indicates that minor personal injury or property damage may result if proper precautions are not taken.

Note
is important information about the product, the way to handle the product or the respective part of the documentation and we wish to especially bring this to your notice.

Qualified personnel

Commissioning and operation of a device are to be carried out by qualified personnel only. Qualified personnel under the terms of the safety instructions contained in this manual are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

Proper and intended use

Please observe the following:

Warning
The product may only be used for the applications specified in the catalog and technical description, and only in conjunction with third-party equipment and components if these have been specifically recommended or approved by Siemens.

This product can function correctly and reliably only if it is transported, stored, assembled, and installed correctly, and operated and maintained as recommended.

Trademarks

SIMATIC® is a registered trademark of Siemens AG.
Other designations used in this document may be registered trademarks; the owner's rights may be violated if they are used by third parties for their own purposes.

Siemens AG
Industry Sector
D-Erlangen

Warranty and liability regarding the application examples

The application examples of the software library „RBG_Failsafe_DS_V5_4“ or „RBG_Failsafe_TS_V13“ (following “application example”) of Siemens AG are a free of charge service for our customers. The application examples are to be understood as non-binding and they could be incomplete concerning configuration and equipment. The application examples shall not constitute customized solutions, but rather simply provide support for typical questions. Every customer is responsible for appropriate conditions and operations of the products within the valid regulations and has to control the results of the application examples and adapt them individually to its system.

Any claims against us – based on whatever legal reason – resulting from the use of the examples, information, programs, engineering and performance data etc., described in this Application Example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act (“Produkthaftungsgesetz”), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract (“wesentliche Vertragspflichten”). The damages for a breach of a substantial contractual obligation are, however, limited to the foreseeable damage, typical for the type of contract, except in the event of intent or gross negligence or injury to life, body or health. The above provisions do not imply a change of the burden of proof to your detriment.

When using these Application Examples, you recognize that we cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these Application Examples at any time without prior notice. If there are any deviations between the recommendations provided in these application examples and other Siemens publications – e.g. Catalogs – the contents of the other documents have priority.

We do not accept any liability for the information contained in this document.

Transfer and reproduction of these application examples or extracts of them are forbidden if not explicit allowed by the Siemens AG.

Security information

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

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

To stay informed about product updates as they occur, sign up for a product-specific newsletter. For more information, visit http://support.automation.siemens.com.

Table of contents

List of diagrams

List of tables

1 Storage and retrieval machines and safety functions

This chapter schematically provides an overview of the application conditions of fail-safe function blocks for storage and retrieval machines and the mechanical hardware versions that are supported.

1.1 General design for safety-related position sensing

The following components are essentially required for use in the fail-safety function blocks for storage and retrieval machines known in the following as "ASRS blocks".

- Fail-safe SIMATIC S7 control – Distributed Safety/Safety Advanced
- SINAMICS S120 inverters with CU320-2 (from firmware release 4.6), called SINAMICS S120 in the following, equipped with an encoder
  - SMC20/SMC30
  - or via DRIVE-CLiQ.
- PROFIBUS/PROFINET data transfer between SINAMICS and F-CPU
- F-DO module to control the brakes
- F-AI module or comparable safety-related signal source for load measurement for overload/slack cable detection
- External mechanical brake and/or motor holding brake

An example of the hardware structure looks like this:

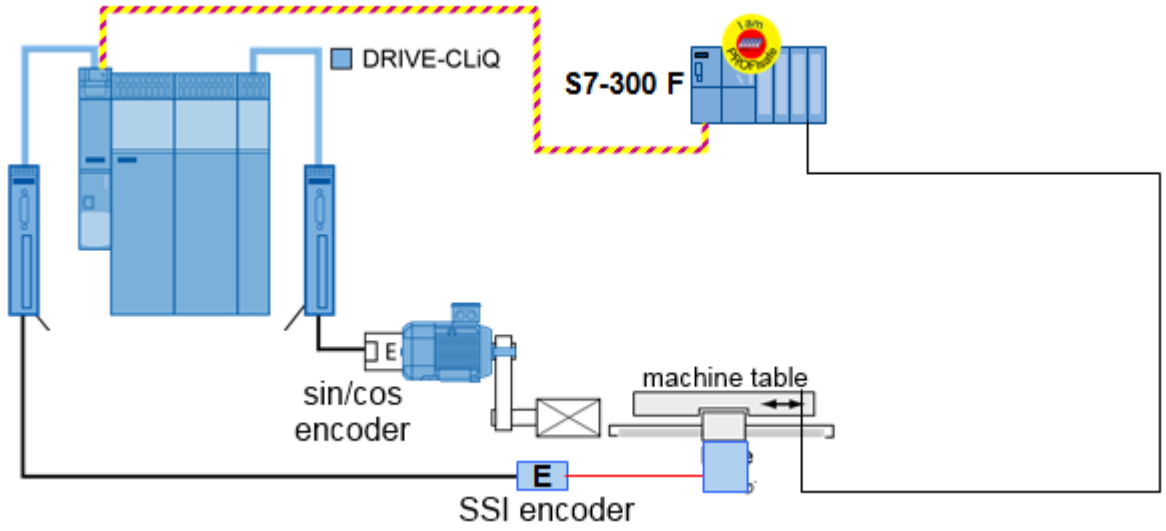


Figure 1: Hardware structure

The package of blocks addresses several versions of encoder combinations, also see Table 1. As a consequence, the following components may be additionally required:

1.2 Supported encoder combinations and mechanical design versions

An overview of the encoder combinations supported by the ASRS blocks is provided in the following.

1.2.1 Safety-related motor encoder with positively-driven mechanical system

Sensing:

- Safety-related sin/cos motor encoders with safety-related connection via PROFIsafe telegram 901 of SINAMICS S120.
- The absolute position is transferred to the F-CPU from the SINAMICS S120 using a standard telegram.

The position actual value of EPOS must be additionally transferred via the standard telegram to obtain the two-channel structure for data transfer to the F-CPU. The position actual value of the motor encoder is additionally determined by SI and is transferred to the F-CPU using a safety-related telegram. The motor encoder must be a safety encoder (safety-related motor encoder with safety-related mounting).
As a consequence, the signal flow of the safety function looks like this:

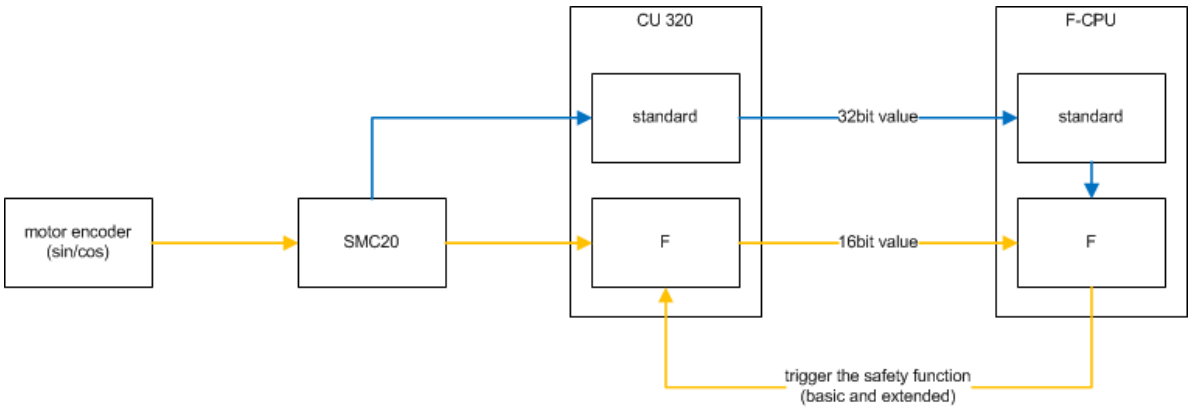


Figure 2: Single-encoder system

1.2.2 Two-encoder system with connection via SINAMICS S120

Sensing:

- The sin/cos motor encoder is connected with the SINAMICS S120 via an SMC20 or a DRIVE CLiQ interface (SMI), the direct measuring system (SSI) via an SMC30. The position control of the EPOS is realized via the direct measuring system.

The position actual value of EPOS (basic positioner, drive function) must be transferred via the standard telegram to obtain the two-channel structure for data transfer to the F-CPU. The position actual value of the motor encoder is determined by SI and is transferred to the F-CPU using a safety-related telegram. The motor encoder must be a safety-related encoder, suitable for safety applications. Safety-related mounting is not required, as possible errors are monitored using the cross comparison with the second encoder.

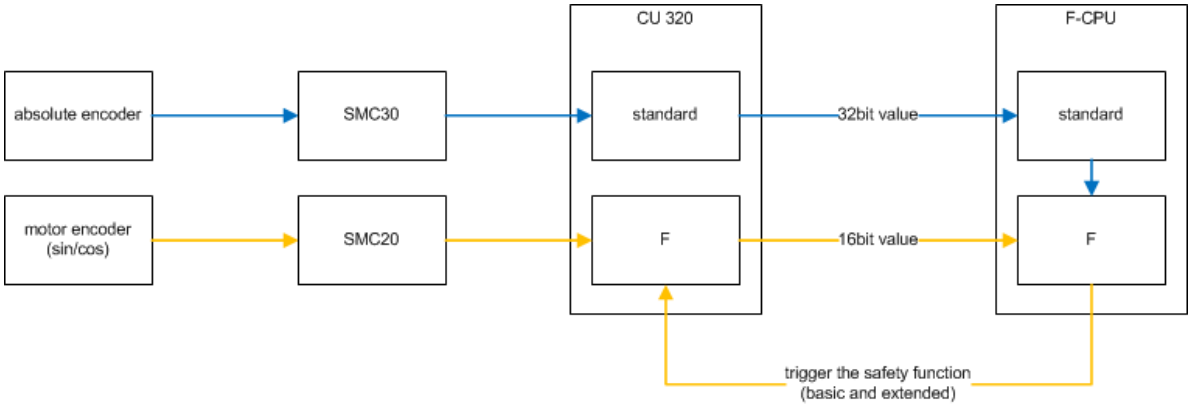


Figure 3: Two-encoder system, version 1

1.2.3 Two-encoder system with connection via distributed I/O

Sensing:

a) Sin/cos motor encoder via PROFIsafe telegram from SINAMICS S120, direct measuring system using a standard telegram of the SSI module (e.g. SM338) to the F-CPU.

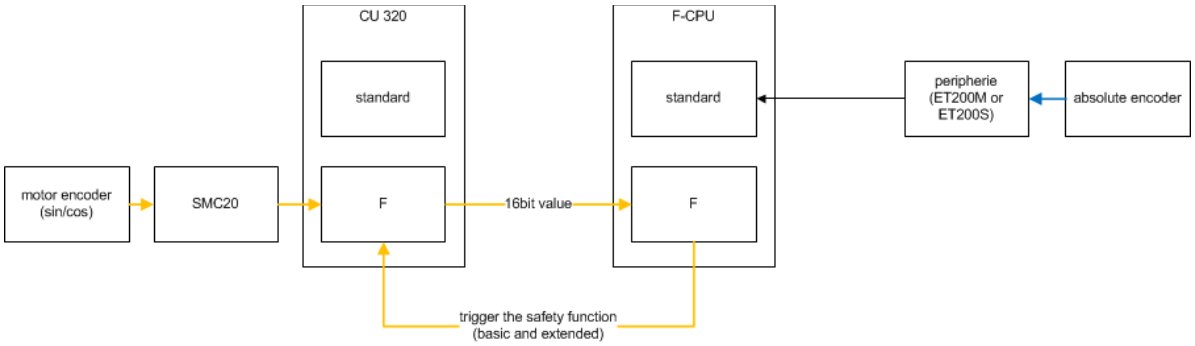


Figure 4: Two-encoder system, version 2

b) Sin/cos motor encoder via PROFIsafe telegram from SINAMICS S120, direct encoder using a standard telegram PROFIBUS/PROFINET-capable encoder.

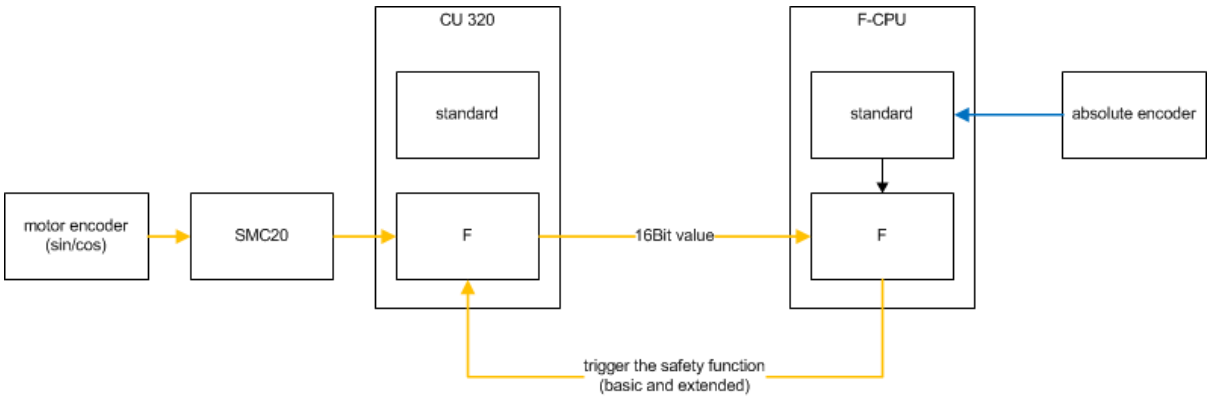


Figure 5: Two-encoder system, version 3

1.2.4 Three-encoder system

Sensing:

a) Sin/cos motor encoder via PROFIsafe telegram from SINAMICS S120. Two direct measuring systems using a standard telegram:
   - Position 1 via SINAMICS S120
   - Position 2 from distributed I/O with secure communication via an F module.

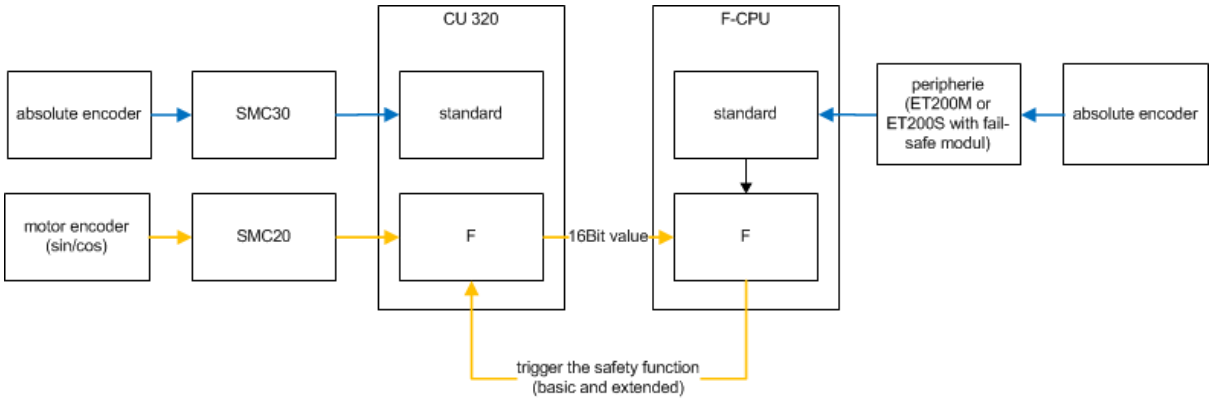


Figure 6: Three encoder system, version 1

b) Sin/cos motor encoder via PROFIsafe telegram from SINAMICS S120. Two direct measuring systems using a standard telegram:
   - Positions 1 and 2 via distributed I/O. One channel with secure communication via F module.

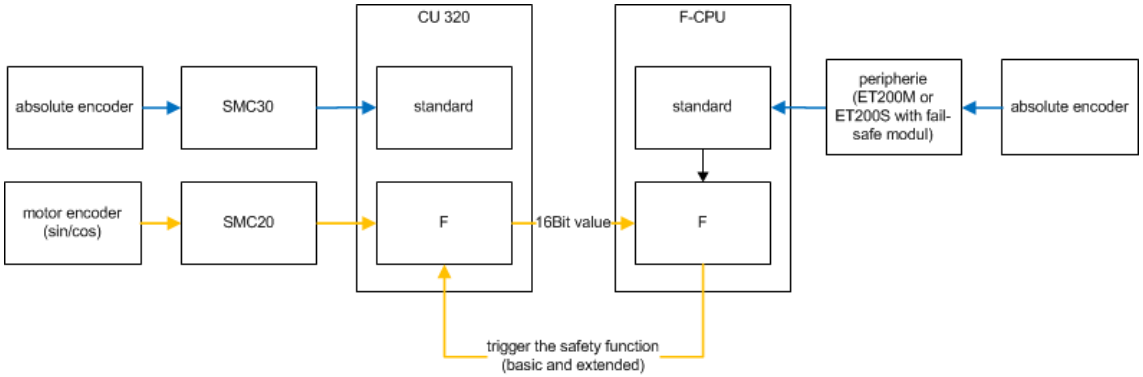


Figure 7: Three encoder system, version 2

Three-encoder systems are used if a high degree of slip is to be expected, and as a result the motor encoder cannot be used to check the plausibility of the position. Instead, the plausibility of the position is checked by making a cross comparison between the two direct measuring systems.

Safety notes and instructions
To identify a bus driver that has "frozen up", i.e. there is no longer any communication between the measuring system and CPU, there must be at least one fail-

safe module in a channel in the station, via which the direct measuring system is read-in. If the communication between the encoder and CPU is inadmissibly slow, or completely fails, then the F module involved signals a communication error. This is then evaluated in the safety program and must be used to initiate a stop response.

Safety notes and instructions
Both direct measuring systems must be installed, opposing one another to achieve the specified diagnostics coverage.

1.2.5 Summary of the encoder versions

The encoder combinations possible in principle and their ability to be used are summarized in the following table. POS1, POS2 as well as POS_SI take reference to the interconnection at the "F_SAFE_POS" block described in more detail under Chapter 3.2. Refer below for the legend.

A motor encoder (MSSI or MNSI) is always require to sense the safety-related position and velocity; the encoder signals are acquired via the SI part of the drive.

The following encoder combinations should be provided depending on the particular application scenario:

	POS1	POS2	POS_SI
Version 1: Safety-related motor encoder with positively-driven mechanical system	LM-ST-NS	-	MNSI[1] / MSSI
Version 2: Two-encoder system: Connected via SINAMICS S120.	LM-SMx-NS	-	MNSI / MSSI
Version 3: Two-encoder system: Connected via the distributed I/O.	LD-DP-NS	-	MNSI / MSSI
Version 4 a): Three-encoder system. Position encoder via SINAMCS 120 and distributed PLC I/O. Secure communication via F module.	LM-SMx-NS[2]	LD-DP-KS[2]	MNSI / MSSI
Version 4) b): Three-encoder system. Position encoder via PLC I/O. Secure communication via F module.	LD-DP-NS[2]	LD-DP-KS[2]	MNSI / MSSI

Table 1: Overview of possible encoder combinations

[1] Not permissible with single-channel encoder connection to the motor shaft; permissible for two independent encoders, e.g. for a double axis drive. LM-ST-NS via motor encoder 1 and MNSI via motor encoder 2.
[2] Overwriting a process image must be detected using counter-rotating encoders

Legend:
MSSI: Motor encoder, safe connection via SI F telegram
MNSI: Motor encoder, no safe connection via SI F telegram
LM-ST-NS: Position actual value from the motor encoder via a standard 32-bit telegram (from Epos), not safety related
LD-SMx-NS: Epos position actual value from a direct measuring system via SMC/SMI via 32-bit standard telegram, not safety-related
LD-DP-NS: Position actual value from a direct measuring system via distributed I/O, no safety-related communication (e.g. PROFINET encoder, SM338).
LD-DP-KS: Position actual value from a direct measuring system via distributed I/O, safety-related communication using an F module connected to the backplane bus.

1.2.6 Safety-related parameters of the encoder versions

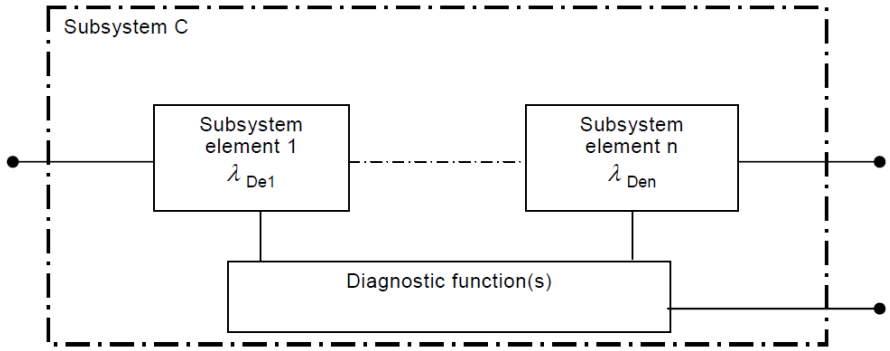
Chapter 1.2.5 lists the various encoder versions. Which of these versions is used depends on the particular customer application; for the safety-related use of the subsequently descried software solution, it is mandatory that one of the described versions is used.

As a result of the different encoder versions, and the resulting wide range of hardware versions that can be used, the user must determine the safety integrity level of the safety function. To be in compliance with EN 528 (RN15, 2008), this must be at least SIL2/PL_r d over the complete safety function (sense -> evaluate -> respond).

To verify this, in this section, parameters are described that have a direct influence on the software solution when calculating the safety integrity level based on DIN EN 62061 (RN01, 2005). Only the sense block is discussed. The evaluate block corresponds to a SIMATIC F-CPU with Distributed Safety/Safety Advanced, certified up to SIL3/PL e; the respond block is a SINAMICS S120, certified up to SIL2/PL d. The precise parameters of the evaluate and respond blocks should be taken from the corresponding data sheets.

While versions 2-4 comply with the requirements according to subsystem D, version 1 fulfills the requirements according to subsystem C, as shown below:

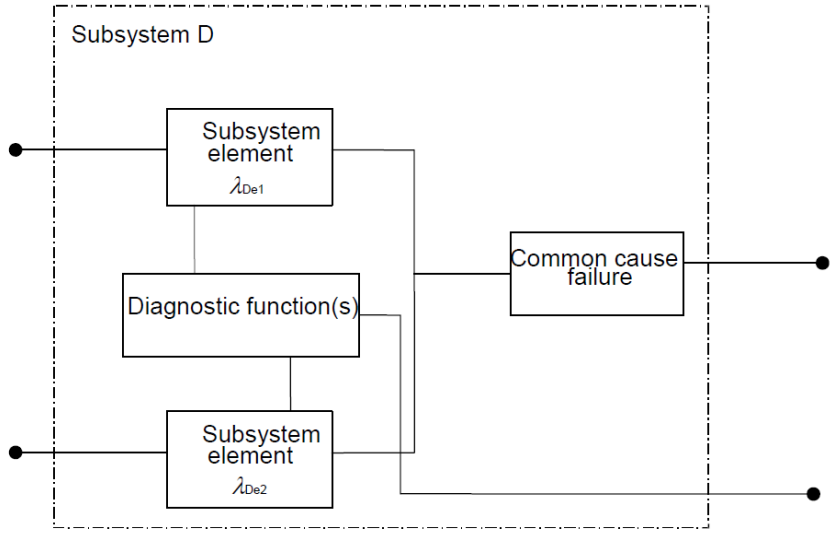
1.2.6.1 Subsystem C according to DIN EN 62061 (Chapter 6.7.8.2.4):



$$\lambda_{DssC} = \lambda_{De1} + (1 - DC_1) + \ldots + \lambda_{Den}(1 - DC_n)$$

$$PFH_{DssC} = \lambda_{DssC} \times 1\,h$$

1.2.6.2 Subsystem D according to DIN EN 62061 (Chapter 6.7.8.2.5):



$$\lambda_{DssD} = (1-\beta)^2 \left\{ \left[\lambda_{De1} \times \lambda_{De2} \times (DC_1 + DC_2) \right] \times T_2/2 + \left[\lambda_{De1} \times \lambda_{De2} \times (2 - DC_1 - DC_2) \right] \times T_1/2 \right\} + \beta \times (\lambda_{De1} \times \lambda_{De2})/2$$

$$PFH_{DssD} = \lambda_{DssD} \times 1h$$

	Subsystem (SFF/HFT)	SIL CL limit	λ_{Ds1}	λ_{Ds2}	DC_1	DC_2	β	T1	T2
Version 1: Safety-related motor encoder with positively-driven mechanical system	C ((≥0.99[1]/ 0)	Internal safety functions of the SINAMICS S120, certified according to SIL2/PLd							
Version 2: Two-encoder system: Connected via SINAMICS S120.	D (≥0.99[1]/ 1[2])	3[3]	Dependent on the hardware	Dependent on the hardware	99% with the diagnostics implemented in the ASRS block library	Corresponding to DC1	0.02 acc. to Table 3: Assessing common cause faults according to DIN EN 62061 Annex F.1	Dependent on the hardware	Corresponding to the call interval of the safety program
Version 3: Two-encoder system: Connected via the distributed I/O.									
Version 4 a): Three-encoder system. Position encoder via SINAMCS 120 and distributed PLC I/O. Secure communication via F module.									
Version 4) b): Three-encoder system. Position encoder via PLC I/O. Secure communication via F module.									

Table 2: Parameters according to DIN EN 62061

Comments regarding Table 2: Parameters according to DIN EN 62061:

1) As a result of the diagnostics implemented in the ASRS block library, all potentially dangerous faults are identified by the diagnostics, it follows: $\lambda_{DU} \rightarrow 0$. From the calculation of SFF ($SFF = \frac{\sum \lambda_S + \sum \lambda_{DD}}{\sum \lambda_S + \sum \lambda_{DD} + \sum \lambda_{DU}}$), SFF ≥ 0.99 is immediately obtained.
2) The failure of a subsystem element does not result in the loss of the SRCF, as this fault would be discovered by a comparison value – as well as by a plausibility check (as a result of the redundant design). This directly means that HFT = 1.
3) According to DIN EN 62061 Table 5, for HFT = 1 and SFF = 1, an SIL CL limit of 3.

The evaluation of common cause faults according to DIN EN 62061 Annex F.1, is shown in the following table. In some instances, the measures applied against common cause faults are implemented in the ASRS block library; in other instances, the appropriate measures must be taken by users. The measures that users must always take (mandatory) are appropriately listed in the following table. If additional measures are to be taken, which are shown in gray in the table, then these can improve the CCF factor or the β value, on the other hand measures that are not taken, reduce the CCF factor or β value.

Property	Reference	Points	Reason
Separation/isolation			
Are SRECS signal cables for the individual channels separated from other channels at all locations – or are they adequately protected?	1a	**5**	Users must appropriately install the sensors
Is the detection of signal transfer errors adequate when using information coding/decoding?	1b	**10**	Provided by the solution implemented in the ASRS block library
Are SRECS signal cables and electric power supply cables separated at all locations – or adequately protected?	2	5	
Are subsystem elements provided as physically separate units in local housings if they can contribute to a CCF?	3	**5**	Users must appropriately install the sensors
Diversity/redundancy			
Are various electronic technologies used in the subsystem, for example in some instances electronics or programmable electronics, and in other instances, an electromechanical relay?	4	8	
Are elements used in the subsystem that employ different physical principles (e.g. sensing elements at a protective door that employee mechanical and magnetic sensing techniques)?	5	10	
Are elements used in the subsystem that have different time responses with reference to functional operation and/or failure types?	6	**10**	Provided by the solution implemented in the ASRS block library
Do the subsystem elements have a diagnostics test interval of ≤ 1 min?	7	**10**	Provided by the solution implemented in the ASRS block library. **Note:** The safety program must be called at intervals less than 1 min!
Complexity/design/application			
Are cross connections between subsystem channels prevented, with the exception of cross connections that are used for diagnostics?	8	**2**	Provided by the solution implemented in the ASRS block library
Assessment/analysis			
Have the results of failure types and effect analyses been evaluated in order to define sources of failures as a consequence of a common fault. Further, have these types of sources – that have been previously determined – been eliminated as a result of the design?	9	**9**	Provided by the solution implemented in the ASRS block library
Are field failures analyzed and fed back into the design process?	10	9	
Competence/training			
Do the development engineers of the subsystems understand the reasons for and effects of failures originating from a common cause?	11	**4**	Requirements placed on users

Monitoring ambient conditions			
Is it probable that the subsystem elements always operate within the temperature, humidity, corrosion, dust and vibration range etc. in which they have been tested, without the ambient conditions being externally monitored?	12	**9**	Users must appropriately select the sensors for the application
Is the subsystem immune to negative electromagnetic influences up to and including the limits defined in Annex E?	13	**9**	Users must appropriately select the sensors for the application
Result		**73**	**According to DIN EN 62061 Table F.2: β = 0.02**

Table 3: Assessing common cause faults according to DIN EN 62061 Annex F.1

Total number of points	**Factor of the failures resulting from a common cause (β)**
< 35	10 % (0.1)
35 to 65	5 % (0.05)
65 to 85	2 % (0.02)
85 to 100	1 % (0.01)

Table 4: DIN EN 62061 Annex F.2

2 System and software requirements

2.1 General

The fail-safe function blocks for storage and retrieval machines described in the following chapters can be used in conjunction with the Siemens fail-safe automation system

- IM 151-7 F-CPU,
- IM 151-8F PN/DP CPU,
- IM 154-8F CPU,
- CPU 315F-2 DP,
- CPU 315F-2 PN/DP,
- CPU 317F-2DP,
- CPU 317TF-2DP,
- CPU 317F-2 PN/DP,
- CPU 319F-3 PN/DP,
- CPU 414F-3 PN/DP,
- CPU 416F-2,
- CPU 416F-3 PN/DP as well as
- WinAC RTX F.

To be able to use the fail-safe function block F_ENDZONE (Chapter 3.6), the Siemens automation system that is being used must permit data blocks that require over 40 kbyte load memory and 20 kByte work memory. This information is provided in the data sheet of the control system that you are using.

As introduction, the safety aspects when creating fail-safe function blocks are discussed before their properties are discussed in detail.

The fail-safe function blocks for storage and retrieval machines were developed to create individual subfunctions in order to guarantee that the blocks can be used in a modular fashion.

2.2 Safety requirements

The S7-IM 151-7F CPU, IM 154-8F CPU, S7-315F, S7-317F, S7-319F, S7-414F, S7-416F automation systems as well as WinAC RTX comply with the following safety requirements:

Safety Integrity Levels SIL1 to SIL3 according to
IEC 61508

2.3 Software

The following *Siemens SIMATIC* software must be installed on the PC/PG when using fail-safe functions for storage and retrieval machines:

- SIMATIC STEP 7 Professional V5.5 + SP3 or higher
- SIMATIC S7 Distributed Safety Programming V5.4 + SP5
- SIMATIC S7 F ConfigurationPack

and/or

- SIMATIC STEP 7 Professional V13 or higher
- SIMATIC STEP 7 Safety Advanced V13

and for parameterizing the drives

- SINAMICS MICROMASTER STARTER V4.3 SP3 or higher

The actual version as well as all predecessor versions of the SIMATIC S7 F ConfigurationPack can be downloaded at the following link:

http://support.automation.siemens.com/CH/view/de/15208817

The actual version as well as all predecessor versions of SINAMICS MICROMASTER STARTER can be downloaded at the following link:
http://support.automation.siemens.com/WW/view/de/26233208

2.4 Safety aspects when generating the blocks

The blocks for the safety-related control of storage and retrieval machines were generated using the certified fail-safe function blocks in F-FBD. The development tool compiler generates fail-safe blocks. These can then be loaded into libraries and called in any F- FBs and F- FCs.

Regarding the internal implementation and the software development process employed, the fail-safe function blocks for storage and retrieval machines are compliant with PLd/SIL2. However, it must also be verified that the function blocks from the ASRS block library used in the user software are in compliance with the relevant standards regarding their behavior and their principle of operation. Generally, this verification can be realized in the form of a function test.
Also due to restrictions regarding the hardware components that can be used – specifically inverters – the safety level that can be achieved using the fail-safe function blocks for storage and retrieval machines is restricted to PLd/SIL2.

For the verification of the safety-related parameters required for the hardware components, please refer to Chapter 1.2.6.

2.5 Standards that are complied with

The ASRS block library was developed in compliance with the following standards:

DIN EN 528 (see Chapter 2.5.1)
DIN EN ISO 13849-1
DIN EN ISO 13849-2
DIN EN 62061

2.5.1 Demarcation of DIN EN 528 to the ASRS block library

Requirements according to EN528:2008 Table C.2			Covered by the ASRS block library
Consecutive No..	Safety function	Section	
1	Function for monitoring access through doors	5.3.3 5.3.4 5.10.3.3 5.10.3.6	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
2	Stop function	5.3.7	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
3	Emergency Stop function	5.3.8 5.3.8.1 5.3.8.3	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
4	Function to stop hoisting motion at the end stop and for power failure	5.4.1.1 5.4.2 a), b), c)	Yes

Requirements according to EN528:2008 Table C.2			Covered by the ASRS block library
Con-secutive No..	Safety function	Section	
5	Function to stop travel motion at the end of a travel route (e.g. end of the aisle), when the power fails, for collisions, if more than one unit is traveling on the same rails.	5.5.1.1 5.5.1.2 5.5.3	Yes
6	Function with additional brake and velocity reduction, if the vehicle must travel around curves with a reduced velocity	5.5.1.2 a) 5.5.2	Yes
7	Function of the additional brake and velocity reduction, if the end buffers are not configured so that the unit can approach them with at least 70 % of the rated velocity	5.5.1.2 b) 5.5.2	Yes
8	Function to prevent the load and the load suspension device from colliding with the rack	5.4.6.6 5.6.2 5.6.3 5.6.5 5.6.7 5.6.8.2 5.10.7.1	Yes
9	Load suspension device – interlocks	5.6.5 a), b)	Yes
10	Load suspension device – Interlocks	5.6.5 c)	Yes
11	Load suspension device – rack compartment occupied	5.6.5 d)	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
12	Load suspension device – load position monitoring	5.6.7	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
13	Function of control devices for dangerous motion (manual open-loop control)	5.7.6	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
14	Interlocking function with imple-mentation devices	5.8.2 5.8.3 5.8.4	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
15	Function, which only permits a slow velocity when a person is located on the emergency control station	5.3.7 5.9.4	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
16	Function to stop the unit when it is necessary to gain access through emergency exits and covers	5.10.3.2 c) 5.10.3.2 e) 5.10.3.4 5.10.6.4	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)

Requirements according to EN528:2008 Table C.2			Covered by the ASRS block library
Con-secutive No..	Safety function	Section	
17	In the environment of the unit, securing against inadvertent motion of the load Rack compartment sensor	5.10.7.1 a)	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
18	In the environment of the unit, securing against inadvertent motion of the load Slide-through guard	5.10.7.1 b)	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)
19	In the environment of the unit, securing against inadvertent motion of the load Slide-through guard	5.10.7.1 c)	No User interconnection required (e.g. Distributed Safety/Safety Advanced library)

Additional sections of the EN528:2008 that are covered by the ASRS block library:

- 5.4.3.1 – overload protection
- 5.4.3.2 - protection against slack cable state

3 Fail-safe function blocks for storage and retrieval machines

3.1 Overview

3.1.1 Safety notes and instructions

As stated in Chapter 2.5, the fail-safe function blocks for storage and retrieval machines cover the requirements laid down in EN528:2008.

The safety integrity level of the various safety functions is only obtained through the correct interconnection. For this reason, the correct interconnection of each fail-safe function block in this library and the overall functionality of the safety functions must be validated against the relevant application-specific hardware and software based on positive and negative tests.

The tests should first be carried out in a part of the system where there is sufficient clearance to the fixed end stops (buffers). It must also be ensured that the system can be safely shutdown if an emergency occurs during the various tests.

The tests should be documented, for example, using trace recordings, so that limit value violations, shutdown conditions and stopping distances can be clearly identified. This therefore allows a clear statement to be made about the correct function of each individual safety function.

3.1.2 Fail-safe blocks

The library "RBG_Failsafe_DS_V5_4" for SIMATIC S7 Distributed Safety Programming or "RBG_Failsafe_TS_V13" for SIMATIC STEP 7 Safety Advanced includes the following blocks:

F_SAFE_POS	Function block to generate a safety-related position and velocity actual value
F_SLP_MONITOR	Function block for safe position monitoring
F_ENDZONE_DB	Function block to monitor the velocity at the end of the travel range
F_BRAKE_TEST	Function block to perform a safe brake test in conjunction with the *SBT* drive function
F_LOAD_MONITOR	Function block to detect overload and slack cable condition with the option of testing the measuring device
F_SBR_MONITOR	Function block to monitor the brake ramp
F_SCALE_DINT, SCALE_DINT	Blocks to safely scale 32-bit values
F_MIN_MAX	Function to select minimum/maximum value

The following fail-safe blocks are additionally required that are programmed using SIMATIC S7 Distributed Safety Programming:

F_TP	Generates a pulse with a specific duration
F_W_BO	Converts a value in the WORD format into 16 pieces of data in the bool data type

F_BO_W	Converts 16 pieces of data in the BOOL data type into a value in the WORD format
F_INT_RD	Reads an INT value data type indirectly from an F-DB

These blocks are contained in the "Distributed Safety(V1)/ F-Applikation Blocks" library.

The following fail-safe blocks are additionally required under SIMATIC STEP 7 Safety Advanced:

TP	Generates a pulse with a specific duration
W_BO	Converts a value in the WORD format into 16 pieces of data in the bool data type
BO_W	Converts 16 pieces of data in the BOOL data type into a value in the WORD format
RD_FDB	Reads a value indirectly from an F-DB

These blocks are located under Statements -> Simple statements.

Note
When generating under SIMATIC STEP 7 Safety Advanced, for the fill-safety function blocks "F_BRAKE_TEST", "F_SLP_MONITOR", "F_ENDZONE_DB16000", "F_ENDZONE_DB15999", "F_ENDZONE_DB15998", "F_ENDZONE_DB511", "F_ENDZONE_DB510" and "F_ENDZONE_DB509", the following warning messages are output:

"Network: 00,Initialization of local data '%L0.1 ()' is not realized before the first jump instruction. Initialize before the first jump instruction."

These warning messages involve additional compiler data, which have no relevance for users and have no impact on the safety integrity.

Note
The library blocks listed under SIMATIC STEP 7 Safety Advanced must be set to Version 1.0 before integrating the ASRS library. The elements of the system library that are used must be set to Version 1.0 in the safety administration. Otherwise, error messages can be generated when compiling the safety program.

3.1.3 Block connections

For fail-safe blocks, several issues regarding
block connections must be observed:

Note
Although connections EN and ENO appear in the LAD/FBD editor, they are neither evaluated nor supplied by the program code of the F block – further, they may neither be interconnected nor parameterized.

3.1.4 Block numbers and signatures

Block number	*Block name*	SIMATIC S7 Distributed Safety Programming		SIMATIC STEP 7 Safety Advanced	
		Block signature	Initial value signature	Block signature	Initial value signature
FB 200	F_SAFE_POS	980B	B06D	51BB	B06D
FB 201	F_SLP_MONITOR	8DE4	5896	E66B	5896
FB 203	F_BRAKE_TEST	610B	64F3	610B	64F3
FB 204	F_LOAD_MONITOR	2644	129F	2644	129F
FB 207	F_SBR_MONITOR	2935	A68E	2935	A68E
FB 208	F_SCALE_DINT	0797	8DF7	0797	8DF7
FB 210	F_ENDZONE_DB16000	5796	222F	0884	222F
FB 211	F_ENDZONE_DB15999	7CE3	222F	23F1	222F
FB 212	F_ENDZONE_DB15998	FB9B	222F	A489	222F
FB 213	F_ENDZONE_DB511	7AA8	222F	25BA	222F
FB 214	F_ENDZONE_DB510	FDD0	222F	A2C2	222F
FB 215	F_ENDZONE_DB509	7459	222F	2B4B	222F
FC 206	F_MIN_MAX	6A25	*None, as FC*	6A25	*None, as FC*
FC 208	SCALE_DINT	*None, as standard block*	*None, as standard block*	*None, as standard block*	*None, as standard block*

Figure 8: Block signatures

In the following chapter, unique FB/FC numbers are assigned for the blocks to be implemented. When required, these can be adapted to address the requirements of the specific machine; this means that the blocks can be freely renumbered and the blocks do not refer to one another.

3.1.5 Integration in cyclic interrupts - OBs

Safety notes and instructions
Blocks F_SAFE_POS, F_BRAKE_TEST, F_LOAD_MONITOR and F_SBR_MONITOR may only be integrated in an F-runtime group, which is called from a cyclic interrupt OB 3x. It is not permissible to integrate them in the OB 1. The cycle time of the cyclic interrupt OBs is parameterized in the HW Config (CPU parameter "Cyclic interrupts, execution").

Safety notes and instructions
The cycle time of the safety program is configured to specifically address the requirements derived from the risk assessment for the machine in which the blocks are used. The user is responsible for correctly performing the risk assessment and appropriately configuring the times.

Note
We recommend that the SCALE_DINT block is called in the same cyclic interrupt OB as the safety program, e.g. immediately before the F_CALL block.

3.1.6 Using data blocks/multi-instances

Note
All of the data blocks listed in this manual, with the exception of the DBs permanently assigned to F_ENDZONE_DBxxxxx, are used arbitrarily. The fail-safe function blocks of the "RBG_Failsafe_DS_V5_4"/"RBG_Failsafe_TS_V13" library can be combined with any data block still not assigned in the user program

Note
With the exception of F_SCALE_DINT, ASRS blocks can be called as multi-instance without any restrictions. It is mandatory that F_SCALE_DINT is called with its own instance data block.

3.1.7 Response times

The response times required must be taken from the relevant risk assessment. This involves a block library that can be universally used. This is the reason that a specific value cannot be specified for the response times of the individual safety functions.

Safety notes and instructions
Depending on the required response time, parameters T_SAMPLE (and therefore the call interval of the safety program) – as well as POS_SI_T_SAMPLE – must be parameterized in the ASRS blocks so that under no circumstances are they shorter than the maximum permitted response time. It should be taken into account that also the hardware components used influence the response time. The s7fcoti table can be used to calculate the response time from the sensor to the actuator.

3.1.8 Execution times

The values of the execution times of the fail-safe ASRS blocks running on the supported F-CPUs can be taken from the table in the Appendix I). These values are required to calculate the response time.

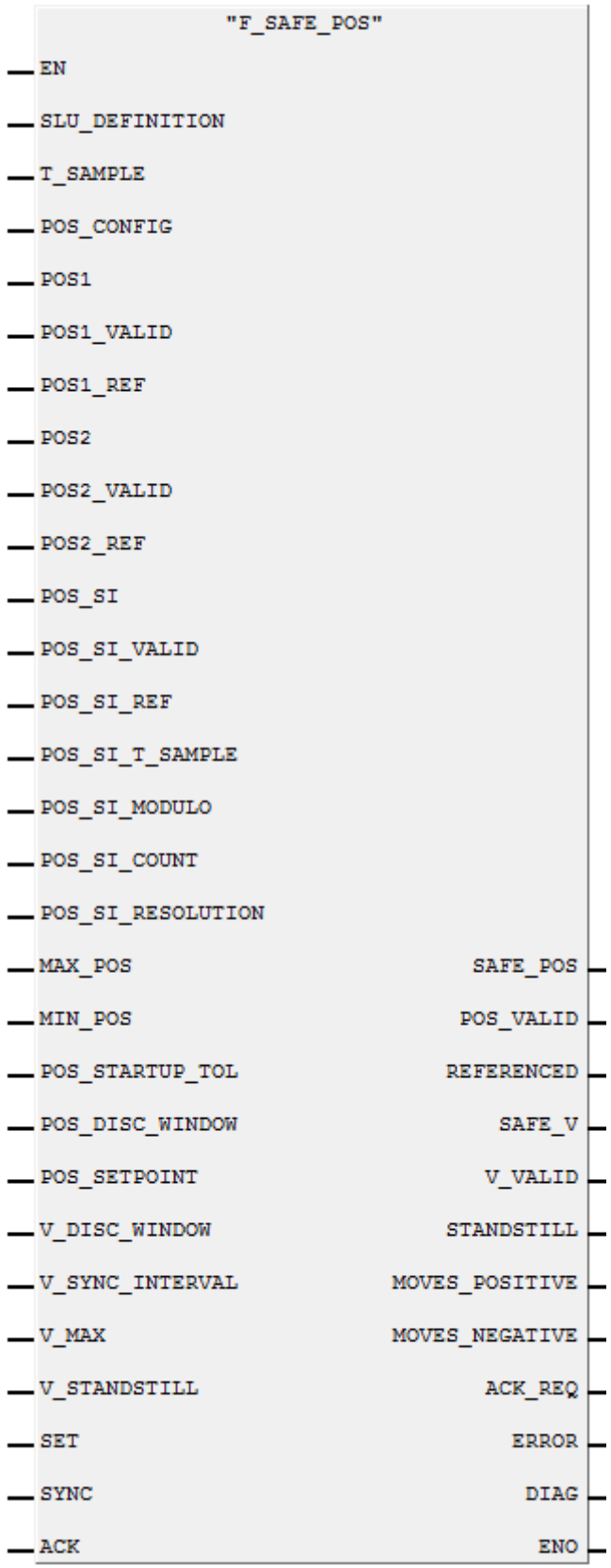
Safety notes and instructions
It is the users responsibility to interconnect and parameterize the ASRS blocks according to the standards applicable for their particular application. This is especially true for the test rates/intervals for the brake test and the overload/slack cable detection – as well as all load and velocity limits.

Safety notes and instructions
All position limits must be selected so that when these are exceeded the particular axis can come to a standstill before the end of the travel range. The value to be parameterized is also dependent on the maximum velocity to be expected for the specific application – as well as the maximum possible and permitted deceleration.

3.2 Fail-safe function block F_SAFE_POS

3.2.1 Introduction

The fail-safe function block F_SAFE_POS generates a safety-related position actual value based on the discrepancy comparison from two encoders. A velocity is calculated from the motor encoder value and is verified by a position discrepancy comparison using an absolute encoder. Within a time that can be parameterized, the positions between the motor encoder and the second encoder used to check the plausibility may not differ from one another by more than a slip tolerance so that the velocity value can be considered to be safety-related.
The safety-related position and velocity form the basis for the other blocks described in this document.
Redundant position sensing is always required if the position cannot be uniquely sensed using the Safety Integrated (SI) motor encoder in the drive. This is the case if the encoder cannot be mounted in a safety-related fashion, the mechanical system has slip or is subject to elongation (e.g. travel gear with wheel-rail system or hoisting gear with cable winch). Then, the position-related safety functions in the SI of the drive cannot be used. A direct measuring system must be employed to monitor the position. This is realized in the F-CPU using this particular block. The motor measuring system can then only be used to check the plausibility of the direct position actual value.
For applications where considerable slip is to be expected, and therefore the motor encoder cannot be used regarding the plausibility of the position, this block offers the option of deriving the safety-related position by deriving the discrepancy between two direct measuring systems.



Note

When using this block, block **F_BO_W/BO_W (FC 176)** and block **F_W_BO/W_BO (FC177)** must be available in the block folder. It is not permissible that these blocks are renumbered!

3.2.2 Connections

All bool variables listed in the following tables are preassigned FALSE, all integer variables are preassigned 0 – and all word variables preassigned W#16#0.

3.2.2.1 Inputs

Name	Data type	Description
SLU_DEFINITION	Int	**Definition of the unit SLU [mm]** Conversion factor from unit SLU to mm 1 <= SLU_DEFINITION <= 32
T_SAMPLE	Int	**Block sampling time in [ms]** Interval in which the safety program is called 0 < T_SAMPLE <= 2 * POS_SI_T_SAMPLE
POS_CONFIG	Word	**Configuration word for encoder interconnection** Bit 0: 1: Three-encoder system: two direct measuring systems + motor encoder 0: two-encoder system one direct measuring system + motor encoder
POS1	Int	**Value of the direct measuring system after scaling [SLU]** Value from the process image after scaling using F_SCALE_SLU
POS1_VALID	Bool	**Encoder signal status** 1: encoder signal valid 0: encoder fault
POS1_REF	Bool	**Status, absolute reference of the encoder signal** 1: encoder referenced 0: encoder not referenced
POS2	Int	**Value of the optional 2nd measuring system after scaling [SLU]** Value from the process image after scaling using F_SCALE_SLU
POS2_VALID	Bool	**Encoder signal status** 1: encoder signal valid 0: encoder fault
POS2_REF	Bool	**Status, absolute reference of the encoder signal** 1: encoder referenced 0: encoder not referenced
POS_SI	Int	**Value of the motor encoder Safety Integrated [RESOLUTION]** Value from the process image
POS_SI_VALID	Bool	**Encoder signal status** 1: encoder signal valid 0: encoder fault
POS_SI_REF	Bool	**Status, absolute reference of the encoder signal** 1: encoder referenced 0: encoder not referenced
POS_SI_T_SAMPLE	Int	**Sampling time of Safety Integrated [ms]** SI sampling time configured in the drive 1 <= POS_SI_T_SAMPLE <= 1023
POS_SI_MODULO	Int	**Modulo range safety-related position actual value PROFIsafe [RESOLUTION]** Modulo range configured in the drive 1 <= POS_SI_MODULO
POS_SI_COUNT	Int	**Safety Integrated cycle counter** Cyclic counter value of telegram 901
POS_SI_RESOLUTION	Int	**Scaling factor, position actual value Safety Integrated p9574 [µm]** 1 <= POS_SI_RESOLUTION <= 546
MAX_POS	Int	**Max. permissible position [SLU]**
MIN_POS	Int	**Min. permissible position [SLU]**

POS_STARTUP_TOL	Int	**Max. permissible deviation when powering up [SLU]** When starting, the value of the second encoder used to check the plausibility must not deviate more than this value from encoder 1 POS_STARTUP_TOL >= 0
POS_DISC_WINDOW	Int	**Max. permissible encoder deviation in operation [SLU]** The value of the second encoder used to check the plausibility must not deviate more than this value from encoder 1 POS_DISC_WINDOW >= 0
POS_SETPOINT	Int	**Reference position [SLU]** With a pos. edge at SET, POS1 as well as the redundant encoder (POS2 or POS_SI; depending on POS_CONFIG Bit 0) are synchronized at this position
V_DISC_WINDOW	Int	**Tolerance window, velocity monitoring** Max. permissible increase of the encoder deviation [SLU] within V_SYNC_INTERVALL for velocity monitoring V_DISC_WINDOW >= 0
V_SYNC_INTERVALL	Int	**Tracking interval, velocity monitoring [ms]** V_SYNC_INTERVALL > 0
V_MAX	Int	**Max. permissible velocity [SLU/mm]**
V_STANDSTILL	Int	**Velocity limit for standstill detection [SLU/min]**
SET	Bool	**Referencing** 0 -> 1: offset of encoders regarding calibrate POS_SETPOINT
SYNC	Bool	**Synchronizing** 0 -> 1: encoder used to check the plausibility is synchronized with encoder 1
ACK	Bool	**Acknowledging** If a fault occurs in normal operation, then this must be reset using ACK before the system can be restarted. The acknowledgment is only realized with a positive edge at ACK, and has no effect in normal operation.

3.2.2.2 Outputs

Name	Data type	Description
SAFE_POS	Int	**Safety-related position actual value [SLU]** Safety-related position for all additional blocks in this block package
POS_VALID	Bool	**Status, position actual value** 1: SAFE_POS was generated in a safety-related fashion
REFERENCED	Bool	**Status, absolute reference** 1: Both encoders are referenced, and the discrepancy between the two encoders is within the tolerance window
SAFE_V	Int	**Safety-related velocity actual value [SLU/min]** Safety-related velocity for all additional blocks in this block package
V_VALID	Bool	**Status, velocity actual value** 1: SAFE_V was generated in a safety-related fashion
STANDSTILL	Bool	**Zero speed detection** 1: Actual velocity less than V_STANDSTILL
MOVES_POSITIVE	Bool	**Motion in the positive direction**
MOVES_NEGATIVE	Bool	**Motion in the negative direction**
ACK_REQ	Bool	**Acknowledgment request** If a fault has occurred, but is no longer active, and can therefore be acknowledged, then this block indicates this using a 1 signal at ACK_REQ.
ERROR	Bool	**Error** This output is set if the block has been incorrectly parameterized – or if the block identifies a potentially dangerous combination of input signals in operation. The output re-

		mains set until an error is no longer active and has been acknowledged.
DIAG	Word	**Diagnostic word** Information regarding the function status and errors of the block are output here.

3.2.2.3 Structure of DIAG

Bit No.	Description	Reset condition
0	Value range violation of the input variables	1 <= SLU_DEFINITION <= 32 and 1 <= POS_SI_MODULO and 1 <= POS_SI_RESOLUTION <= 546 and 1 <=POS_SI_ T_SAMPLE <= 1023 and V_SYNC_INTERVALL > 0 and 0 < T_SAMPLE <= 2 * POS_SI_T_SAMPLE and V_DISC_WINDOW >= 0 and POS_DISC_WINDOW >= 0 and POS_STARTUP_TOL >= 0
1	The ratio between the input variables cannot be represented as an integer number	The following ratios must be able to be represented as an integer number: (POS_SI_MODULO/1000) * POS_SI_RESOLUTION / 360 and SLU_DEFINITION * 1000 / POS_SI_RESOLUTION and POS_SI_RESOLUTION * 60 / T_SAMPLE * SLU_DEFINITION and V_SYNC_INTERVALL / T_SAMPLE
2	Incorrect relationship between the input variables	The following relationship between the variables must apply: V_MAX >= V_STANDSTILL and MAX_POS >= MIN_POS
3	Position when powering up not plausible, Safety-related referencing required	Reference point approach until a positive edge at SET
4	Max. permissible position discrepancy exceeded	Pos. edge at SYNC or SET
5	Min. one encoder not referenced, safety-related referencing required	Reference point approach until a positive edge at SET
6	Actual position > MAX_POS	Actual position <= MAX_POS and positive edge at ACK
7	Actual position < MIN_POS	Actual position >= MIN_POS and positive edge at ACK
8	Actual velocity > V_MAX	Actual velocity <= V_MAX and positive edge at ACK
9	Max. permissible velocity discrepancy exceeded	Velocity discrepancy <= V_DISC_WINDOW and positive edge at ACK
10	Invalid raw encoder values	POS1 and POS_SI/POS2 supply valid values (VALID = 1) and positive edge at ACK
11	Too many SI clock cycles between two F-CALL calls. Internal calculation overflow	Positive edge at ACK
12	Reserved	---
13	Reserved	---
14	Reserved	---
15	Reserved	---

3.2.3 Interrelationship between the assignment of the block inputs and the drive configuration

In Distributed Safety, only the INTEGER value range is available for analog values; it is not possible to use the DINT or REAL data types – known from the standard program – in the safety program. As result, not only a coarse resolution (only an integer number of millimeters) is obtained when parameterizing the safety functions in SLU, but also restrictions apply with reference to the internal plausibility monitoring of the two encoder signals as well as the velocity monitoring. As a consequence, it is not feasible to calculate the velocity, based on an absolute position actual value of telegram 901.

However, in order to allow the velocity to be calculated based on the motor encoder, the axis must always be configured in Safety Integrated as rotary axis with modulo correction. In this case, the spindle pitch is also taken into account in the gearbox ratio, so that the absolute values of positions and velocities have the units of mm and mm/min.
For users, there is then only a deviation in the displayed unit (degrees and rpm) in the SI dialogs in STARTER.
Depending on the scaling factor p9574 of safety-related telegram 901, the modulo range p9505 is set to a multiple of 360. This means that the position actual value cannot overflow beyond the 16-bit integer range.

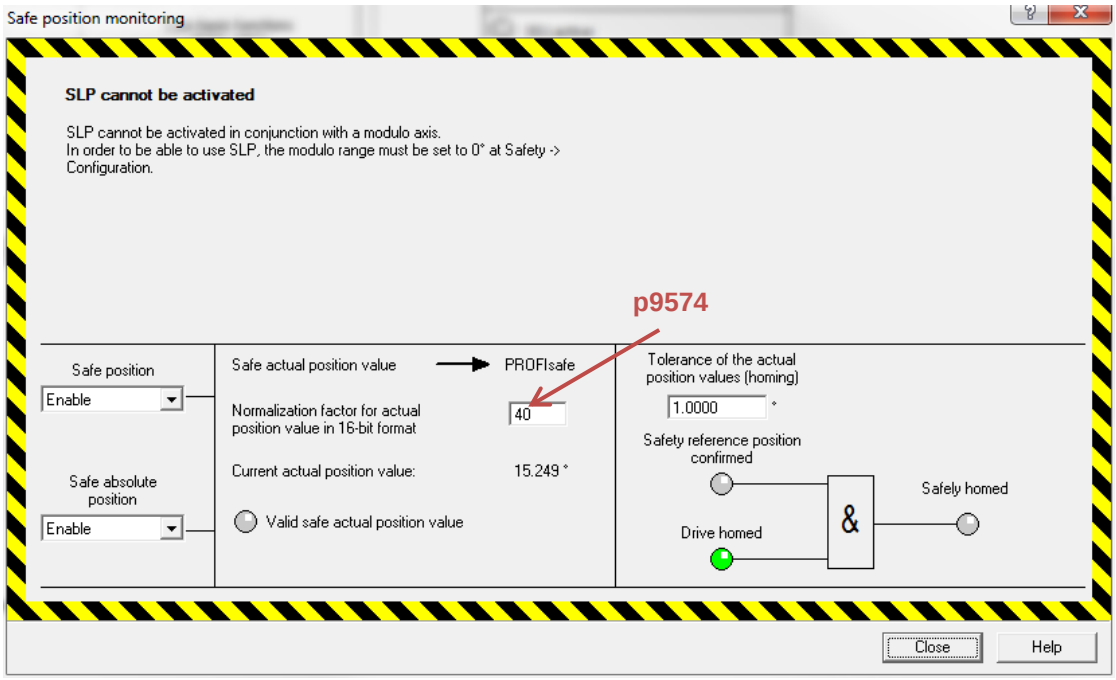


Figure 9: Scaling factor p9574

Scaling factor p9574 defines the resolution of the safety-related position actual value in telegram 901, and therefore input parameter POS_SI_RESOLUTION.
This means that the SI position actual value (capable of overflow and with a high resolution) can be transferred to the F-CPU, where the velocity actual value is calculated.
Correspondingly, the 16-bit position actual value should always be interconnected as modulo value at input POS_SI.

The modulo range p9505 configured in the drive, multiplied by 1000/p9574, is parameterized at input POS_SI_MODULO.

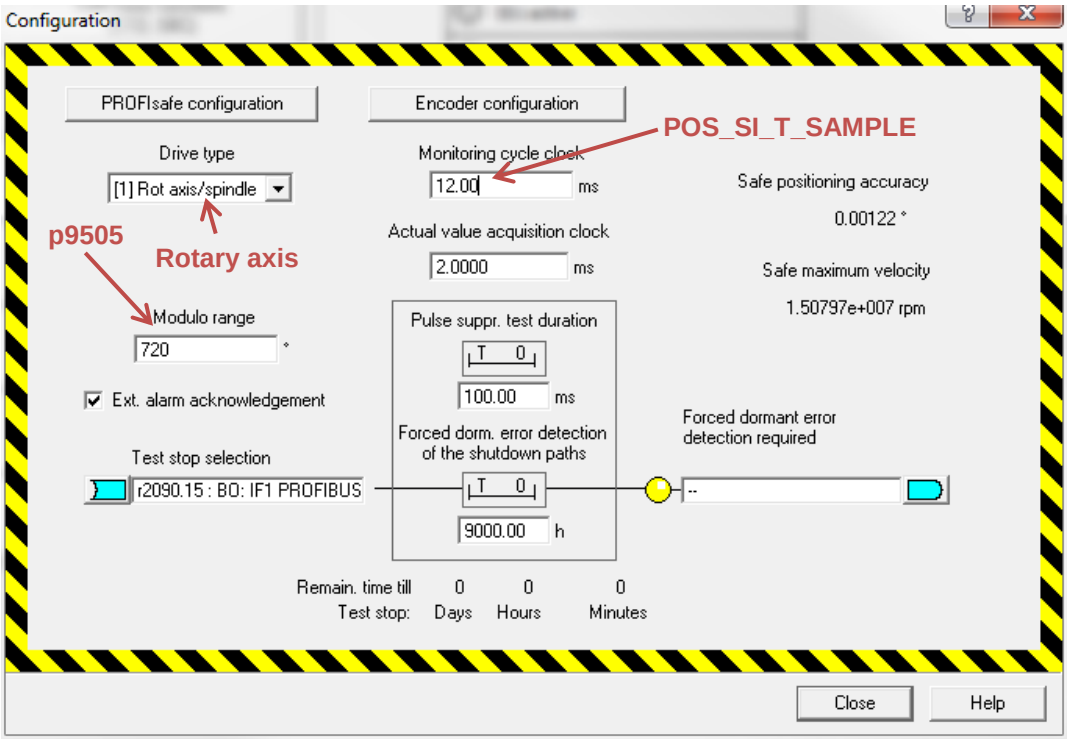


Figure 10: SI configuration

As described above, the drive type must be set to rotary axis. The modulo range must be selected so that during one cycle in the PLC, two rollovers cannot occur. As a consequence, this value depends on the maximum velocity.

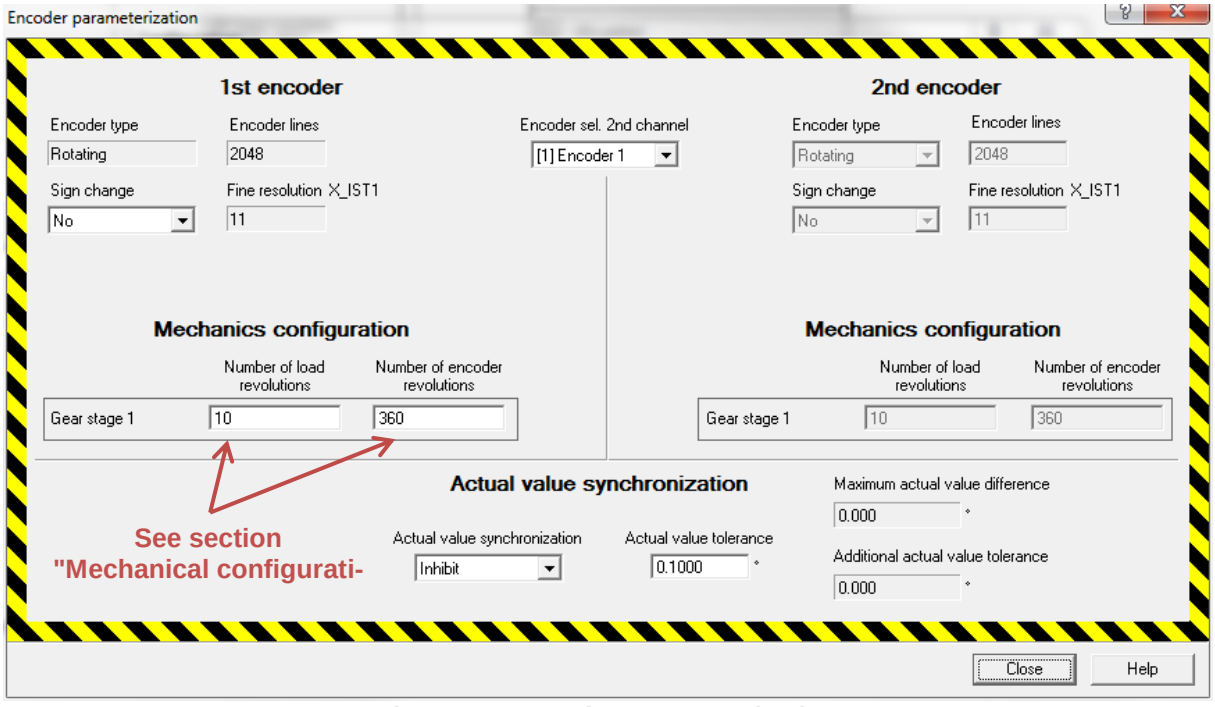


Figure 11: Encoder parameterization

The modulo value at POS_SI is directly converted into the SLU unit in the block; POS_SI_RESOLUTION and SLU_DEFINITION define the conversion factor.

Configuring the mechanical system:

Converting from degrees to a unit length must be realized using the gearbox ratio. This is realized by entering divisor **360** for the number of encoder revolutions. Based on the number of load revolutions, the feed constant/spindle pitch can now be specified, which corresponds to SLU_DEFINITION – i.e. the relationship between mm/SLU at F_SAFE_POS.

As example, if a spindle pitch of 10mm/revolution would be set at Fig. 11, a SLU_DEFINITION of 10mm/SLU would be entered at the block. Now, at each motor revolution, the block increases the SAFE_POS value by 1 SLU.

If there is a real gearbox between the motor and load in the system, then this ratio must be additionally taken into account.

Configuring the encoder at the block

To calculate a position actual value in SLU, the incremental relative position change of the modulo actual value – as well as its overflows – are acquired by the block and summed. As a consequence, a relative position is available to check the plausibility of the first channel POS_1 for CONFIG bit0 = 0. The direction of rotation of the motor encoder is adapted in the SI section of the drive.

The 32-bit value of the direct encoder is scaled to SLU in the standard program section of the CPU. The scaled value is interconnected at POS1; assuming a second direct encoder is being used, its scaled value is interconnected at POS2 – and CONFIG bit0 is set to 1.
The plausibility of the calculated velocity from POS_SI is always checked using the encoder interconnected at POS1.
For POS_CONFIG bit0 = 0, the plausibility of position POS1 is checked using the encoder interconnected at POS_SI; for POS_CONFIG bit0 = 1, the plausibility of POS1 is checked with respect to POS2.

Note
POS1 and POS2 expect opposing values!

Note
It should be ensured that the relative position change of the modulo actual value between two calls of the F_SAFE_POS block does not have two overflows, as in this case, only one overflow would be detected.

Safety notes and instructions
The signals interconnected at POS1 and POS2 must come from two independent sources. If one signal source is connected at both inputs, then non-plausible values of this channel cannot be identified, for example. As a consequence, the safety integrity of the block is no longer guaranteed.

3.2.4 Principle of operation

3.2.4.1 Parameterization

1. The position-defining encoder is interconnected at POS1, i.e. after the conversion to SLU (this is realized in the standard program using block SCALE_DINT), the encoder raw value is available as a 16-bit value at output SAFE_POS.
2. A possibly available additional validity query for the position value (e.g. error bit of the module) can be interconnected at input POS1_VALID. If information of this type is not available, then the input must be permanently set to TRUE.
3. The "Position actual value referenced" information is interconnected at input POS1_REF.
4. The sampling time of the block is parameterized at input T_SAMPLE. For instance, the configured call interval of the cyclic interrupt OB, from which the safety program is called.
5. The resolution of a position increment from POS_SI in µm is defined at POS_SI_RESOLUTION; POS_SI_T_SAMPLE defines the sampling rate of SI in the drive, POS_SI_COUNT should be interconnected with the counter value from the telegram 901.
6. The significance of SLU in mm is defined at SLU_DEFINITION; T_SAMPLE, POS_SI_RESOLUTION and SLU_DEFINITION are relevant for calculations made inside the block.
7. The modulo value of the motor encoder from telegram 901 is interconnected at input POS_SI
8. For applications involving a high degree of slip, it is possible to interconnect a second direct encoder at POS2

9. You can switch over between the mode for one and two direct encoders using the WORD parameter specified at POS_CONFIG. When the 0 bit is set, then the block is in the mode for two direct encoders, otherwise in the mode for one direct encoder.
10. The significance of inputs POS2_VALID and POS2_IS_REF or POS_SI_REF and POS_SI _VALID is equivalent to the corresponding inputs for POS1.
11. As described above, the modulo range configured in the drive is parameterized at POS_SI_MODULO.

When parameterizing (assigning parameters) it should be ensured that the following relationships can be represented as integer numbers:

POS_SI_MODULO / 1000

(POS_SI_MODULO/1000) * POS_SI_RESOLUTION / 360

SLU_DEFINITION * 1000 / POS_SI_RESOLUTION

POS_SI_RESOLUTION * 60 / (POS_SI_T_SAMPLE * SLU_DEFINITION)

V_SYNC_INTERVALL / T_SAMPLE

Further, the following relationship between the input variables must apply:

V_MAX >= V_STANDSTILL

MAX_POS >= MIN_POS

Sampling rates/isochronous mode

To calculate the velocity, the block clock cycle of the F_SAFE_POS block on the CPU (T_SAMPLE, generally, the interval in which the safety program is called) is not used as time basis, but instead, the SI clock cycle in the drive. (POS_SI_T_SAMPLE)
To achieve this, the communication between the CPU and drive must be isochronous; the bus cycle must be less or equal to the SI clock cycle in the drive. The safety program does not have to be called at an isochronous execution level. Further, it must be guaranteed that T_SAMPLE <= 2x POS_SI_T_SAMPLE. Otherwise, inadmissibly high subsampling will occur (as a result of calculations executed in the block).

The permissible value ranges of the individual inputs should be taken from the table describing the inputs.

The block identifies if not all of the mentioned preconditions are satisfied, and this is signaled as parameterizing error with the appropriately set DIAG bits.

Note
The block only checks the parameterization at the 1st call. This results in a subsequent increased block performance.
As a consequence, reparameterization is not permitted when safety operation is deactivated. The safety program must be regenerated and loaded each time that any of the block operating parameters are changed.

3.2.4.2 Starting characteristics

12. After a CPU restart, outputs V_VALID, POS_VALID and REFERENCED initially have a 0 signal.
13. V_VALID is set to 1 after acknowledgment using a positive signal edge at ACK.
14. To be able to travel to the reference point, POS_VALID must be set to 1 by synchronizing both encoders using a positive signal edge at SYNC. Now, the position value is valid as such, and can be used for information regarding the relative position; however, it is not permissible to evaluate the position as safety-related absolute position as long as REFERENCED has a 0 signal.

15. If the axis is at the reference point defined using input POS_SETPOINT, then the block is referenced using a positive signal edge at SET, REFERENCED changes to 1. The position value output at SAFE_POS can now also be used as absolute position.

3.2.4.3 Actual position value

16. When all of the relevant encoders return (POS_x_VALID), if the calculated position value from POS1 deviates by more than POS_STARTUP_TOL from the internally saved reference value (which was saved when POS_x_VALID went to zero) then REFERENCED is reset to 0, ERROR outputs a 1 single and DIAG bit 3 is set.
17. If, in operation, the calculated positions from POS1 and POS_SI differ by more than the value parameterized at input POS_DISC_WINDOW and if POS_CONFIG bit0 is not set, then POS_VALID is set to 0, ERROR outputs a 1 signal and DIAG bit 4 is set.
18. If POS_CONFIG bit0 = 1 – and in operation the calculated positions from POS1 and POS2 differ by more than the value parameterized at input POS_DISC_WINDOW, then POS_VALID is set to 0, ERROR outputs a 1 signal and DIAG bit 4 is set.
19. If none of these errors occurs, and at POS1_VALID and at POS_SI_VALID a 1 signal is available – and for POS_CONFIG bit0 = 1, also POS2_VALID has a 1 signal – then the POS_VALID output is 1 to indicate that SAFE_POS is valid.
20. When SAFE_POS exceeds the value of MAX_POS, ERROR and DIAG bit 6 are set. POS_VALID returns to 0.
21. When SAFE_POS falls below the value of MIN_POS, ERROR and DIAG bit 7 are set. POS_VALID returns to 0.
22. The block itself does not provide any type of retraction logic. Using a suitable external logic circuit, users must ensure that when REFERENCED = 0, the axis can only travel with a safely reduced velocity.

Warning

As long as REFERENCED outputs a 0 signal, the position can only be used for relative position information; an absolute evaluation is only permissible when REFERENCED = 1.

For REFERENCED = 0, it is only permissible that the axes travel with a safely reduced speed depending on the specific application.

Warning

As long as POS_VALID outputs a 0 signal, the position actual value is not generated as a safety-related value. An application-specific stop response should be initiated for a falling edge.

3.2.4.4 Actual velocity value

23. The safety-related velocity, calculated from the modulo value of the motor encoder interconnected at POS_SI, is output at SAFE_V.
24. If SAFE_V falls below the value parameterized at V_STANDSTILL, this standstill is signaled at output STANDSTILL using a 1 signal.
25. If SAFE_V is greater than/equal to V_STANDSTILL, a 1 signal is output at MOVES_POSITIVE if over time SAFE_POS assumes increasingly higher values – or a 1 signal is output at MOVE_NEGATIVE if over time SAFE_POS assumes increasingly lower values.
26. If SAFE_V exceeds the value parameterized at V_MAX, ERROR is set to 1 and DIAG bit 8 is set. V_VALID returns to 0.
27. At input V_DISC_WINDOW it is parameterized as to what extent (specified in SLU) the values of POS1 and POS_SI are permitted to drift apart within V_SYNC_INTERVALL without initiating a velocity error.
28. After the time parameterized at V_SYNC_INTERVALL, the discrepancy between the relative positions from POS1 and POS_SI for the velocity monitoring, accumulated in the block, is eliminated to permit tolerance for slip.
29. If the drift between POS1 and POS_SI exceeds the value parameterized at V_DISC_WINDOW, then ERROR and DIAG bit 9 are set. V_VALID returns to 0.

Warning

As long as V_VALID outputs a 0 signal, the velocity actual value is not generated as a safety-related value.

3.2.4.5 Referencing

30. For a positive signal edge at SET a safety-related adjustment is carried out in the book itself, where, for both position raw values, a separate position offset, relative to the value specified at input POS_SETPOINT, is determined and saved. The REFERENCED output is set if referencing was successfully completed.
31. If referencing is to be successfully completed, both encoder actual values must be valid and referenced (POS1_VALID/REF & POS_SI_VALID/REF = 1 (for POS_CONFIG bit 0 = 0) or POS1_VALID/REF & POS2_VALID/REF = 1 (for POS_CONFIG bit 0 = 1)
32. REFERENCED is then the set with a rising edge at input SET and the offsets are internally saved.
33. REFERENCED returns to zero as soon as the position tolerance window POS_DISC_WINDOW is violated – or as soon as an encoder is no longer referenced.
34. ERROR as well as DIAG bit 5 is set to 1 if the above specified conditions are not satisfied.

After an encoder fault, the block can reproduce SAFE_POS without requiring a reference point approach.
If POS1_VALID and POS_SI_VALID or POS2_VALID (depending on POS_CONFIG bit 0) have a rising edge, the position is reproduced in the block according to the following scheme:

35. POS_CONFIG bit 0 == 0:
    A check is made as to whether POS1 corresponds to reference value X_{REF}, taking into account POS_STARTUP_TOL. Reference value X_{REF} was previously saved with a falling edge at POS_VALID and in the safely referenced mode (REFERENCED was set at this instant in time). The fact that at reactivation POS1 corresponds to the internally saved reference value X_{REF} confirms the adjustment. The relative encoder is then automatically calibrated.

36. POS_CONFIG bit 0 == 1:
    A check is made as to whether both actual values are referenced (POS1_REF and POS2_REF == 1), and lie within the tolerance window POS_STARTUP_TOL. If this is the case, then REFERENCED and POS_VALID are again set to 1.
37. If one of the encoders is not referenced, or if the position tolerance was violated, then REFERENCED returns to 0.

38. In the case that POS_STARTUP_TOL was exceeded, in order to set REFERENCED back to 1, a reference point approach must be carried out, as described under Point 30.
39. Non-plausible values at POS_SI are flagged using DIAG bit No. 11, and lead to a parameterizing error. A positive edge at ACK is used for reset.

Note

An encoder fault always leads to the withdrawal of input signal POS1_VALID or POS_SI_VALID/POS2_VALID. As a consequence, the safety-related actual value is immediately declared to be invalid, and the position tolerance monitoring is hidden. As a consequence, an encoder fault does not mean that the safety-related reference is immediately lost, and the safety-related position can be reproduced after the encoder returns.

3.2.4.6 Synchronizing encoders

40. Using a positive edge at SYNC, both encoders are synchronized, i.e. the descrepancy that has accumulated between the two encoders is set to 0. Both encoder actual values must be valid to do this.

Warning

Cyclic synchronizing means that the two-channel structure is bypassed for the position actual value processing, and is therefore not permissible. The specific application defines just when synchronizing is permissible, and it is the responsibility of the user to ensure that the correct logic interlocking is used.

3.2.4.7 Acknowledging faults

41. DIAG and ERROR are reset to 0 with a positive edge at ACK, assuming that no other fault is active. The block immediately indicates that it can be acknowledged with a 1 signal at its output ACK_REQ. ACK_REQ is reset to 0 after a positive edge at ACK.

3.2.5 Additional diagnostic functions

To optimize the system response, the currently effective position and velocity discrepancy can be read out using the instance DB of the block for diagnostic purposes.

Note

F instance data can only be accessed from the standard program.

The actual absolute value of the velocity discrepancy is designated in the block as "abs_v_discrepancy", and is in the instance DB at address DBW118 – the data type is INTEGER

The actual absolute value of the position discrepancy is designated in the block as "abs_delta_pos", and is in the instance DB at address DBW120 – the data type is INTEGER

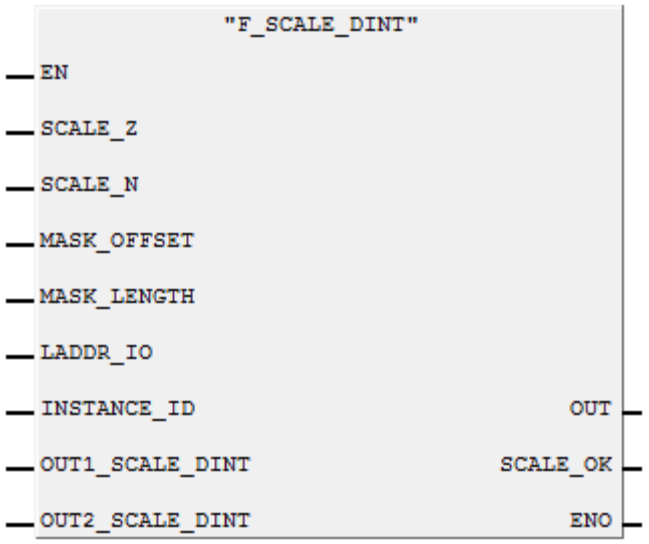
This data can be accessed from the standard program as shown in the following example:

```
FB1 : Read diagnostic data
Comment:

Network 1 : Amount of current speed discrepancy (F_SAFE_POS)
      L     "F_SAFE_POS_IDB".abs_v_discrepancy    DB200.DBW118
      T     #V_Discrepancy                       #V_Discrepancy

Network 2 : Amount of current position discrepancy (F_SAFE_POS)
      L     "F_SAFE_POS_IDB".abs_delta_pos    DB200.DBW120
      T     #POS_Discrepancy                  #POS_Discrepancy
```

3.3 Fail-safe function block F_SCALE_DINT

3.3.1 Introduction

Fail-safe function block F_SCALE_DINT, together with an associated block in the standard program (SCALE_DINT), scales a DINT value from the process image to a 16 bit value with unit SLU. The actual scaling itself is realized in the standard program; the result is safely transferred into the F part of the CPU. This scaling, i.e. specifying the scaling factor for example is parameterized using block F_SCALE_DINT in the safety program; the same is true for the plausibility check as to whether the scaled value was correctly transferred from the standard part into the F part.
The value scaled by the standard SCALE_DINT block is made available to the safety program after a plausibility check using this block.



3.3.2 Connections

All bool variables listed in the following tables are preassigned FALSE, all integer variables are preassigned 0 – and all word variables preassigned W#16#0.

3.3.2.1 Inputs

Name	Data type	Description
SCALE_Z	Int	**Numerator, scaling factor** This value is used for scaling and adapting the direction of the raw encoder value from the process image to the unit SLU
SCALE_N	Int	**Denominator, scaling factor** This value is used for scaling the raw encoder value from the process image to the unit SLU
LADDR_IO	Int	**Peripheral address** The address of the encoder from HW Config should be interconnected here, whose raw value should be scaled.
MASK_OFFSET	Int	**Masking, bit offset** Position of the first valid bit of the position actual value in the raw encoder value
MASK_LENGTH	Int	**Masking, bit length** Length of the position actual value in the encoder raw value in bits
INSTANCE_ID	Int	**Number instance DB** The number of the instance DB of the block call is parameterized at this input. This parameter is required to check the plausibility of the data transfer between F_SCALE_DINT and

		the SCALE_DINT block in the standard program. Here, it is mandatory that precisely the same number as that of the instance DB of the particular F_SCALE_DINT block is parameterized.
OUT1_SCALE_DINT	Word	**OUT1 value of the "SCALE_DINT" block** This output should be connected with the corresponding output OUT1 of the SCALE_DINT block in the standard program using a bit memory word.
OUT2_SCALE_DINT	Word	**OUT2 value of the "SCALE_DINT" block** This output should be connected with the corresponding output OUT2 of the SCALE_DINT block in the standard program using a bit memory word.

3.3.2.2 Outputs

Name	Data type	Description
OUT	Int	**Scaling result**
SCALE_OK	Bool	**Scaling status** 1: Scaling is valid if a currently scaled value is output at OUT 0: Error, the last valid value is output at OUT A stop response should be initiated if this output changes to 0.

3.3.3 Principle of operation

3.3.3.1 Parameterization

1. At input SCALE_Z the numerator is parameterized, at input SCALE_N the denominator of the conversion factor from a 32-bit encoder raw value to 16 bit SLU.
2. If SCALE_N <= 0 is parameterized, then ERROR is set to 1, SCALE_OK is set to 0, and the last valid value is output at OUT.
3. If the scale value is to be inverted, then this is realized using a negative sign of SCALE_Z.
4. LADDR_IO is used to define from which peripheral address of the standard block the 32-bit value to be scaled is supplied.
5. Parameters MASK_OFFSET and MASK_LENGTH are used to define the screen form (mask) for the position actual value in the encoder raw value.
6. If the sum of parameters MASK_OFFSET and MASK_LENGTH > 32, MASK_LENGTH < 16 or MASK_OFFSET < 0, then ERROR is set to 1 and SCALE_OK is set to zero, and the last valid value is output at OUT.
7. The number of the instance DB of the block call should be parameterized at INSTANCE_ID.
8. If, at this input, a value is parameterize that differs from the actually used instance DB number, then SCALE_OK changes to 0 – and the last valid value is output at OUT until SCALE_OK again has a 1 signal.
9. Block SCALE_DINT in the standard program reads its parameterization from this instance DB.
10. Due to the fact that parameterization is realized using a block in the safety program – and not directly at the standard program SCALE_DINT block, these parameters are taken into account when generating the safety program signature.
11. Therefore, when the scaling factor is changed, this also influences the signature; further, the scaling factor can only be modified when the safety program password is known.
12. The factor defined using SCALE_Z and SCALE_N depends on the resolution of the direct measuring system, which is interconnected at LADDR_IO and the definition of unit SLU, as it was defined at the input SLU_DEFINITION of the F_SAFE_POS block.
13. For a direct measuring system resolution of 1µm and a selected SLU_DEFINITION of 12mm then a factor of 1:12000 would be obtained, for example; SCALE_Z would therefore be 1 and SCALE_N correspondingly 12000.

Safety notes and instructions
It is not permissible that inputs SCALE_Z, SCALE_N, LADDR_IO and INSTANCE_ID are interconnected with bit memories from the standard user program. Only then is it guaranteed that the scaling parameterization cannot be changed without knowing the safety program password.

Note

The permissible value ranges of the individual inputs should be taken from the table describing the inputs.

Note

It is mandatory that this block is called with a dedicated instance DB, it is not permissible to call it as multi-instance.

3.3.3.2 Output of the scaling result

14. Inputs OUT1_SCALE_DINT/OUT2_SCALE_DINT should be interconnected with the corresponding outputs OUT1/OUT2 of block SCALE_DINT in the standard program using bit memory words.
15. Internally, the block checks as to whether scale value transferred via these two inputs is plausible, i.e. whether after decoding, the identical value was transferred.
16. When this is the case, then this value is output at OUT and SCALE_OK signals that scaling is error-free using a 1 signal.
17. If an error occurs for the scaling itself, or for data transfer between the standard and the F part of the scaling, then SCALE_OK is set to 0, and the last valid value is output at OUT.

3.3.3.3 Error response

18. The block does not have to be acknowledged. A value that has been identified as being invalid at OUT is flagged using a 0 signal at SCALE_OK; it must be ensured that the system goes into a safe state by using a suitable logic circuit.
19. A possible logic circuit in this case is to logically combined output SCALE_OK with input POSx_VALID at block F_SAFE_POS.
    As soon as an invalid value is available at OUT, the block F_SAFE_POS now responds in a safety-related way by withdrawing POS_VALID; this signal is not automatically set again using SCALE_OK.

    The behavior of the ASRS block package is such that block F_SCALE_DINT is not required, if the encoder directly supplies the value in the SLU unit.

    Also refer to Chapter 0.

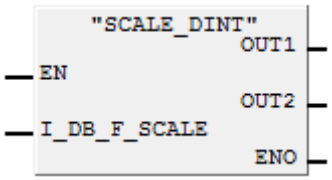
Safety notes and instructions

It must be carefully ensured that the machine does not automatically restart after a 1->0->1 change at SCALE_OK by suitably interconnecting output SCALE_OK in the user program.

3.4 SCALE_DINT function

3.4.1 Introduction

Function SCALE_DINT scales a 32-bit DINT value, which is read from the process image, to a 16-bit INT value. The 32-bit value corresponds to a raw sensor value, which is supplied from a standard I/O module, the 16 bit represents this encoder value in the SLU unit.
The parameter from which the peripheral address should be read as well as the scaling factor are defined in the safety program using the F_SCALE_DINT block.



3.4.2 Connections

3.4.2.1 Inputs

Name	Data type	Description
I_DB_F_SCALE	Pointer	**Instance DB No. of the associated F_SCALE_DINT block** A pointer to the I-DB of the F_SCALE_DINT block in the safety program is parameterized at this input. The parameterizing data for the scaling are read from this I-DB.

3.4.2.2 Outputs

Name	Data type	Description
OUT1	Word	**Intermediate result 1** A bit memory word is interconnected at this output. Using this output, the value of the standard I/O module, scaled to SLU is transferred, coded to the corresponding input of block F_SCALE_DINT in the safety program.
OUT2	Word	**Intermediate result 2** A bit memory word is interconnected at this output. Using this output, the value of the standard I/O module, scaled to SLU is transferred, coded to the corresponding input of block F_SCALE_DINT in the safety program.

Note

OUT1 and OUT2 are coded differently, the identical bit memory word cannot be used for both outputs.

3.4.3 Principle of operation

3.4.3.1 Parameterization

1. The block is parameterized in the safety program by calling the F_SCALE_DINT block. The scaling factor parameter and the start address in HW Config – which supplies the 32-bit value to be scaled – as well as the bit offset and the bit length for masking, are saved in its instance DB.
2. At input I_DB_F_SCALE, a pointer is parameterized to this instance DB in the form DBxxx.DBX0.0 .

3.4.3.2 Protective mechanisms and fault detection

3. The block has know-how protection, so that the internal logic cannot be manipulated. Due to the fact that the block is implemented as a function, then there is no instance DB. This means that there is absolutely no possibility of externally influencing the internal processing.
4. Sign-of-life monitoring identifies if the block is either not called or is not correctly processed.
5. These mechanisms guarantee that the scale value is calculated without being able to be manipulated.
6. For transfer to the safety program at outputs OUT1 and OUT2, the masked and scaled value is provided, logically combined with another check value.
7. The user must interconnect these two values with the corresponding inputs of the F_SCALE_DINT block in the safety program using bit memory words.
8. By transferring the masked and scaled values to the safety program in this way, data transfer errors of the value as well as incorrect user interconnection between SCALE_DINT blocks in the standard and F_SCALE_DINT blocks in the safety program can be diagnosed.

3.4.3.3 Error response

9. The block itself does not signal an error; the error evaluation is carried out using the F_SCALE_DINT block in the safety program.

Safety notes and instructions
It must be carefully ensured that the machine does not automatically restart after a 1->0->1 change at SCALE_OK by suitably interconnecting output SCALE_OK in the user program.

3.4.3.4 Timing

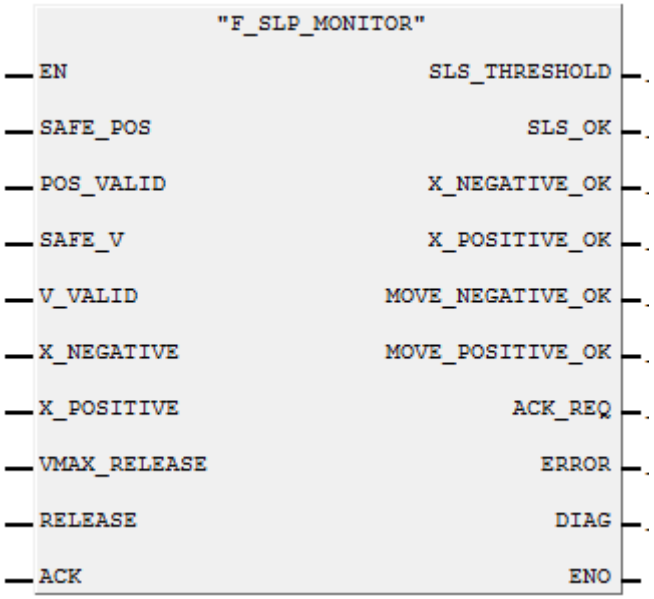
10. As a minimum, the block must be called in the same time grid as the F_SCALE_DINT block.
11. If this is not the case, then F_SCALE_DINT signals an error, i.e. its SCALE_OK output changes to 0.

Note
We recommend that the SCALE_DINT block is called in the same cyclic interrupt OB as the safety program, e.g. immediately before the F_CALL block.

3.5 Fail-safe function block F_SLP_MONITOR

3.5.1 Introduction

The fail-safe function block F_SLP_MONITOR is used to safely monitor the end stops of a travel range. If the defined travel range is exited, the block flags this and depending on the user interconnection, a stop response can be initiated.
The block has a retraction logic function so that the axis can return to the limited travel range. The axis can move away from the end stop with a safely-limited velocity parameterized at the block. The block provides two signals to control the the SDI function in the drive to inhibit continued travel towards the end stop.



Note

When using this block, block **F_BO_W/BO_W (FC 176)** must be available in the block folder. It is not permissible that this is renumbered!

3.5.1 Connections

All bool variables listed in the following tables are preassigned FALSE, all integer variables are preassigned 0 – and all word variables preassigned W#16#0.

3.5.1.1 Inputs

Name	Data type	Description
SAFE_POS	Int	**Safety actual position [SLU]** is supplied from the F_SAFE_POS block. The machine is stopped if this value violates one of the limit values X_POSITIVE or X_NEGATIVE.
POS_VALID	Bool	**Actual position valid** is supplied from the F_SAFE_POS block. 1: position is possible 0: position is not plausible, e.g. the discrepancy between the two encoders is outside the tolerance band. DIAG bit No. 5 is set if a 0 signal is present here.
SAFE_V	Int	**Safe actual velocity [SLU/mm]** is supplied from the F_SAFE_POS block. This input is interrogated if the user activates the retraction mode of the block. If the actual velocity is greater than the upper limit parameterized at VMAX_RELEASE then output SLS_OK is reset and the machine is stopped.

V_VALID	Bool	**Actual velocity valid** is supplied from the F_SAFE_POS block. 1: velocity is plausible 0: velocity is not plausible, e.g. the increase of the deviation between the two encoders over time is outside the tolerance band If a 0 signal is present here and the block is in the retraction mode, then DIAG bit No. 6 is set.
X_NEGATIVE	Int	**Min. permissible position [SLU]** If the value at input SAFE_POS falls below this limit value, then output X_NEGATIVE_OK is reset
X_POSITIVE	Int	**Max. permissible position [SLU]** If the value at input SAFE_POS exceeds this limit value, then output X_POSITIVE_OK is reset
VMAX_RELEASE	Int	**Retraction velocity [SLU/min]** If the block is in the retraction mode, then this value is output at SLS_THRESHOLD. VMAX_RELEASE must be parameterized in the range 1-32767. Otherwise, DIAG bit No. 4 is set
RELEASE	Bool	**Retracting** If the permitted position range was exited, then after a positive edge at this input, the axis can be traversed back into the permissible position range with the velocity parameterized at VMAX_RELEASE. While retracting, motion is immediately stopped if a 0 signal is present at this input.
ACK	Bool	**Acknowledging** If a fault occurs in normal operation, then this must be reset using ACK before the system can be restarted. The acknowledgment is only realized with a positive edge at ACK, and has no effect in normal operation.

3.5.1.2 Outputs

Name	Data type	Description
SLS_THRESHOLD	Int	**SLS limit [SLU/min]** The maximum permissible traversing velocity effective at this time is output at this output. In normal operation, this is 32767, If the user retracts the axis, then VMAX_RELEASE is output here. If VMAX_RELEASE is parameterized <= 0 then equivalent value 1 is output here.
SLS_OK	Bool	**Status, SLS limit** 1: SAFE_V is less than/equal to SLS_THRESHOLD 0: SAFE_V has exceeded the SLS_THRESHOLD value. A stop response should be initiated if this output changes to 0.
X_NEGATIVE_OK	Bool	**Status, minimum position** 1: SAFE_POS is greater than/equal to X_NEGATIVE 0: SAFE_POS has fallen below the value of X_NEGATIVE. A stop response should be initiated if this output changes to 0.
X_POSITIVE_OK	Bool	**Status, maximum position** 1: SAFE_POS is less than/equal to X_POSITIVE 0: SAFE_POS has exceeded the value of X_POSITIVE. A stop response should be initiated if this output changes to 0.
MOVE_NEGATIVE_OK	Bool	**Motion permitted in the negative direction** If a 0 signal is available at this output, then it is not permissible that the machine continues to travel in the negative direction. The output is then set to 0, as soon as SAFE_POS assumes values less than X_NEGATIVE. If SAFE_POS again lies above X_NEGATIVE, then after acknowledgment, the output is set again.

MOVE_POSITIVE_OK	Bool	**Motion permitted in the positive direction** If a 0 signal is available at this output, then it is not permissible that the machine continues to travel in the positive direction. The output is then set to 0, as soon as SAFE_POS assumes values greater than X_POSITIVE. If SAFE_POS again lies below X_POSITIVE, then after acknowledgment, the output is set again.
ACK_REQ	Bool	**Acknowledgment request** If a fault has occurred, but is no longer active, and can therefore be acknowledged, then this block indicates this using a 1 signal at ACK_REQ.
ERROR	Bool	**Error** This output is set if the block has been incorrectly parameterized – or if the block identifies a potentially dangerous combination of input signals in operation. The output remains set until an error is no longer active and has been acknowledged.
DIAG	Word	**Diagnostic word** Information regarding the function status and errors of the block are output here. (also see the table below)

3.5.1.3 Structure of DIAG

Bit No.	Description	Reset condition
0	Lower end position was fallen below	While retracting, SAFE_POS >= X_NEGATIVE and positive edge at ACK
1	Upper end position was exceeded	While retracting, SAFE_POS <= X_POSITIVE and positive edge at ACK
2	Actual velocity greater than the retraction velocity	SAFE_V <= SLS_THRESHOLD and positive edge at ACK
3	Reserved	---
4	Parameterizing error retraction velocity	0 < VMAX_RELEASE parameterized <= 32767
5	Actual position invalid	Actual position valid again
6	Actual velocity invalid	Actual velocity valid again
7	Reserved	---
8	Reserved	---
9	Reserved	---
10	Reserved	---
11	Reserved	---
12	Reserved	---
13	Reserved	---
14	Reserved	---
15	Reserved	---

3.5.2 Principle of operation

3.5.2.1 Parameterization

1. The user must interconnect the safety-related position actual value of the system to be monitored at SAFE_POS, and at input POS_VALID its validity AND'ed with the valid reference (REFERENCED). Block "F_SAFE_POS" (Chapter 3.2) provides three signals as output.
2. It also behaves the same with inputs SAFE_V and V_VALID, which refer to the safety-related actual velocity.
3. The permitted range for the retraction path is parameterized at inputs X_POSITIVE and/or X_NEGATIVE, specifying the upper and lower limits.
4. VMAX_RELEASE must lie in the range 1 – 32767. The block identifies if values less than 1 or values higher than 32767 are parameterized, and DIAG bit 4 is set. ERROR changes to 1.

The block identifies if not all of the mentioned preconditions are satisfied, and this is signaled as parameterizing error with the appropriately set DIAG bits.

Note
The block only checks the parameterization at the 1st call. This results in a subsequent increased block performance.
As a consequence, reparameterization is not permitted when safety operation is deactivated. The safety program must be regenerated and loaded each time that any of the block operating parameters are changed.

3.5.2.2 Position monitoring

5. As long as the position actual value is valid, and is in the permitted range, the block does not signal an error i.e. outputs ERROR and DIAG supply a 0 signal.
6. If the position actual value is in the permitted range, is however identified as not being valid as a result of POS_VALID = 0, then an error code is also output at DIAG. Until acknowledgment, ERROR remains in the actual state, assuming that no additional faults/errors occur as a result of another active monitoring function. All other outputs maintain their actual state until acknowledgment, or the cancellation of RELEASE. This case, DIAG bit No. 5 is set.
7. As soon as POS_VALID again changes to 1, DIAG bit No. 5 then returns to 0.
8. As soon as SAFE_POS lies outside the parameterized travel range, depending on the direction in which this was exited, X_POSITIVE_OK or X_NEGATIVE_OK is set to 0. In the user interconnection, a stop response should be initiated in the drive.
9. In addition, DIAG bit No. 0 is set for falling below the lower end stop – or DIAG bit No. 1 is set for exceeding the upper end stop. ERROR is set to 1.

Safety notes and instructions
Block F_SAFE_POS signals a 0 signal at POS_VALID via output ERROR = 1. When POS_VALID goes to zero, a user interconnection must initiate a stop response in the drive. All other blocks flag this state using an error code ; to avoid a lot of messages occurring at any one time, ERROR is not again set to a 1. The end stops are no longer monitored. End stop monitoring errors can be immediately acknowledged X_NEGATIVE_OK, X_POSITIVE_OK and SLS_OK are again set. If a 1 signal is again present at POS_VALID, the associated DIAG bit 5 is reset, and the end stops are again monitored.

3.5.2.3 Retracting

10. The block retraction function can be activated using a positive edge at RELEASE in order to travel from the end stop back into the permitted travel range. The velocity parameterized at VMAX_RELEASE is then output at the SLS_THRESHOLD output, and depending on the direction in which the end range was violated, MOVE_POSITIVE_OK or MOVE_NEGATIVE_OK is set to 0, in order to prevent additional motion into the end zone. MOVE_POSITIVE_OK = 0 inhibits motion in the positive direction, MOVE_NEGATIVE_OK = 0 inhibits motion in the negative direction.

Note
The signal for RELEASE must be generated in a safety-related fashion, e.g. by using a key-operated switch or similar device.

11. In order to permit retraction, X_POSITIVE_OK or X_NEGATIVE_OK is reset to 1 with a rising edge at RELEASE; using a suitable user interconnection, a drive stop response should then be deselected.
12. If, during retraction, SAFE_V exceeds the value of VMAX_RELEASE, then SLS_OK changes to 0 and DIAG bit 2 is set.
13. A velocity error can always be acknowledged if the actual velocity SAFE_V again lies below SLS_THRESHOLD.
14. As soon as SAFE_POS has returned to the parameterized, permitted range, after acknowledgment, the axis can again travel at the full velocity; this means that the maximum velocity is again set at SLS_THRESHOLD (maximum possible INTEGER value = 32767).
15. When retracting, if the axis is to travel to the opposite end stop, then the block behaves just the same as for a corresponding end of range violation in normal operation. This means that X_POSITIVE_OK or X_NEGATIVE_OK again changes to 0 and it is only possible to move in the direction away from the end stop.
16. If V_VALID = 0 while SAFE_POS is outside the parameterized travel range, then the retraction velocity can no longer be monitored in a safety-related fashion. Therefore, selection using RELEASE = 1 has no effect, and retraction motion is stopped.
17. To exit this state, V_VALID must be again set to a 1 signal using F_SAFE_POS by acknowledging.
18. Retraction can then be continued. Alternatively, a jump can be made back to the initial state by deselecting RELEASE with subsequent acknowledgment. If SAFE_POS is still outside the parameterized travel range, then the response corresponds to Point 9.

Safety notes and instructions
Block F_SAFE_POS signals a 0 signal to V_VALID via output ERROR = 1. When V_VALID goes to zero, a user interconnection must initiate a stop response in the drive. All other blocks flag this state using an error code; to avoid a lot of messages occurring at any one time, ERROR is not again set to a 1. The retraction velocity is no longer monitored. Active retraction monitoring errors can be immediately acknowledged, SLS_OK is again set. Retraction via RELEASE can be exited normally, MOVE_POSITIVE_OK and MOVE_NEGATIVE_OK are set again. If the axis is not in a valid position range at this time, X_NEGATIVE_OK or X_POSITIVE_OK is withdrawn and ERROR set.
If a 1 signal is again available at V_VALID, then the associated DIAG bits 6 is reset, and retraction that may be presently active is again monitored.

Safety notes and instructions
The parameterization of input V_MAX_RELEASE must be adapted to the permissible safely reduced velocity according to the application-specific risk assessment.

Safety notes and instructions

The interconnection of output MOVE_POSITIVE_OK must match the selection of the SDI drive function for the positive direction. For MOVE_POSITIVE_OK = 0, motion must no longer be possible in the positive direction.

The same applies when interconnecting the MOVE_NEGATIVE_OK output – and inhibiting the negative direction of motion.

It is absolutely essential that the block outputs are connected with the correct signals for controlling the drive.

Otherwise, an impermissible motion toward the end stops is possible, which cannot be identified internally by the block.

3.5.2.4 Acknowledging faults

19. DIAG and ERROR are reset to 0 with a positive edge at ACK, assuming that no other fault is active. The block immediately indicates that it can be acknowledged with a 1 signal at its output ACK_REQ. ACK_REQ is reset to 0 after a positive edge at ACK.

3.6 Fail-safe function block F_ENDZONE

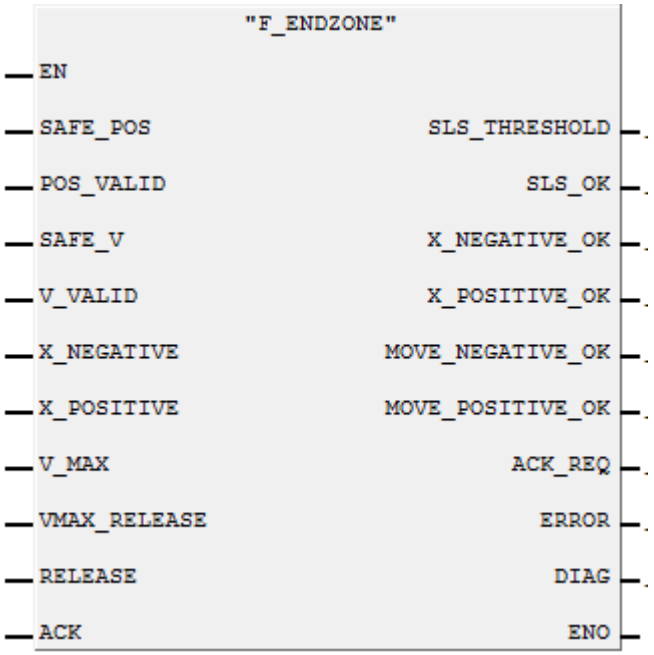
3.6.1 Introduction

The fail-safe function block F_ENDZONE is used to safely monitor the end stops of a travel range. If the monitored system approaches the parameterizable positive or negative end stop, then depending on the actual position along a parameterizable curve, its maximum permissible velocity is limited down to standstill.
The curve is parameterized as table using 10000 individual points in a fail-safe data block assigned to the block.
With the fixed assignment of the fail-safe data block, the block is available in six versions. These only differ regarding the assignment to the fail-safe data block. This means that it is possible to appropriately select the block if the permanently assigned fail-safe data block is already occupied by its own user program or more instances of the block are used. The six versions have an identical function.

FB number	FB name	DB number	DB name
FB210	F_ENDZONE_DB16000	DB16000	F_ENDZONE_DB16000_DB
FB211	F_ENDZONE_DB15999	DB15999	F_ENDZONE_DB15999_DB
FB212	F_ENDZONE_DB15998	DB15998	F_ENDZONE_DB15998_DB
FB213	F_ENDZONE_DB511	DB511	F_ENDZONE_DB511_DB
FB214	F_ENDZONE_DB510	DB510	F_ENDZONE_DB510_DB
FB215	F_ENDZONE_DB509	DB509	F_ENDZONE_DB509_DB

If the axis travels beyond the end stop, or the maximum permitted velocity is exceeded, then the block is signaled and, depending on the user interconnection, a stop response initiated.

The block has a retraction logic function so that when an end stop is passed, the axis can be returned to the permitted travel range. The axis can move away from the end stop with a safely-limited low velocity parameterized at the block. The block provides two signals to control the SDI function in the drive to inhibit continued travel towards the end stop.



Note

When using this block, blocks **F_BO_W/BO_W (FC 176) and F_INT_RD/RD_FDB (FC 179)** must be available in the block folder. It is not permissible that these are renumbered!

3.6.2 Connections

All bool variables listed in the following tables are preassigned FALSE, all integer variables are preassigned 0 – and all word variables preassigned W#16#0.

3.6.2.1 Inputs

Name	Data type	Description
SAFE_POS	Int	**Safety actual position [SLU]** is supplied from the F_SAFE_POS block. The machine is stopped if this value violates one of the limit values X_POSITIVE or X_NEGATIVE.
POS_VALID	Bool	**Actual position valid** is supplied from the F_SAFE_POS block. 1: Position is plausible 0: Position is not plausible, e.g. the discrepancy between the two encoders is outside the tolerance band.
SAFE_V	Int	**Safe actual velocity [SLU/mm]** is supplied from the F_SAFE_POS block. If the actual velocity is higher than the upper limit in operation calculated by the block, or higher than the upper limit parameterized at VMAX_RELEASE in the retraction mode, then output SLS_OK is reset and the machine is stopped.
V_VALID	Bool	**Actual velocity valid** is supplied from the F_SAFE_POS block. 1: The velocity is plausible 0: The velocity is not plausible, e.g. the increase of the deviation between the two encoders over time is outside the tolerance band
X_NEGATIVE	Int	**Min. permissible position [SLU]** If the value at input SAFE_POS falls below this limit value, then output X_NEGATIVE_OK is reset.
X_POSITIVE	Int	**Max. permissible position [SLU]** If the value at input SAFE_POS X exceeds this limit value, then output X_POSITIVE_OK is reset.
V_MAX	Int	**Maximum permissible velocity [SLU/mm]** Also see the diagram under Point 7 in Chapter 3.6.3
VMAX_RELEASE	Int	**Retraction velocity [SLU/min]** If the block is in the retraction mode, then this value is output at SLS_THRESHOLD. VMAX_RELEASE must be parameterized in the range 1 <= VMAX_RELEASE <= V_MAX.
RELEASE	Bool	**Retracting** If the permitted position range was exited, then after a positive edge at this input, the axis can be traversed back into the permissible position range with the velocity parameterized at VMAX_RELEASE. While retracting, motion is immediately stopped if a 0 signal is present at this input.
ACK	Bool	**Acknowledging** If a fault occurs in normal operation, then this must be reset using ACK before the system can be restarted. The acknowledgment is only realized with a positive edge at ACK, and has no effect in normal operation.

3.6.2.2 Outputs

Name	Data type	Description
SLS_THRESHOLD	Int	**SLS limit [SLU/min]** The maximum permissible traversing velocity effective at this time is output at this output. This is cyclically calculated in the block using the parameterized ramp function.
SLS_OK	Bool	**Status, SLS limit** 1: SAFE_V is less than/equal to SLS_THRESHOLD 0: SAFE_V has exceeded the SLS_THRESHOLD value. A stop response should be initiated if this output changes to 0.
X_NEGATIVE_OK	Bool	**Status, minimum position** 1: SAFE_POS is greater than/equal to X_NEGATIVE 0: SAFE_POS has fallen below the value of X_NEGATIVE. A stop response should be initiated if this output changes to 0.
X_POSITIVE_OK	Bool	**Status, maximum position** 1: SAFE_POS is less than/equal to X_POSITIVE 0: SAFE_POS has exceeded the value of X_POSITIVE. A stop response should be initiated if this output changes to 0.
MOVE_NEGATIVE_OK	Bool	**Motion permitted in the negative direction** If a 0 signal is available at this output, then it is not permissible that the machine continues to travel in the negative direction. The output is then set to 0, as soon as SAFE_POS assumes values less than X_NEGATIVE. If SAFE_POS again lies above X_NEGATIVE, then after acknowledgment, the output is set again.
MOVE_POSITIVE_OK	Bool	**Motion permitted in the positive direction** If a 0 signal is available at this output, then it is not permissible that the machine continues to travel in the positive direction. The output is then set to 0, as soon as SAFE_POS assumes values greater than X_POSITIVE. If SAFE_POS again lies below X_POSITIVE, then after acknowledgment, the output is set again.
ACK_REQ	Bool	**Acknowledgment request** If a fault has occurred, but is no longer active, and can therefore be acknowledged, then this block indicates this using a 1 signal at ACK_REQ.
ERROR	Bool	**Error** This output is set if the block has been incorrectly parameterized – or if the block identifies a potentially dangerous combination of input signals in operation. The output remains set until an error is no longer active and has been acknowledged.
DIAG	Word	**Diagnostic word** Information regarding the function status and errors of the block are output here.

3.6.2.3 Structure of DIAG

Bit No.	Description	Reset condition
0	Lower end position was fallen below	While retracting, SAFE_POS >= X_NEGATIVE and positive edge at ACK
1	Upper end position was exceeded	While retracting, SAFE_POS <= X_POSITIVE and positive edge at ACK
2	Retraction velocity exceeded	SAFE_V <= SLS_THRESHOLD and positive edge at ACK
3	Parameterizing error envelope curve, for residual distance 0, the velocity is not 0	Envelope curve parameterized to according to 3.6.3

4	Parameterizing error retraction velocity	0 < VMAX_RELEASE parameterized <= V_MAX
5	Actual position invalid	Actual position valid again
6	Actual velocity invalid	Actual velocity valid again
7	Reserved	---
8	Reserved	---
9	Actual velocity too high regarding actual position and direction	SAFE_V <= SLS_THRESHOLD and positive edge at ACK
10	Reserved	---
11	Reserved	---
12	Reserved	---
13	Reserved	---
14	Reserved	---
15	Reserved	---

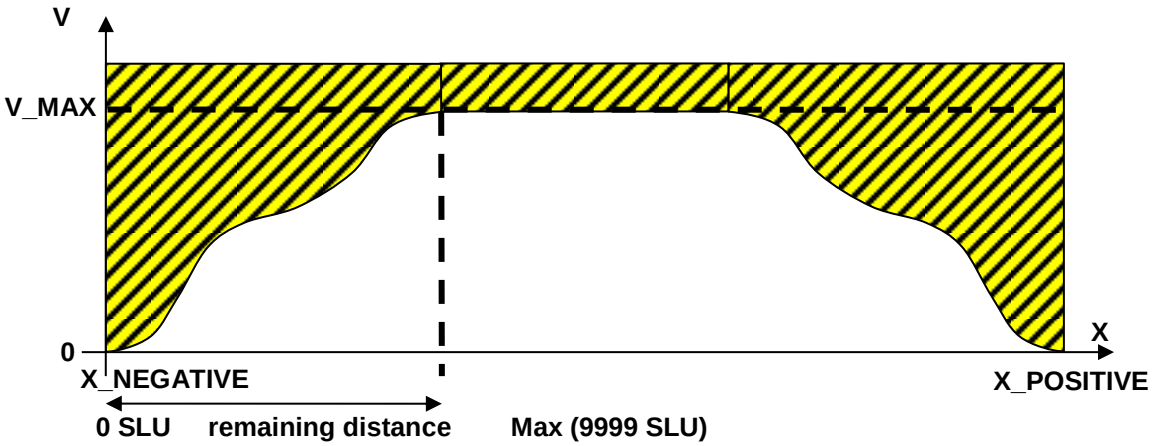
3.6.3 Principle of operation

3.6.3.1 Parameterization

1. The user must interconnect the safety-related position actual value of the system to be monitored at SAFE_POS, and at input POS_VALID its validity AND'ed with the valid reference (REFERENCED). Block "F_SAFE_POS" (Chapter 3.2) provides three signals as output.
2. It also behaves the same with inputs SAFE_V and V_VALID, which refer to the safety-related actual velocity.
3. The lower end stop is parameterized using input X_NEGATIVE and the positive end stop using X_POSITIVE.
4. The velocity envelope curve of the end zone to the monitored is parameterized in a permanently assigned fail-safe data block for the residual distance to an end stop with a maximum distance of 9999 SLU. Correspondingly, the velocity envelope curve is symmetrical in the positive and negative end zones. As a consequence, the fail-safe data block comprises 10000 velocity values, whose index corresponds to the residual distance in SLU, the initial value saved at the index, the maximum permissible velocity at this position in SLU/min.
5. The fail-safe data block must be structured as follows, and the name maintained:

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	v0	INT	0	
+2.0	v1	INT	0	
+4.0	v2	INT	0	
+6.0	v3	INT	0	
+8.0	v4	INT	0	
•				
•				
•				
+19990.0	v9995	INT	0	
+19992.0	v9996	INT	0	
+19994.0	v9997	INT	0	
+19996.0	v9998	INT	0	
+19998.0	v9999	INT	0	
=20000.0		END_STRUCT		

6. The velocity envelope curve must start at index 0 (≙ 0 SLU) with 0 SLU/min, velocity values parameterized higher than the value parameterized at V_MAX, are limited to V_MAX.
7. VMAX_RELEASE must lie in the range 1 – V_MAX. The block identifies if values less than 1 or values higher than 32767 are parameterized, and DIAG bit 4 is set. ERROR changes to 1.



The block identifies if not all of the mentioned preconditions are satisfied, and this is signaled as parameterizing error with the appropriately set DIAG bits.

Safety notes and instructions

The user must validate the parameterized envelope curve himself, and must verify the correct functionality using the appropriate traces and tests (see Chapter 3.1.1).

Note

Requirements regarding the monotony and gradient of the envelope curve depend on the specific application and on the risk assessment.

Note

The block only checks the parameterization at the 1st call. This results in a subsequent increased block performance.
As a consequence, reparameterization is not permitted when safety operation is deactivated. The safety program must be regenerated and loaded each time that any of the block operating parameters are changed.

3.6.3.2 Position and velocity monitoring

8. As long as the position actual value is valid and SAFE_V lies below the parameterized velocity envelope curve, the block does not signal an error i.e. outputs ERROR and DIAG supply a 0 signal.
9. Dependent on SAFE_POS, the associated maximum permissible velocity for this position is output at SLS_ THRESHOLD.
10. If the value at input SAFE_V lies above this limit, and if the system moves towards the end stop, then output SLS_OK is set to 0, ERROR changes to 1, and DIAG bit No. 9 is set. Depending on the user interconnection, a stop response must be initiated in the drive.
11. As soon as SAFE_V returns to the permitted range, i.e. less than the SLS_THRESHOLD, the error can be acknowledged and a 1 signal is output at ACK_REQ.
12. The error can be reset with a positive edge at ACK. ERROR and the corresponding DIAG bits then change back to 0, and SLS_OK again has a 1 signal.
13. If the value at input SAFE_V lies above the permitted velocity, but the system moves away from the end stop, the axis may travel with 100% of velocity, V_MAX is output at SLS_THRESHOLD. As a consequence, no error is signaled here, ERROR and DIAG remain at 0.
14. If a 0 signal is available at POS_VALID, then DIAG bit 5 is set, until acknowledgment, ERROR remains in the actual state, assuming that no additional faults/errors occur as a result of another active monitoring function. All other outputs maintain their actual state until acknowledgment, or the cancellation of RELEASE.

Safety notes and instructions
Block F_SAFE_POS signals a 0 signal at POS_VALID via output ERROR = 1. When POS_VALID goes to zero, a user interconnection must initiate a stop response in the drive. All other blocks flag this state using an error code; to avoid a lot of messages occurring at any one time, ERROR is not again set to a 1. The end stops and the envelope curve are no longer monitored. Maximum velocity V_MAX as well as the validity of the velocity actual value SAFE_V are still monitored. End stop and envelope curve monitoring errors can be immediately acknowledged, X_NEGATIVE_OK, X_POSITIVE_OK and SLS_OK are again set. If a 1 signal is again present at POS_VALID, the DIAG bit 5 is reset, and the end stop and envelope curve monitoring continued.

15. If a 1 signal is again present at POS_VALID, DIAG bit 5 is reset.
16. If a 0 signal is available at V_VALID, then DIAG bit 6 is set, until acknowledgment, ERROR remains in the actual state, assuming that no additional faults/errors occur as a result of another active monitoring function. All other outputs maintain their actual state until acknowledgment, or the cancellation of RELEASE.

Safety notes and instructions
Block F_SAFE_POS signals a 0 signal to V_VALID via output ERROR = 1. When V_VALID goes to zero, a user interconnection must initiate a stop response in the drive. All other blocks flag this state using an error code; to avoid a lot of messages occurring at any one time, ERROR is not again set to a 1. The maximum velocity and the envelope curve are no longer monitored. Active maximum velocity and envelope curve monitoring errors can be immediately acknowledged, SLS_OK is again set. Retraction via RELEASE can be exited normally, MOVE_POSITIVE_OK and MOVE_NEGATIVE_OK are set again. If the axis is not in a valid position range at this time, X_NEGATIVE_OK or X_POSITIVE_OK is withdrawn and ERROR set.
If a 1 signal is again present at V_VALID, the DIAG bit 6 is reset, and the maximum velocity and envelope curve monitoring continued.

17. If a 1 signal is again present at V_VALID, DIAG bit 6 is reset.
18. If the permitted travel range is to be exited, i.e. SAFE_POS values are higher than X_POSITIVE or less than X_NEGATIVE, then the block behaves in a similar fashion to the "F_SLP_MONITOR" block (Chapter 3.5).
    Depending on the direction in which the travel range was exited, X_POSITIVE_OK or X_NEGATIVE_OK is set to 0. In the user interconnection, a stop response should be initiated in the drive.

3.6.3.3 Retracting

19. The block retraction function can be activated by selecting RELEASE. If the system is within the permitted travel range, MOVE_POSITIVE_OK and MOVE_NEGATIVE_OK are reset to 1, at SLS_THRESHOLD, the velocity parameterized at VMAX_RELEASE is output, and the system internally monitors against this value. The velocity envelope curve is still monitored. If this supplies a more restrictive value for the permitted velocity than VMAX_RELEASE, then the permitted velocity is limited to the more restrictive value. If the permitted travel range is exited, the block responds as described under Point 18.
20. The block retraction function can be activated using a positive edge at RELEASE in order to travel from the end stop back into the permitted travel range. The velocity parameterized at VMAX_RELEASE is then output at the SLS_THRESHOLD output, and depending on the direction in which the end range was violated, MOVE_POSITIVE_OK or MOVE_NEGATIVE_OK is set to 0, in order to prevent additional motion into the end zone. MOVE_POSITIVE_OK = 0 inhibits motion in the positive direction, MOVE_NEGATIVE_OK = 0 inhibits motion in the negative direction.

Note
The signal for RELEASE must be generated in a safety-related fashion, e.g. by using a key-operated switch or similar device.

21. In order to permit retraction, X_POSITIVE_OK or X_NEGATIVE_OK is reset to 1 with a rising edge at RELEASE; using a suitable user interconnection, a drive stop response should then be deselected.
22. If, during retraction, SAFE_V exceeds the value of VMAX_RELEASE or the permitted velocity of the opposite end zone, assuming this lies under VMAX_RELEASE, then SLS_OK changes to 0 and DIAG bit 2 is set.
23. A velocity error can always be acknowledged if the actual velocity SAFE_V again lies below SLS_THRESHOLD.
24. As soon as SAFE_POS has returned to the parameterized, permitted range, after acknowledgment, the axis can again travel at the full velocity; this means that the permitted velocity of the envelope curve monitoring is output and monitored at SLS_THRESHOLD.
    ERROR and DIAG change back to 0.
25. If V_VALID = 0 while SAFE_POS is outside the parameterized travel range, then the retraction velocity can no longer be monitored in a safety-related fashion. Therefore, selection using RELEASE = 1 has no effect.
26. To continue retraction, a 1 signal must again be set at block F_SAFE_POS by acknowledging V_VALID.
27. Retraction can then be continued. Alternatively, a jump can be made back to the initial state by deselecting RELEASE with subsequent acknowledgment. If SAFE_POS is still outside the parameterized travel range, then the response corresponds to Point 17.

Safety notes and instructions
The parameterization of input V_MAX_RELEASE must be adapted to the permissible safely reduced velocity according to the application-specific risk assessment.

Warning
The interconnection of output MOVE_POSITIVE_OK must match the selection of the SDI drive function for the positive direction. For MOVE_POSITIVE_OK = 0, motion must no longer be possible in the positive direction.
The same applies when interconnecting the MOVE_NEGATIVE_OK output – and inhibiting the negative direction of motion.
It is absolutely essential that the block outputs are connected with the correct signals for controlling the drive.
Otherwise, an impermissible motion toward the end stops is possible, which cannot be identified internally by the block.

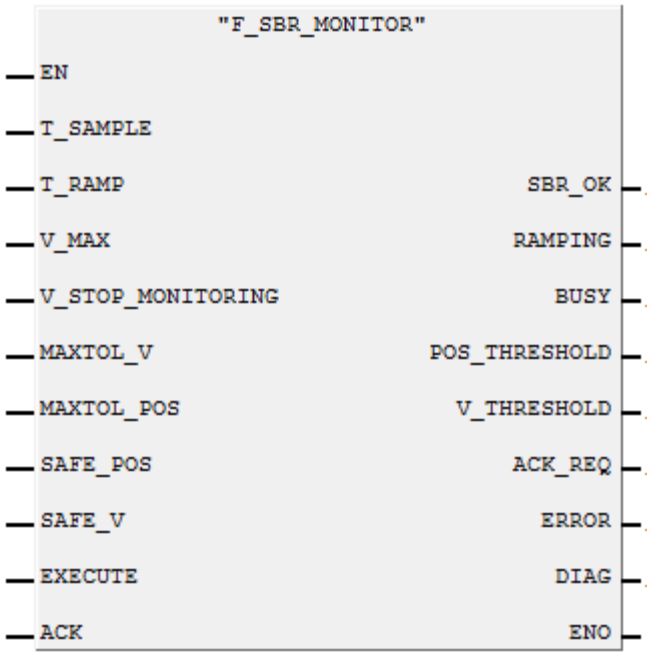
3.6.3.4 Acknowledging faults

28. DIAG and ERROR are reset to 0 with a positive edge at ACK, assuming that no other fault is active. The block immediately indicates that it can be acknowledged with a 1 signal at its output ACK_REQ. ACK_REQ is reset to 0 after a positive edge at ACK.

3.7 Fail-safe function block F_SBR_MONITOR

3.7.1 Introduction

Fail-safe function block F_SBR_MONITOR monitors to ensure that the braking ramp is maintained. If the velocity is not reduced to the down ramp parameterized in the drive, e.g. after SS1 is initiated, then the block provides a signal to initiate STO or to close the brake.



Note

When using this block, block **F_BO_W/BO_W (FC 176)** must be available in the block folder. It is not permissible that this is renumbered!

3.7.2 Connections

All bool variables listed in the following tables are preassigned FALSE, all integer variables are preassigned 0 – and all word variables preassigned W#16#0.

3.7.2.1 Inputs

Name	Data type	Description
T_SAMPLE	Int	**Sampling time [ms]** The block sampling time, i.e. the interval in which the safety program is called (cyclic interrupt OB interval for F-CALL block) is parameterized here in ms T_SAMPLE > 0
T_RAMP	Int	**Ramp-down time [ms]** Here, the identical value in ms for the ramp-down time from maximum velocity down to standstill is parameterized in the drive. In conjunction with V_MAX, the gradient of the down ramp is a calculated from this value. T_RAMP >= 0
V_MAX	Int	**Max. permissible velocity [SLU/mm]** The identical value for the maximum operating velocity as in the drive is parameterized here. In conjunction with T_RAMP, the gradient of the down ramp is a calculated from this value.

V_STOP_MONITORING	Int	**Shutdown threshold for monitoring [SLU/min]** As soon as the actual velocity falls below this threshold, the block can be acknowledged after initiating brake ramp monitoring.
MAXTOL_V	Int	**Velocity tolerance [SLU/min]** max. permissible value that SAFE_V can exceed the configured braking ramp
MAXTOL_POS	Int	**Position tolerance [SLU]** max value that SAFE_POS can exceed the position limit according to the configured braking ramp
SAFE_POS	Int	**Safety actual position [SLU]** is supplied from the F_SAFE_POS block, the signal source is a direct measuring system, which is read-in via the standard program. In the block, the velocity is derived from how this value changes with respect to time. If, after SS1 has been initiated, the block identifies that the axis is not braked along the configured ramp, then the block provides a 0 signal at SBR_OK, which can be used to initiate STO.
SAFE_V	Int	**Safe actual velocity [SLU/mm]** is supplied from the F_SAFE_POS block, the signal source is the motor encoder, which is ready in via the SI part of the drive. If, after SS1 has been initiated, the block identifies that the axis is not braked along the configured ramp, then the block provides a 0 signal at SBR_OK, which can be used to initiate STO.
EXECUTE	Bool	**Start monitoring** The block becomes active with a rising edge at this input; i.e. the brake ramp monitoring is started (taking into account T_DELAY)
ACK	Bool	**Acknowledging** If a fault occurs in normal operation, then this must be reset using ACK before the system can be restarted.

3.7.2.2 Outputs

Name	Data type	Description
SBR_OK	Bool	**Status of the brake ramp monitoring** 1: Brake ramp is maintained or monitoring is not active. 0: Drive does not brake as a minimum with the configured down ramp STO should be initiated if this output changes to 0.
RAMPING	Bool	**Status of the braking ramp** 1: The axis is braking
BUSY	Bool	**Status of the ramp monitoring** 1: Position and velocity limit monitoring active
POS_THRESHOLD	Int	**Position limit value [SLU]** effective limit for ramp monitoring regarding the position change
V_THRESHOLD	Int	**Velocity limit tolerance [SLU/min]** effective limit for ramp monitoring regarding the velocity
ACK_REQ	Bool	**Acknowledgment request** If a fault has occurred, but is no longer active, and can therefore be acknowledged, then this block indicates this using a 1 signal at ACK_REQ.
ERROR	Bool	**Error** This output is set if the block has been incorrectly parameterized – or if the block identifies that the SS1 braking ramp has been violated in operation. The output remains set until an error is no longer active and has been acknowledged.
DIAG	Word	**Diagnostic word**

		Information regarding the function status and errors of the block are output here. (also see the table below)

3.7.2.3 Structure of DIAG

Bit No.	Description	Reset condition
0	SS1 braking ramp not maintained	SAFE_V falls below V_STOP_MONITORING and a positive edge at ACK
1	Parameterizing error T_RAMP: not an integer multiple of T_SAMPLE	The ratio between T_RAMP and T_SAMPLE is an integer number
3	Parameterizing error V_MAX: V_MAX / (T_RAMP / T_SAMPLE) cannot be represented as integer number	The ratio between V_MAX and the number of cycles for the braking ramp specified by T_RAMP and T_SAMPLE is an integer number
4	T_SAMPLE <= 0	T_SAMPLE parameterized > 0
5	MAXTOL_V > V_MAX	MAXTOL_V parameterized <= V_MAX
6	T_RAMP < 0	T_RAMP parameterized >= 0
7	Reserved	---
8	Reserved	---
9	Reserved	---
10	Reserved	---
11	Reserved	---
12	Reserved	---
13	Reserved	---
14	Reserved	---
15	Reserved	---

3.7.3 Principle of operation

3.7.3.1 Parameterization

1. The actual velocity calculated by the F_SAFE_POS block is interconnected at the SAFE_V input.
2. In the block, the gradient of the braking ramp is determined using T_RAMP and V_MAX. To do this, using T_SAMPLE and T_RAMP, the number of cycles is determined, which is required to brake from V_MAX down to standstill. In each cycle, the internally calculated maximum permissible velocity is then. correspondingly reduced by the block.

When parameterizing (assigning parameters) it should be observed that the following relationships must be able to be represented as integer multiple:

T_RAMP / T_SAMPLE

V_MAX / ramp_cycles

Further, the following relationships must exist between the input variables:

MAXTOL_V <= V_MAX

The permissible value ranges of the individual inputs should be taken from the table describing the inputs.

The block identifies if not all of the mentioned preconditions are satisfied, and this is signaled as parameterizing error with the appropriately set DIAG bits.

Safety notes and instructions
Parameterizing V_MAX and T_RAMP must be selected so when it is identified that the permissible travel range has been exited, and with the resulting stop response, when STO is initiated, the axis can always be braked to standstill before the end of the physical travel range.

Note
The block only checks the parameterization at the 1st call. This results in a subsequent increased block performance.
As a consequence, reparameterization is not permitted when safety operation is deactivated. The safety program must be regenerated and loaded each time that any of the block operating parameters are changed.

3.7.3.2 Ramp monitoring

3. Brake ramp monitoring is activated with a rising edge at EXECUTE.
4. If SAFE_V exceeds the internally calculated maximum permissible value, then output SBR_OK changes to 0, ERROR changes to 1 and DIAG bit 0 is set.
5. SBR_OK is also set to 0, if in each cycle SAFE_POS changes by more than the maximum position increase per cycle calculated by the block. This means that the ramp monitoring function has a two-channel structure.
6. In this case, ERROR also changes to 1 and DIAG bit 0 is set.
7. Monitoring is exited as soon as EXECUTE is returned to 0, and the internally calculated velocity ramp has reached a value of 0.
8. A tolerance value for the velocity and position monitoring can be parameterized using inputs MAXTOL_V and MAXTOL_POS. SBR_OK is set to 0 if SAFE_V exceeds the internally calculated ramp + MAXTOL_V, or if the position increase with respect the position at the instant of the selection is greater than the internally calculated maximum value + MAXTOL_POS.

Warning
STO must be immediately initiated for a 0 signal at SBR_OK.

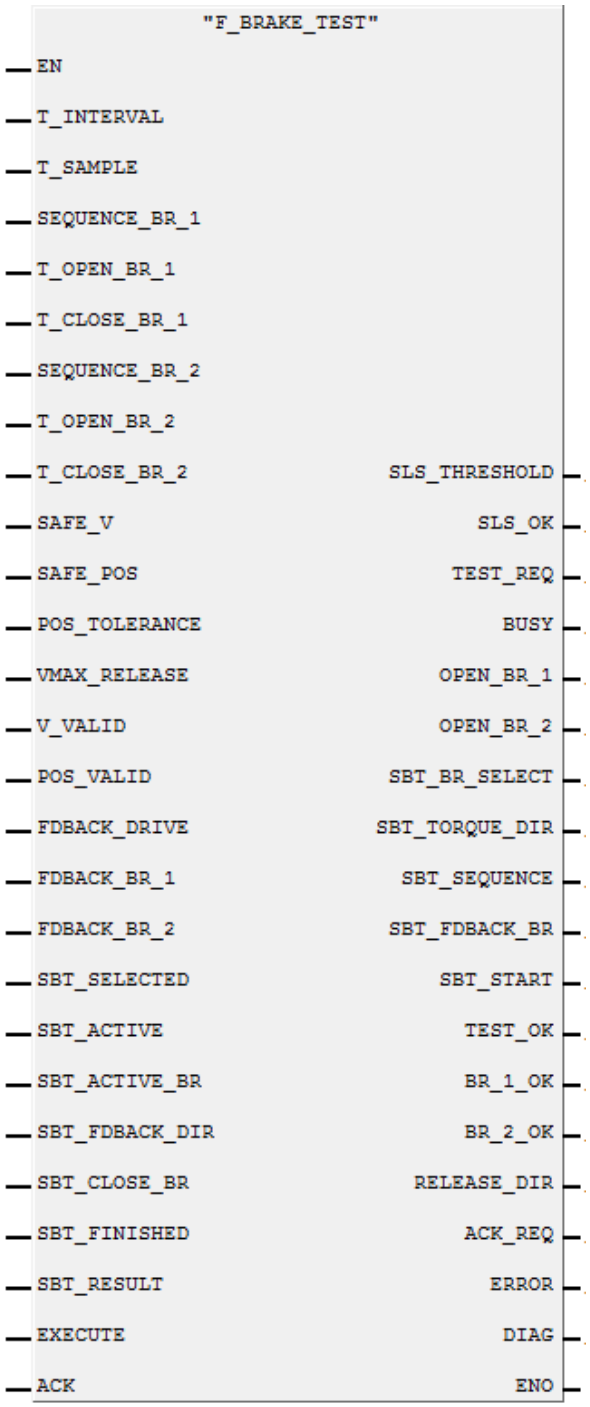
3.7.3.3 Acknowledging faults

9. DIAG and ERROR are reset to 0 with a positive edge at ACK, assuming that no other fault is active.
   The block immediately indicates that it can be acknowledged with a 1 signal at its output ACK_REQ. ACK_REQ is reset to 0 after a positive edge at ACK.
10. After SBR_OK has changed to a 0 signal, i.e. the braking ramp was therefore not maintained, the block can only be acknowledged if the actual velocity at SAFE_V falls below the value at V_STOP_MONITORING. ACK_REQ then changes to a 1 signal.

3.8 Fail-safe function block F_BRAKE_TEST

3.8.1 Introduction

The fail-safe function block F_BRAKE_TEST is used to control the SBT drive function to test a motor holding brake or an external brake.
The specified torque and the test profiles are saved in the SI part of the drive in the Safe Brake Test (SBT). When requested, the block handles the automatic coordination of the parameterized test sequences.
The function of two independent brakes is tested by establishing a torque against the brake that is closed.
If the brake test is unsuccessful, the block supports a retraction logic with SDI and SLS. This means that only travel with reduced velocity is possible, and for an application, e.g. hoisting gear, only downwards.



Note

When using this block, block **F_BO_W/BO_W (FC 176),** block **F_W_BO/W_BO (FC 177)** and block **F_TP/TP (FB 184)** must be in the block folder. It is not permissible that these are renumbered!

3.8.1 Connections

All bool variables listed in the following tables are preassigned FALSE, all integer variables are preassigned 0, all TIME variables are preassigned T#0ms – and all word variables are preassigned W#16#0.

3.8.1.1 Inputs

Name	Data type	Description
T_INTERVAL	Time	**Test interval** The block requests a brake test after this time has elapsed. This is signaled at output TEST_REQ using a 1 signal.
T_SAMPLE	Int	**Sampling time [ms]** The block sampling time, i.e. the interval in which the safety program is called (cyclic interrupt OB interval for F-CALL block) is parameterized here in ms. T_SAMPLE >= 1
SEQUENCE_BR_1	Word	**Configuration parameters** The test pattern to be executed and the brake type for brake 1 defined according to the following scheme via this input: Bit 0: Test with test sequence 1 positive Bit 1: Test with test sequence 1 negative Bit 2: Test with test sequence 2 positive Bit 3: Test with test sequence 2 negative Bit 4: 0: external brake; 1: Motor holding brake
T_OPEN_BR_1	Int	**Brake opening time 1 [ms]** The brake must have been completely opened within this time; otherwise, a read back error is identified and the test is exited as not having been successfully completed. DIAG bit 0 is additionally set for this particular case. T_OPEN_BR_1 >= 1
T_CLOSE_BR_1	Int	**Brake closing time 1 [ms]** The brake must have been completely closed within this time; otherwise, a read back error is identified and the test is exited as not having been successfully completed. DIAG bit 0 is additionally set for this particular case. T_CLOSE_BR_1 >= 1
SEQUENCE_BR_2	Word	**Configuration parameters** The test pattern to be executed and the brake type for brake 2 defined according to the following scheme via this input: Bit 0: Test with test sequence 1 positive Bit 1: Test with test sequence 1 negative Bit 2: Test with test sequence 2 positive Bit 3: Test with test sequence 2 negative Bit 4: 0: external brake; 1: Motor holding brake
T_OPEN_BR_2	Int	**Brake opening time 2 [ms]** The brake must have been completely opened within this time; otherwise, a read back error is identified and the test is exited as not having been successfully completed. DIAG bit 1 is additionally set for this particular case. T_OPEN_BR_2 >= 1
T_CLOSE_BR_2	Int	**Brake closing time 2 [ms]** The brake must have been completely closed within this time; otherwise, a read back error is identified and the test is exited as not having been successfully completed. DIAG bit 1 is additionally set for this particular case. T_CLOSE_BR_2 >= 1

SAFE_V	Int	**Safe actual velocity [SLU/mm]** is supplied from the F_SAFE_POS block. If the brake test was not successfully completed and if the actual velocity is greater than the upper limit parameterized at VMAX_RELEASE, then output SLS_OK is reset and the machine is stopped. DIAG bit 2 is additionally set for this particular case.
SAFE_POS	Int	**Safety actual position [SLU]** is supplied from the F_SAFE_POS block. This is required to monitor standstill during the brake test. If the axis moves by more than the value parameterized at POS_TOLERANCE, then the test is considered not to have been successfully completed and is exited. DIAG bit 3 is additionally set for this particular case.
POS_TOLERANCE	Int	**Threshold for standstill detection [SLU]** If, during a brake test, the axis moves by more than this absolute value, then the test is exited and is considered not to have been successfully completed. DIAG bit 3 is additionally set for this particular case. POS_TOLERANCE >= 0
VMAX_RELEASE	Int	**Retraction velocity [SLU/min]** If the test was not successfully completed, then this value is output at SLS_THRESHOLD until a brake test has been successfully completed. VMAX_RELEASE must be parameterized in the range 1-32767. Otherwise, DIAG bit No. 4 is set
V_VALID	Bool	**Actual velocity valid** is supplied from the F_SAFE_POS block. 1: velocity is plausible 0: velocity is not plausible, e.g. the increase of the deviation between the two encoders over time is outside the tolerance band DIAG bit No. 6 is set if a 0 signal is present here.
POS_VALID	Bool	**Actual position valid** is supplied from the F_SAFE_POS block. 1: position is possible 0: position is not plausible, e.g. the discrepancy between the two encoders is outside the tolerance band. DIAG bit No. 5 is set if a 0 signal is present here.
FDBACK_DRIVE	Bool	**Brake control, normal operation** 0: close brake 1: open brake
FDBACK_BR_1	Bool	**Feedback signal, brake 1** 0: open 1: closed
FDBACK_BR_2	Bool	**Feedback signal, brake 2** 0: open 1: closed
SBT_SELECTED	Bool	**Drive feedback signal – SBT selected** 1: SBT selected 0: function not selected
SBT_ACTIVE	Bool	**Drive feedback signal – SBT status** 1: test running; drive establishes a torque against the closed brake 0: test not active; drive is passive
SBT_ACTIVE_BR	Bool	**Drive feedback signal – active brake** The drive signals back the number of the actually tested brake: 0: brake 1 1: brake 2
SBT_FDBACK_DIR	Bool	**Drive feedback signal – direction of the torque being established** Here, the drive signals back the direction of the torque currently being established:

		0: positive 1: negative
SBT_CLOSE_BR	Bool	**SBT brake control for an external brake** The drive issues the command to open/close the external brake via this input. 0: open ext. brake 1: close ext. brake
SBT_FINISHED	Bool	**Drive feedback signal – test sequence status** 0: test being executed 1: test completed
SBT_RESULT	Bool	**Drive feedback signal – test result** 0: brake faulty 1: brake successfully tested
EXECUTE	Bool	**Start brake test** The brake test is started with a positive edge at this input. After a successful test, the time for the test interval is restarted and output TEST_OK is again set.
ACK	Bool	**Acknowledging** If a fault occurs in normal operation, then this must be reset using ACK before the system can be restarted. The acknowledgment is only realized with a positive edge at ACK, and has no effect in normal operation.

3.8.1.2 Outputs

Name	Data type	Description
SLS_THRESHOLD	Int	**SLS limit [SLU/min]** The maximum permissible traversing velocity effective at this time is output at this output. In normal operation, this is 32767; for an unsuccessful brake test, VMAX_RELEASE is output here. If VMAX_RELEASE is parameterized <= 0 then equivalent value 1 is output here.
SLS_OK	Bool	**Status, SLS limit** 1: SAFE_V is less than/equal to SLS_THRESHOLD 0: SAFE_V has exceeded the SLS_THRESHOLD value. A stop response should be initiated if this output changes to 0.
TEST_REQ	Bool	**Request brake test**
BUSY	Bool	**Test status** 1: test being executed 0: test not selected
OPEN_BR_1	Bool	**Control signal for external brake 1** 1: open brake 0: close brake
OPEN_BR_2	Bool	**Control signal for external brake 2** 1: open brake 0: close brake
SBT_BR_SELECT	Bool	**Drive communication: Brake selection** 0: brake 1 1: brake 2
SBT_TORQUE_DIR	Bool	**Drive communication: Torque preselection** 0: positive 1: negative
SBT_SEQUENCE	Bool	**Drive communication: Select test sequence** 0: sequence 1 1: sequence 2
SBT_FDBACK_BR	Bool	**Drive communication: Status of external brake** 0: open 1: closed
SBT_START	Bool	**Drive communication: start** 1: start the test sequence
TEST_OK	Bool	**Status of the test result**

		0: test not successful 1: test successfully completed
BR_1_OK	Bool	**Status of brake 1** 0: faulty/error 1: OK
BR_2_OK	Bool	**Status of brake 2** 0: faulty/error 1: OK
RELEASE_DIR	Bool	**Status of the motion direction** for an unsuccessful test, is FALSE;
ACK_REQ	Bool	**Acknowledgment request** If a fault has occurred, but is no longer active, and can therefore be acknowledged, then this block indicates this using a 1 signal at ACK_REQ.
ERROR	Bool	**Error** This output is set if the block has been incorrectly parameterized – or if the block identifies a potentially dangerous combination of input signals in operation. The output remains set until an error is no longer active and has been acknowledged.
DIAG	Word	**Diagnostic word** Information regarding the function status and errors of the block are output here. (also see the table below)

3.8.1.3 Structure of DIAG

Bit No.	Description	Reset condition
0	Runtime error T_OPEN/T_CLOSE_BR_1 not maintained	positive edge at ACK, restart the test, reset for a successfully completed test
1	Runtime error T_OPEN/T_CLOSE_BR_2 not maintained	positive edge at ACK, restart the test, reset for a successfully completed test
2	SLS monitoring initiated SAFE_V exceeds V_MAX_RELEASE or V_VALID == 0 while the axis moves for an unsuccessful test	SAFE_V <= VMAX_RELEASE and V_VALID == 1 and a positive edge at ACK
3	No standstill during the test SAFE_POS changes during the test by more than POS_TOLERANCE	positive edge at ACK, restart the test, reset for a successfully completed test
4	Parameterizing error, value range	T_SAMPLE >= 1 and T_OPEN_BR_1 >= 1 and T_CLOSE_BR_1 >= 1 and T_OPEN_BR_2 >= 1 and T_CLOSE_BR_2 >= 1 and POS_TOLERANCE >= 0 and V_MAX_RELEASE >= 1
5	Parameterizing error, integer multiple ratio	T_OPEN_BR_1 / T_SAMPLE can be represented as integer number, and T_CLOSE_BR_1 / T_SAMPLE can be represented as integer number T_OPEN_BR_2 / T_SAMPLE can be represented as integer number, and T_CLOSE_BR_2 / T_SAMPLE can be represented as integer number
6	Non-plausible feedback from SBT	positive edge at ACK, restart the test, reset for a successfully completed test
7	Drive enable missing for active brake test	positive edge at ACK, restart the test, reset for a successfully completed test
8	No safety-related position available for active brake test	positive edge at ACK, restart the test, reset for a successfully completed test
9	Time monitoring: No feedback signal SBT_SELECTED within the monitoring time after selecting SBT	positive edge at ACK, restart the test, reset for a successfully completed test

10	Time monitoring external brake request initiated by SBT	positive edge at ACK, restart the test, reset for a successfully completed test
11	Reserved	---
12	Reserved	---
13	Reserved	---
14	Warning: no safety-related position, SBT not possible	POS_VALID = 1
15	Warning: no safety-related velocity, SBT not possible	V_VALID = 1

3.8.2 Principle of operation

3.8.2.1 Parameterization

When parameterizing (assigning parameters) it should be observed that the following relationships must be able to be represented as integer multiple:

T_OPEN_BR_1 / T_SAMPLE

T_CLOSE_BR_1 / T_SAMPLE

T_OPEN_BR_2 / T_SAMPLE

T_CLOSE_BR_2 / T_SAMPLE

The permissible value ranges of the individual inputs should be taken from the table describing the inputs.

The block identifies if not all of the mentioned preconditions are satisfied, and this is signaled as parameterizing error with the appropriately set DIAG bits.

Note
The block only checks the parameterization at the 1st call. This results in a subsequent increased block performance.
As a consequence, reparameterization is not permitted when safety operation is deactivated. The safety program must be regenerated and loaded each time that any of the block operating parameters are changed.

3.8.2.2 Interface to the SINAMICS S120

The interface between F_BRAKE_TEST and SINAMICS S120 is subsequently described. The communication runs in the standard telegram via status/control word S_ZSW3B(Safety Info Channel status word 3)/S_STW3B (Safety Control Channel control word 3). To do this, SBT selection should be interconnected to "SBT via SCC (p10235)". Interconnection of the signals to the block should be taken from the subsequent table.

3.8.2.2.1 Communication direction F_BRAKE_TEST -> SINAMICS S120

Bit	Meaning	Remarks		Parameters	F_BRAKE_TEST
0	Select brake test	1	Brake test selected	r10235.0	BUSY
		0	Brake test deselected		
1	Start brake test	1	Start brake test requested	r10235.1	SBT_START
		0	Start brake test not requested		
2	Brake selection	1	Test brake 2 selected	r10235.2	SBT_BR_SELECT
		0	Test brake 1 selected		
3	Select direction of rotation	1	Negative direction selected	r10235.3	SBT_TORQUE_DIR
		0	Positive direction selected		
4	Select test sequence	1	Test sequence 2 selected	r10235.4	SBT_SEQUENCE
		0	Test sequence 1 selected		
5	Status of external brake	1	External brake closed	r10235.5	SBT_FDBACK_BR
		0	External brake open		
6…15	Reserved	--	--	--	--

3.8.2.2.2 Communication direction, SINAMICS S120 -> F_BRAKE_TEST

Bit	Meaning	Remarks		Parameters	F_BRAKE_TEST
0	Brake test	1	Brake test selected	r10234.0	SBT_SELECTED
		0	Brake test deselected		
1	Setpoint input, drive/external	1	Setpoint input for the drive	r10234.1	--
		0	Setpoint input, external (control system)		
2	Active brake	1	Test brake 2 active	r10234.2	SBT_ACTIVE_BR
		0	Test brake 1 active		
3	Brake test active	1	Test active	r10234.3	SBT_ACTIVE
		0	Test inactive		
4	Brake test result	1	Test successful	r10234.4	SBT_RESULT
		0	Test error		
5	Brake test completed	1	Execute test	r10234.5	SBT_FINISHED
		0	Test incomplete		
6	External brake request	1	Close brake	r10234.6	SBT_CLOSE_BR
		0	Open brake		
7	Actual load sign	1	Negative sign	r10234.7	SBT_FDBACK_DIR
		0	Positive sign		
8…13	Reserved	--	--	--	--
14	Acceptance test SLP (SE) selected	1	Acceptance test SLP (SE) selected	r10234.14	--
		0	Acceptance test SLP (SE) deselected		
15	Acceptance test mode selected	1	Acceptance test mode selected	r10234.15	--
		0	Acceptance test mode deselected		

3.8.2.2.3 Setting the safe brake test in the inverter

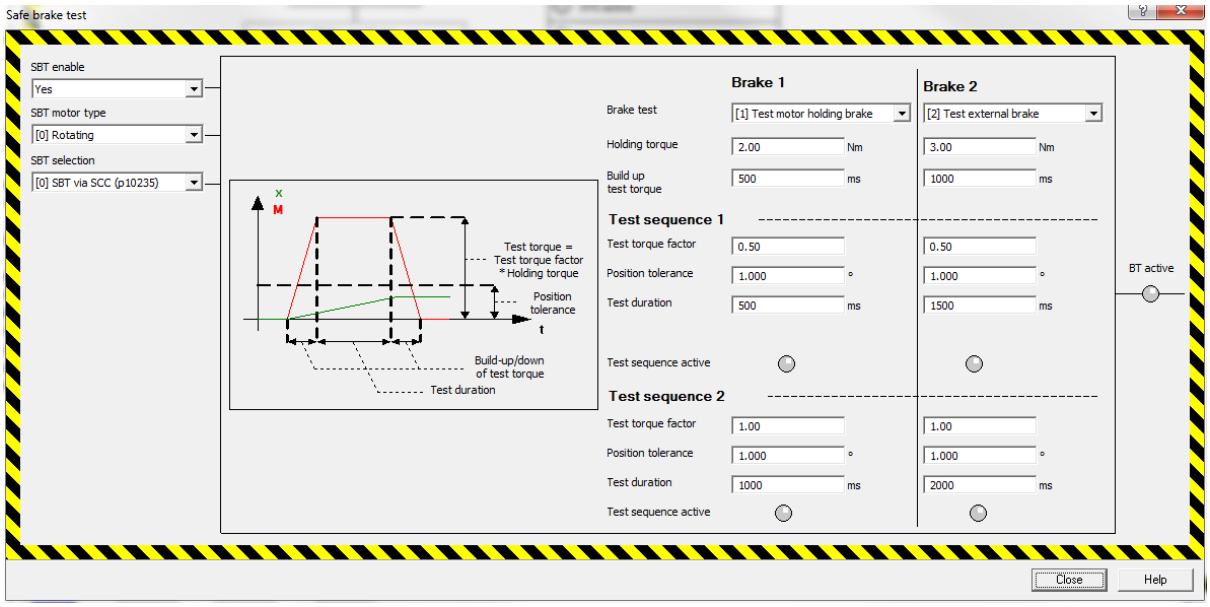


Figure 12: Setting the safe brake test

The sequence of the brakes that are to be tested must match the configuration in the SINAMICS S120 and at the F_BRAKE_TEST. Otherwise, when carrying out the test, the block and the SINAMICS S120 output an error.

The parameters of the test sequences are set in SINAMICS S120, the selection as to which test sequences are to be performed and how is specified at F_BRAKE_TEST.

3.8.2.3 Test sequence and error handling

1. After the time parameterized at T_INTERVAL has elapsed, the block requests a brake test via output TEST_REQ.
   This is started using a rising edge at EXECUTE, BUSY is set to 1.
2. The test sequence for the particular brake is parameterized using input SEQUENCE_BR_1 or SEQUENCE_BR_2.

   Specifying SEQUENCE_BR_1/2 bit-coded:
   - bit 0: Test with test sequence 1 positive
   - bit 1: Test with test sequence 1 negative
   - bit 2: Test with test sequence 2 positive
   - bit 3: Test with test sequence 2 negative
   - bit 4: 0: external brake; 1: Motor holding brake

   Brake 1 is always tested first, followed by brake 2.
3. When the first error occurs, the test is canceled. ERROR then changes to 1, BUSY is reset to 0.
4. At BR_1_OK or BR_2_OK a 0 signals that the test for this brake was not successful, output TEST_OK is set to zero.
5. These signals are only set back to 1 after the test has been successfully completed.
6. As long as the test was not successfully completed, at SLS_THRESHOLD, the velocity parameterized at input VMAX_RELEASE is output, and output TEST_OK has a 0 signal.
7. RELEASE_DIR is set to 0 if the test was successful. By appropriately controlling the drive-side SDI safety function, it is possible to permit further travel only in the safety-related direction; i.e., for a hoisting gear, slowly downwards. As soon as the test was successfully completed, RELEASE_DIR again outputs a 1 signal.
8. If SAFE_V exceeds the value of SLS_THRESHOLD, then SLS_OK changes to 0, and DIAG bit 2 is set.
9. If both brakes were successfully tested, then at SLS_THRESHOLD the maximum value (maximum INTEGER value = 32767) is output for the permissible velocity.
10. DIAG and ERROR are reset to 0 with a positive edge at ACK, assuming that no other fault is active.
11. The block immediately indicates that it can be acknowledged with a 1 signal at its output ACK_REQ. ACK_REQ is reset to 0 after a positive edge at ACK.
12. If a brake test was unsuccessful, then a new test can be started after first acknowledging using a positive signal edge at ACK. To start, a positive edge is required at EXECUTE.
13. If a 0 signal is available at V_VALID, then DIAG bit 15 is set.
14. Further, ERROR changes to 1 and SLS_OK to 0, if the block is presently in the retraction mode.
15. To exit this state, V_VALID must be again set to a 1 signal using F_SAFE_POS by acknowledging.

16. If the test is started using EXECUTE = 1, then the block initially signals this at output BUSY using a 1 signal. This and all additional SBT_... output parameters of the block must be interconnected with the corresponding signals to control the drive-side SBT function.
17. Depending on the parameterized test sequence, in the block, outputs SBT_BR_SELECT, SBT_TORQUE_DIR and SBT_SEQUENCE are switched.
18. If the test was started, then the drive must provide a feedback signal indicating this condition at input SBT_SELECTED. The user must also appropriately establish this signal interconnection.
19. The drive provides a feedback signal about the brake that has just been tested at input SBT_ACTIVE_BR. This feedback is used to check the plausibility. Block sets ERROR as well as DIAG bit 6 if there is a contradiction to the control signals.

Warning
The interconnection of output RELEASE_DIR must match the selection of the SDI drive function for the direction to be inhibited depending on the specific application. For RELEASE_DIR = 0, motion in this direction should no longer be possible.

It is absolutely essential that the block output is interconnected with the correct signal for controlling the drive.

Otherwise, an impermissible motion toward the end stops is possible, which cannot be identified internally by the block.

Safety notes and instructions
The parameterization of input V_MAX_RELEASE must be adapted to the permissible safely reduced velocity according to the application-specific risk assessment.

Safety notes and instructions
Parameter "T_INTERVAL" defines in which cyclic intervals it is necessary to test the brakes. The value to be configured here depends on the specific application, and is also dependent on the risk assessment and the actual hardware architecture of the safety function.

Note
A brake test is requested at each CPU stop-start transition.

Note
Block F_SAFE_POS signals a 0 signal at POS_VALID via output ERROR = 1. All additional blocks, i.e. also F_BRAKE_TEST, indicate this state using an error code; to avoid a flood of messages, ERROR is not again set to a 1 signal, assuming that the block is not performing a brake test at this instant in time. If a brake test is active, and a 0 signal is available at input POS_VALID, then ERROR is also set to 1.

3.8.2.4 Testing an external brake

If a 0 signal is available at SEQUENCE_BR_1/2.BIT4, then an external brake is tested according to the following scheme:

20. If a 1 signal is available at SBT_CLOSE_BR, then depending on the status of SBT_ACTIVE_BR, the block deactivates either OPEN_BR_1 or OPEN_BR_2; i.e. it closes the brake presently being tested. Within the time parameterized at T_CLOSE_BR_1/T_CLOSE_BR_2 a 1 signal must be present at the feedback signal channel FDBACK_BR_1/ FDBACK_BR_2.
21. If this is not the case, then the test is canceled as described above. ERROR and DIAG bit 0/1 (depending on the brake presently being tested) change to 1.
22. After T_CLOSE_BR_1/T_CLOSE_BR_2 has expired, and with a 1 signal at FDBACK_BR_1/ FDBACK_BR_2, the closed brake is signaled to the drive using SBT_FDBACK_BR = 1; the drive then executes its test profile.
23. During the test, it is monitored as to whether at input SAFE_POS the value changes by a maximum of POS_TOLERANCE. If the change is higher, then the test is canceled as described above. ERROR and DIAG bit 3 change to a 1 signal.
24. Once the drive has exited the test, using a 0 signal at block input via SBT_CLOSE_BR, the command to open the brake is output.
25. A 1 signal is then again available at output OPEN_BR_1/ OPEN_BR_2.
26. After the time parameterized at input T_OPEN_BR_1/ T_OPEN_BR_2 there must be 0 signal at the feedback signal channel FDBACK_BR_1/ FDBACK_BR_2.
27. If this is not the case, then the test is canceled as described above. ERROR and DIAG bit 0/1 (depending on the brake presently being tested) change to 1.
28. After T_OPEN_BR_1/ T_OPEN_BR_2 has expired, and with a 0 signal at FDBACK_BR_1/ FDBACK_BR_2, the opened brake is signaled to the drive using SBT_FDBACK_BR = 0.
29. If the brake was successfully tested, then the drive signals this using SBT_FINISHED = 1.
30. For a successfully completed test, a 1 signal is available at input SBT_RESULT.
31. This test pattern is possibly repeated for the second brake, or, depending on SEQUENCE_BR_2.BIT4, the following test pattern is used for the second brake:

3.8.2.5 Testing a motor holding brake

If a 1 signal is available at SEQUENCE_BR_1/2.BIT4, then a motor holding brake at the drive is tested according to the following scheme:

32. In this mode, the drive directly controls the brake. This means that the drive autonomously executes its test profile; the block ignores SBT_CLOSE_BR.

33. During the test, it is monitored as to whether at input SAFE_POS the value changes by a maximum of POS_TOLERANCE. If the change is higher, then the test is canceled as described above. If the brake was successfully tested, then the drive signals this using SBT_FINISHED = 1.
34. For a successfully completed test, a 1 signal is available at input SBT_RESULT.

3.8.2.6 Test completed

35. If the test is still been performed for brake 2 and the configured sequences for brake 1 have already been performed without any errors, then a 1 signal is available at BR_1_OK; however, BR_2_OK and TEST_OK are still 0.
36. If the test was successfully performed for all configured test sequences, then this is signaled using a 1 signal at output TEST_OK; BR_2_OK then also has a 1 signal.
37. The time monitoring for when the next test is due (T_INTERVAL) is restarted, and then the block resets output BUSY to 0.

3.8.2.7 Acknowledging faults

38. DIAG and ERROR are reset to 0 with a positive edge at ACK, assuming that no other fault is active.
    The block immediately indicates that it can be acknowledged with a 1 signal at its output ACK_REQ. ACK_REQ is reset to 0 after a positive edge at ACK.
39. If a brake test was unsuccessful, then this must first be acknowledged with a positive edge at ACK before a new test can be started using EXECUTE.

3.8.3 Application example for safely controlling external brakes

In the following function example, the external brakes at the F-DO channel A20.0 ("BRAKE1") and A20.1 ("BRAKE2") and the F_BRAKE_TEST block for the brake test should be controlled and safety function STO initiated.
The signal for STO (designated here as "STO_select") is low active, i.e. a 1 means that STO is not active, 0 means that at least one safety function is requesting an STO.
The feedback signals of the brakes are wired to the two standard inputs E1.0 ("FDBACK_BRAKE1") and E1.1 ("FDBACK_BRAKE2"), a 1 single means that the brake is closed, 0 signal means that the brake is open.
In this particular example, 100 ms is used as the monitoring time for opening/closing the brake. This time also depends on the response time for the safety functions specified in the risk assessment.

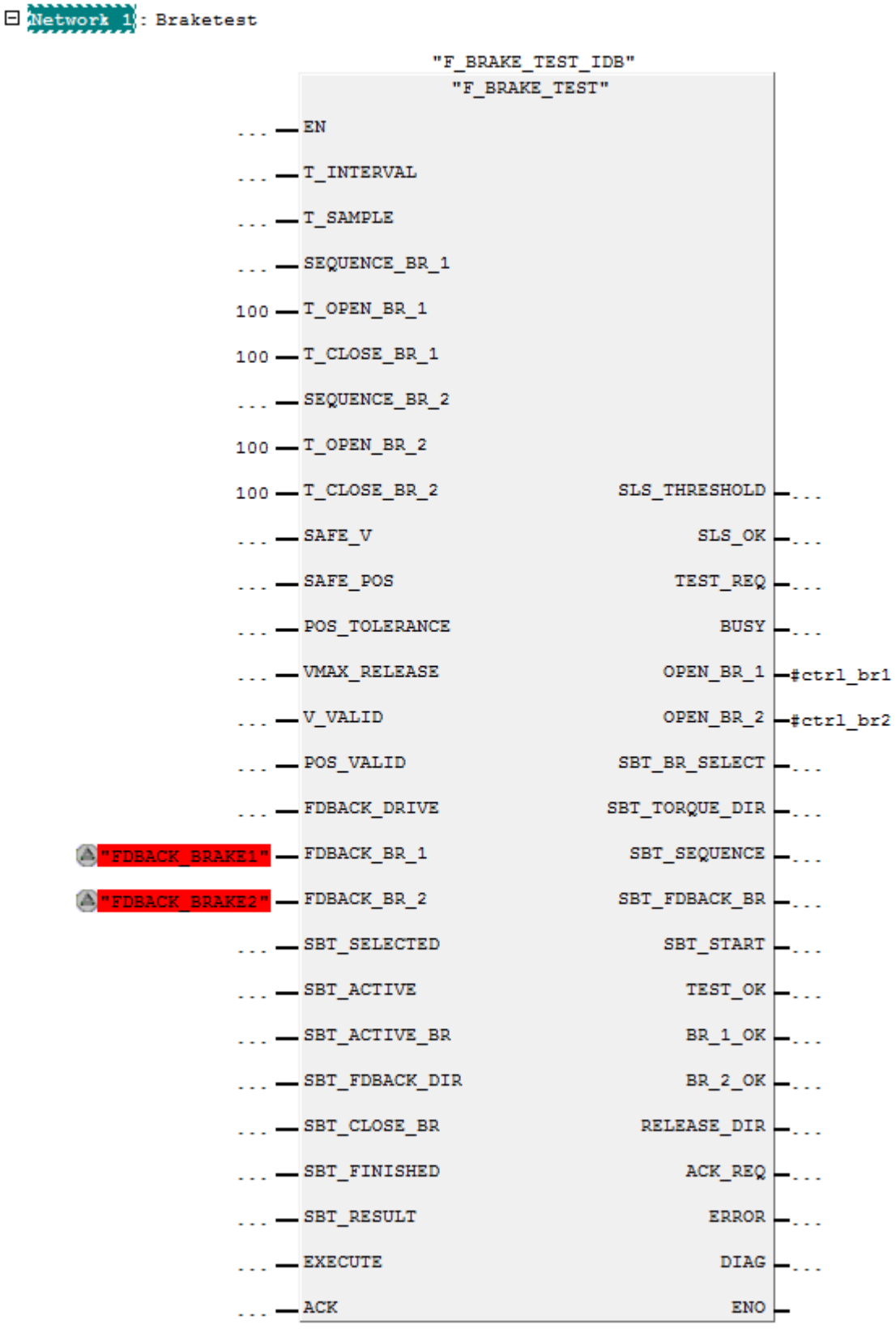
Safety notes and instructions
The parameterization of inputs T_OPEN_BR_x and T_CLOSE_BR_x used in this example as well as FDB_TIME must be adapted to the required safety function response time in the specific application.

It is not permissible that the monitoring time exceeds the specified response time.

For reasons of transparency, in the following code example, only the relevant interconnections for the above description of the application are established.
In order that a program that can be executed is obtained, block F_BRAKE_TEST must be parameterized according to the description in Chapter 3.8.2.

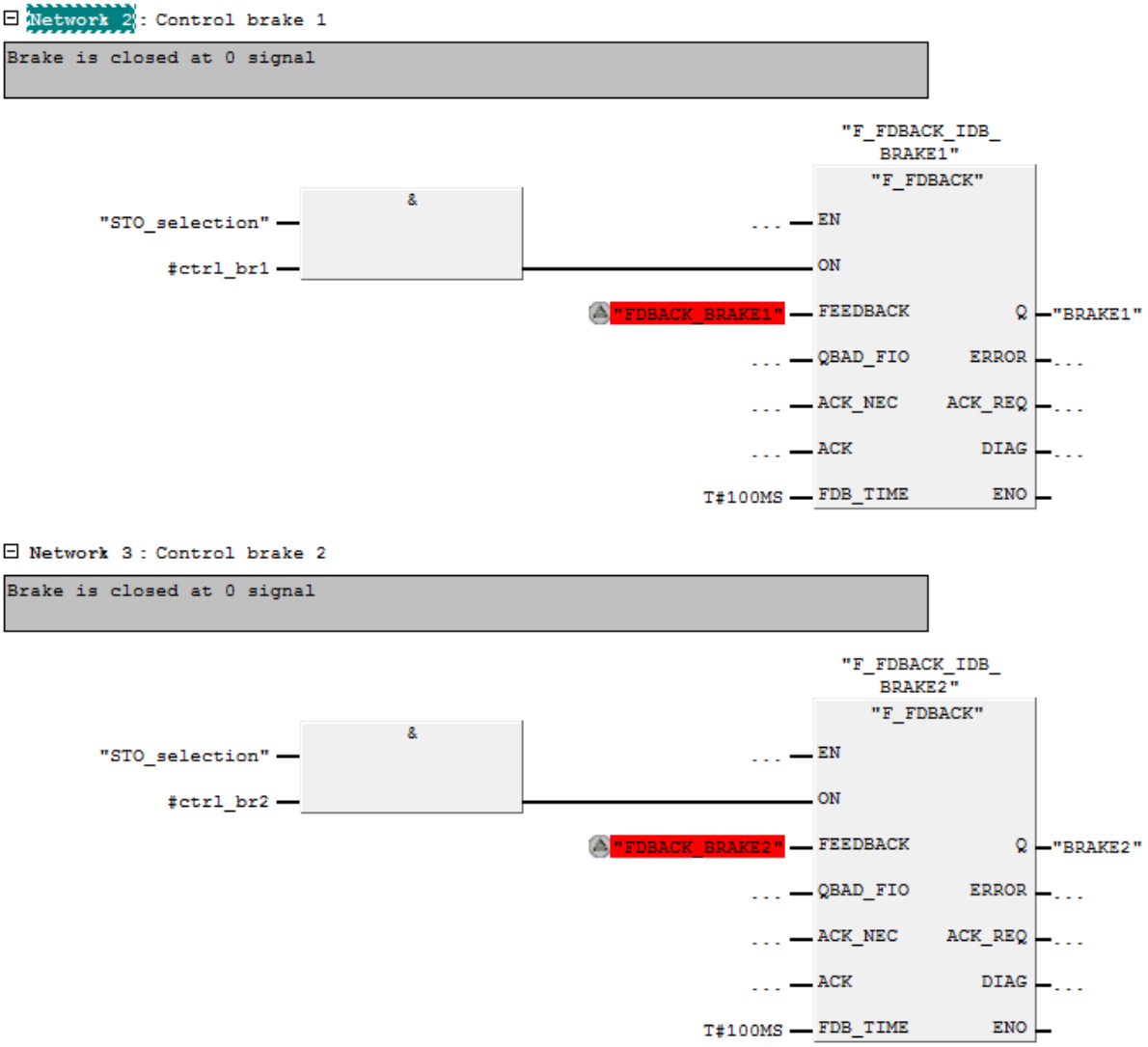
The example is split up into three networks.

Block F_BRAKE_TEST is called in the first network. This does not directly control the brakes, but transfers the control commands to networks 2 and 3 via the temporary variables #ctrl_br1 and #ctrl_br2.



In networks 2 and 3, these signals are fed together with the "STO_select" signal to a F_FDBACK function block. Block F_FDBACK is included in the Distributed Safety (V1) library under number FB216, and implements a feedback circuit monitoring function.
You can obtain additional information about this block via F1 Help, directly in the SIMATIC Manager LAD/FBD/STL Editor.

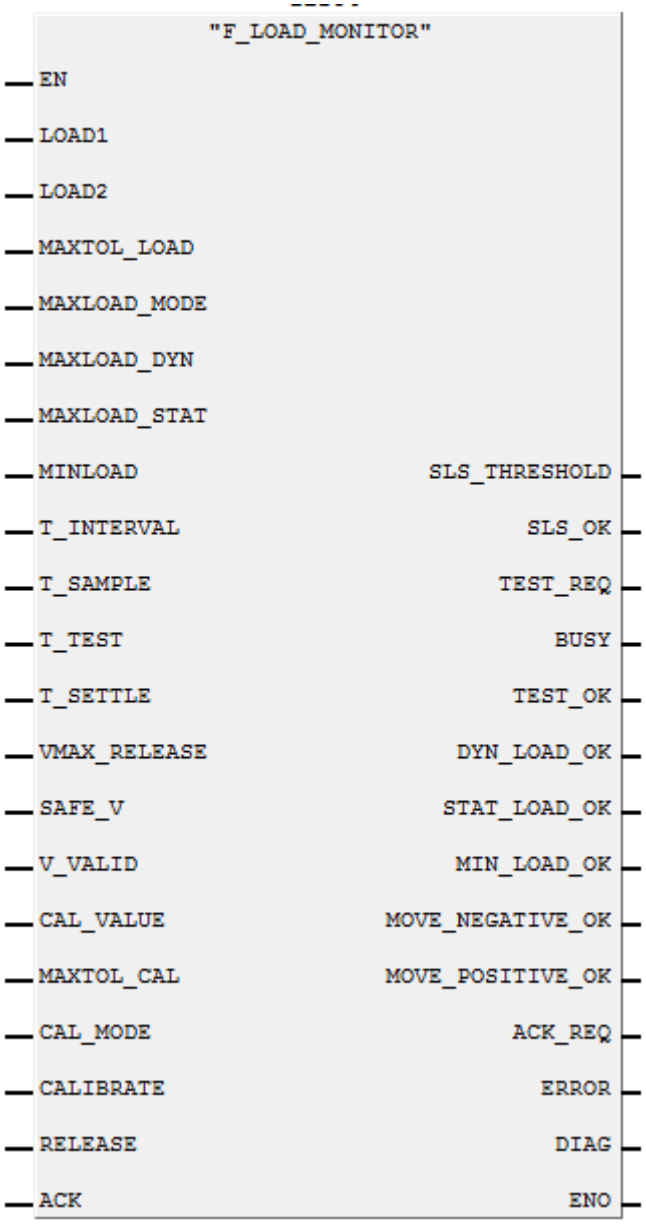
Assuming there are no feedback signal errors of the brake, and the logical combination at the ON input of the F_FDBACK is a 1 signal, then the brakes at output A20.0 ("BRAKE1") and A20.1 ("BRAKE2") are opened.



3.9 Fail-safe function block F_LOAD_MONITOR

3.9.1 Introduction

The fail-safe function block F_LOAD_MONITOR has the function of guaranteeing safety-relevant overload and slack cable detection.
The actual load value is supplied from an analog module and connected measuring source or via the drive torque value.
Retraction logic is available when detecting a slack cable or an overload condition during operation. When a slack cable is detected, retraction is monitored so that retraction is only permissible upwards with a reduced velocity. For an overload condition, retraction is only possible downwards.
The block offers the option of making a distinction between static and dynamic loads, as is the case when quickly lifting loads.
To check that the measuring equipment is functioning correctly, after a parameterizable interval, the block requests that a calibration run is performed.



Note

When using block, block **F_BO_W/BO_W (FC 176)** and block **F_TP/TP (FB 184)** must be available in the block folder. It is not permissible that these are renumbered!

3.9.1 Connections

All bool variables listed in the following tables are preassigned FALSE, all integer variables are preassigned 0, all TIME variables are preassigned T#0ms – and all word variables are preassigned W#16#0.

3.9.1.1 Inputs

Name	Data type	Description
LOAD1	Int	**Load channel 1 [%]** 10000 = 100.00%
LOAD2	Int	**Load channel 2 [%]** 10000 = 100.00%
MAXTOL_LOAD	Int	**Tolerance window load monitoring [%]** 10000 = 100.00% 0 <= MAXTOL_LOAD <= 10000
MAXLOAD_MODE	Bool	**Monitoring mode** 0: monitoring for a steady state load 1: monitoring for a dynamic load
MAXLOAD_DYN	Int	**Maximum dynamic load [%]** 10000 = 100.00% 0 <= MAXLOAD_DYN <= 10000
MAXLOAD_STAT	Int	**Maximum steady state (static) load [%]** 10000 = 100.00% 0 <= MAXLOAD_STAT <= 10000
MINLOAD	Int	**Min. load** 10000 = 100.00% 0 <= MINLOAD <= 10000
T_INTERVALL	Time	**Test interval** The block requests that the measuring equipment is tested after this time has elapsed. This is signaled at output TEST_REQ using a 1 signal.
T_SAMPLE	Int	**Sampling time [ms]** The block sampling time, i.e. the interval in which the safety program is called (cyclic interrupt OB interval for F-CALL block) is parameterized here in ms.
T_TEST	Int	**Test duration [ms]**
T_SETTLE	Int	**Settling time [ms]**
VMAX_RELEASE	Int	**Retraction velocity [SLU/min]** For an overload/underload, this value is output at SLS_THRESHOLD 0 < VMAX_RELEASE <= 32767
SAFE_V	Int	**Safe actual velocity [SLU/mm]** is supplied from the F_SAFE_POS block. If the actual velocity in the retraction mode is greater than the upper limit parameterized at VMAX_RELEASE then output SLS_OK is reset and the machine is stopped.
V_VALID	Bool	**Actual velocity valid** is supplied from the F_SAFE_POS block. 1: velocity is plausible 0: velocity is not plausible, e.g. the increase of the deviation between the two encoders over time is outside the tolerance band
CAL_VALUE	Int	**Calibration value [%]** 10000 = 100.00% relevant for testing the measuring equipment 0 <= CAL_VALUE <= 10000

MAXTOL_CAL	Int	**Calibration tolerance [%]** 10000 = 100.00% 0 <= MAXTOL_CAL <= 10000
CAL_MODE	Bool	**Test mode** 0: Test with constant load value 1: Test with defined load step
CALIBRATE	Bool	**Start the measuring equipment test** The calibration routine is started with a positive edge at this input. After a successful test, the time for the test interval is restarted and output TEST_OK is again set.
RELEASE	Bool	**Retracting** If the block identifies an overload/underload condition, then after a positive edge at this input, the axis can be moved with the velocity parameterized at VMAX_RELEASE in the direction enabled by the block using MOVE_POSITIVE_OK/MOVE_NEGATIVE_OK. While retracting, motion is immediately stopped if a 0 signal is present at this input.
ACK	Bool	**Acknowledging** If a fault occurs in normal operation, then this must be reset using ACK before the system can be restarted. The acknowledgment is only realized with a positive edge at ACK, and has no effect in normal operation.

3.9.1.2 Outputs

Name	Data type	Description
SLS_THRESHOLD	Int	**SLS limit [SLU/min]** The maximum permissible traversing velocity effective at this time is output at this output. In normal operation, this is 32767; when an overload/underload condition is detected, VMAX_RELEASE is output here. If VMAX_RELEASE is parameterized <= 0 then equivalent value 1 is output here.
SLS_OK	Bool	**Status, SLS limit** 1: SAFE_V is less than/equal to SLS_THRESHOLD 0: SAFE_V has exceeded the SLS_THRESHOLD value. A stop response should be initiated if this output changes to 0.
TEST_REQ	Bool	**Request measuring equipment test** 1: T_INTERVALL expired 0: test not required
BUSY	Bool	**Test status** 1: test being executed 0: test not selected
TEST_OK	Bool	**Status of the test result** 0: test not successful 1: test successfully completed
DYN_LOAD_OK	Bool	**Status, dyn. overload** 0: overload detected 1: load OK
STAT_LOAD_OK	Bool	**Status, stat. overload** 0: overload detected 1: load OK
MIN_LOAD_OK	Bool	**Underload status** 0: slack cable detected 1: load OK
MOVE_NEGATIVE_OK	Bool	**Motion permitted in the negative direction (lowering)** If a 0 signal is available at this output, then it is not permissible that the machine continues to travel in the negative direction. The output is then set to 0, as soon as the block detects a slack cable condition.

MOVE_POSITIVE_OK	Bool	**Motion permitted in the positive direction (lifting)** If a 0 signal is available at this output, then it is not permissible that the machine continues to travel in the positive direction. The output is then set to 0, as soon as the block detects an overload condition.
ACK_REQ	Bool	**Acknowledgment request** If a fault has occurred, but is no longer active, and can therefore be acknowledged, then this block indicates this using a 1 signal at ACK_REQ.
ERROR	Bool	**Error** This output is set if the block has been incorrectly parameterized – or if the block identifies a potentially dangerous combination of input signals in operation. The output remains set until an error is no longer active and has been acknowledged.
DIAG	Word	**Diagnostic word** Information regarding the function status and errors of the block are output here.

3.9.1.3 Structure of DIAG

Bit No.	Description	Reset condition
0	Discrepancy error, load monitoring	LOAD1 and LOAD2 within MAXTOL_LOAD and a positive edge at ACK
1	Overload detected	LOAD1 and LOAD2 less than MAXLOAD_STAT or MAXLOAD_DYN (depending on MAXLOAD_MODE) – MAXTOL_LOAD and positive edge at ACK
2	Slack cable detected	LOAD1 and LOAD2 greater than MINLOAD + MAXTOL_LOAD and a positive edge at ACK
3	Parameterizing error, load limits	MINLOAD < MAXLOAD_STAT <= MAXLOAD_DYN
4	Settling process when calibrating takes an inadmissibly long time	Restart the test
5	Inadmissibly high load fluctuation when calibrating	Restart the test
6	Parameterizing error, test times	T_TEST > T_SETTLE > 0, and both times an integer multiple of T_SAMPLE
7	Retraction velocity exceeded	SAFE_V <= SLS_THRESHOLD and positive edge at ACK
8	Parameterizing error, value range	0 < VMAX_RELEASE <= 32767 and 0<= MAXLOAD_DYN / MAXLOAD_STAT / MINLOAD / CAL_VALUE / MAXTOL_LOAD / MAXTOL_CAL parameterized <= 10000
9	Actual velocity invalid	Actual velocity again valid and positive edge at ACK
10	Invalid value range input variables	LOAD1, LOAD2 in the range 0 to 10000 and a positive edge at ACK
11	Reserved	---
12	Reserved	---
13	Reserved	---
14	Reserved	---
15	Reserved	---

3.9.2 Scaling the input variables

The block expects the input of load limits or the actual load values as percentage to two decimal places, i.e. a value of 10000 corresponds to 100%. The user must make this scaling corresponding to the reference variable of the module used.
For F-AI modules, for example, the reference variable is 27648. The F-library Distributed Safety (V1) includes the "F_SCA_I" block specifically for this purpose.
The user must program this scaling himself if hardware with different reference variables is used.

Safety notes and instructions
The user must correctly calculate the load limit values corresponding to the requirements laid down in EN528. The user must appropriately interconnect the calculated limit values at the block.

3.9.3 Principle of operation

3.9.3.1 Parameterization

When parameterizing (assigning parameters) it should be observed that the following relationships must be able to be represented as integer multiple:

T_TEST / T_SAMPLE

T_SETTLE / T_SAMPLE

Further, the following relationships must exist between the input variables:

MINLOAD < MAXLOAD_STAT <= MAXLOAD_DYN

T_TEST > T_SETTLE > 0

The permissible value ranges of the individual inputs should be taken from the table describing the inputs.

The block identifies if not all of the mentioned preconditions are satisfied, and this is signaled as parameterizing error with the appropriately set DIAG bits.

Note
The block only checks the parameterization at the 1st call. This results in a subsequent increased block performance.

As a consequence, reparameterization is not permitted when safety operation is deactivated. The safety program must be regenerated and loaded each time that any of the block operating parameters are changed.

3.9.3.2 Load monitoring

1. If two independent sources are used to measure the force, then these should be interconnected at LOAD1 or LOAD2 after scaling. If one measuring source is sufficient, then this is interconnected at both inputs.
2. If the difference between the two inputs is higher than the value parameterized at MAXTOL_LOAD, then ERROR is set = 1 and DIAG bit is set to 0
3. In addition, the velocity parameterized at VMAX_RELEASE is output at SLS_THRESHOLD.
4. If both values lie within the window that can be parameterized using MAXTOL_LOAD, then using a positive edge at ACK, output ERROR and DIAG bit 0 can again be reset to 0.
5. Input MAXLOAD_MODE can be used to make a distinction between monitoring for steady-state overload (MAXLOAD_MODE = 0) or dynamic overload (MAXLOAD_MODE = 1).
6. For MAXLOAD_MODE =0, as soon as the value at LOAD1 or LOAD2 exceeds the value parameterized at MAXLOAD_STAT, then this error is signaled using a 0 signal at STAT_LOAD_OK.
7. In addition, ERROR is set to 1 and DIAG bit 1 is set

8. For MAXLOAD_MODE =1, as soon as the value at LOAD1 or LOAD2 exceeds the value parameterized at MAXLOAD_DYN, then this error is signaled using a 0 signal at DYN_LOAD_OK.
9. In addition, ERROR is set to 1 and DIAG bit 1 is set
10. As long as one of these errors is active, the velocity parameterized at VMAX_RELEASE is output at SLS_THRESHOLD.
11. The behavior when MINLOAD is fallen below is equivalent

3.9.3.3 Retracting

12. The block retraction function can be activated using a 1 signal at input RELEASE. Further travel in the positive direction is then no longer permissible, the block signals this with a 0 signal at MOVE_POSITIVE_OK. Using an appropriate interconnection with the drive, the user must ensure that in this case, retraction is only possible downwards.
13. In order to permit retraction, DYN_LOAD_OK or STAT_LOAD_OK is reset to 1 with a rising edge at RELEASE; using a suitable user interconnection, a drive stop response should then be deselected.
14. If, while retracting, SAFE_V exceeds the value of SLS_THRESHOLD, then SLS_OK changes to 0.
15. If, in both cases LOAD1 and LOAD2 are again less than the effective limit - MAXTOL_LOAD, then ERROR and DIAG bit 1 can be reset to 0 using a positive edge at ACK.
16. The maximum velocity value is again output at SLS_THRESHOLD. (maximum INTEGER value = 32767)
17. VMAX_RELEASE must lie in the range 1 – 32767; if a value less than 1 is parameterized, the block identifies this and signals it with DIAG bit No. 8. ERROR changes to 1. 1 is then output as equivalent value for the retraction velocity.
18. The retraction velocity can no longer be monitored in a safety-related fashion if during retraction, V_VALID = 0. Therefore, selection using RELEASE = 1 has no effect, and retraction motion is stopped. DIAG bit 9 as well as ERROR change to 1, a 0 signal is present at SLS_OK.
19. To exit this state, V_VALID must be again set to a 1 signal using F_SAFE_POS by acknowledging.

Safety notes and instructions
The signal for RELEASE must be generated in a safety-related fashion, e.g. by using a key-operated switch or similar device.

Safety notes and instructions
The parameterization of input V_MAX_RELEASE must be adapted to the permissible safely reduced velocity according to the application-specific risk assessment.

Warning
The interconnection of output MOVE_POSITIVE_OK must match the selection of the SDI drive function for the positive direction. For MOVE_POSITIVE_OK = 0, motion must no longer be possible in the positive direction.
The same applies when interconnecting the MOVE_NEGATIVE_OK output – and inhibiting the negative direction of motion.
It is absolutely essential that the block outputs are connected with the correct signals for controlling the drive.
Otherwise, an impermissible motion toward the end stops is possible, which cannot be identified internally by the block.

3.9.3.4 Testing the measuring equipment

20. After the time that can be parameterized at T_INTERVAL expires, the force sensor must be tested; the block indicates this using a 1 signal at TEST_REQ.
21. The test is started using a positive edge at CALIBRATE, output BUSY changes to 1.
22. Depending on input CAL_MODE, a constant load or a defined load step is expected as test variable.

Safety notes and instructions
Parameter "T_INTERVAL" defines in which cyclic intervals it is necessary to test the measuring equipment. The value to be configured here depends on the specific application, and is also dependent on the risk assessment and the actual hardware architecture of the safety function.

3.9.3.4.1 Case a): Test with constant load

23. If a 0 signal is available at input CAL_MODE, then within T_SETTLE, the load measured at LOAD1 and LOAD2 must assume the calibration value that can be parameterized at CAL_VALUE – taking into account the tolerance parameterized at MAXTOL_CAL.
24. If this is not the case, then ERROR changes to a 1 signal, and at DIAG bit 4 is set.
25. For the time parameterized at T_TEST, the load value measured at LOAD1 and LOAD2 must not deviate from CAL_VALUE by more than MAXTOL_CAL.
26. If this is not the case, then ERROR changes to a 1 signal, and at DIAG bit 5 is set.
27. After T_TEST expires, and for a valid load value, BUSY is reset to 0 and output TEST_OK is set to a 1 signal.
28. If T_TEST is parameterized <= T_SETTLE, then DIAG bit 5 and ERROR are set to 1.

3.9.3.4.2 Case b): Test with a defined load step

29. If a 1 signal is available at input CAL_MODE, then the load value must go through a load range (stroke). In this case, the signal at LOAD1 and LOAD2 must assume the expected range of CAL_VALUE within T_SETTLE.
30. If this is not the case, then ERROR changes to a 1 signal, and at DIAG bit 4 is set.
31. During T_TEST, the measured range (stroke) must not deviate by more than MAXTOL_CAL from the expected range (stroke) that can be parameterized at CAL_VALUE.
32. If this is not the case, then ERROR changes to a 1 signal and at DIAG bit 5 is set to 1.
33. If, after T_SETTLE expires, the measured signal level at LOAD1 and LOAD2 is not higher than the initial value (before the test stroke was started) by the value CAL_VALUE (taking into account MAXTOL_CAL), then ERROR is also set to 1 and DIAG bit 4 is set.
34. After T_TEST expires and there is a valid value for the load step, BUSY is reset to 0 and output TEST_OK is set to a 1 signal.
35. If T_TEST is parameterized <= T_SETTLE, then DIAG bit 6 and ERROR are set to 1.
36. A successful test is signaled at block output TEST_OK using a 1 signal. BUSY is reset to 0. TEST_OK remains set to 1 until the next time that TEST_REQ changes to 1 or a new test is started

Note
A measuring equipment test is requested at each CPU stop-start transition.

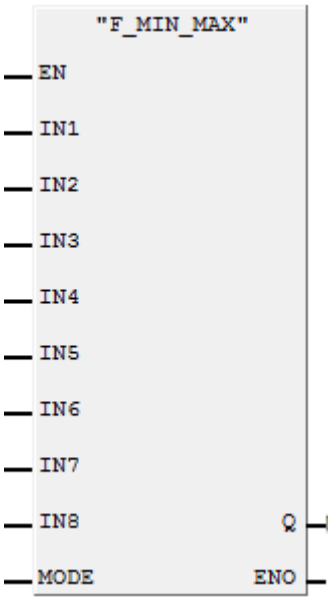
3.9.3.5 Acknowledging faults:

37. DIAG and ERROR are reset to 0 with a positive edge at ACK, assuming that no other fault is active.
38. The block immediately indicates that it can be acknowledged with a 1 signal at its output ACK_REQ.
39. ACK_REQ is reset to 0 after a positive edge at ACK.

3.10 Fail-safe function F_MIN_MAX

3.10.1 Introduction

A minimum/maximum value evaluation from up to 8 INTEGER values is implemented using the F_MIN_MAX fail-safe function. In the context of the additional fail-safe function blocks from the RBG_Failsafe_DS_V5_4 library, this block can be used, for example to select the most restrictive SLS limit that is currently active.



3.10.2 Connections

3.10.2.1 Inputs

Name	Data type	Description
IN1	Int	Operand 1 for evaluation
IN2	Int	Operand 2 for evaluation
IN3	Int	Operand 3 for evaluation
IN4	Int	Operand 4 for evaluation
IN5	Int	Operand 5 for evaluation
IN6	Int	Operand 6 for evaluation
IN7	Int	Operand 7 for evaluation
IN8	Int	Operand 8 for evaluation
MODE	Bool	Selects minimum/maximum evaluation 0: minimum evaluation 1: maximum evaluation

3.10.2.2 Outputs

Name	Data type	Description
Q	Int	Depending on the mode MODE, minimum or maximum value of the 8 inputs

3.10.3 Principle of operation

3.10.3.1 Parameterization

1. The block is implemented as function. This means that when called, all inputs must be interconnected. If a minimum/maximum evaluation is to be performed using less than 8 signals, then the signal sources should be interconnected a multiple number of times so that all of the inputs are assigned at the block.

3.10.3.2 Determining the minimum/maximum value

2. If a 1 signal is available at the MODE input, then the block determines the maximum of the 8 IN1-IN8 inputs. The highest of these up to 8 INTEGER values is made available at output Q.
3. If a 0 signal is available at input MODE, then the block performs a minimum evaluation; the lowest of these 8 INTEGER values is output at Q.

4 Block interaction

4.1 Overview

This chapter explains the most important points that must be taken into account when using the fail-safe function blocks for storage and retrieval machines. Further, the necessary interconnection options between the blocks are shown in the form of examples.

The block package has a modular structure, and can be individually adapted to address the particular application.
The blocks implement a predefined autonomous subfunction. Depending on the particular machine, not all of the blocks in the library will always be required.

If additional functions are required to specifically control your application, you must create these yourself by adding additional fail-safe functions. The signals of these functions are then in turn interconnected with the ASRS blocks.

Safety notes and instructions

The safety-relevant times and the interconnection of inputs and outputs must be parameterized so that they are compliant with the guidelines and directives applicable for the particular plant or system. Further, they must be checked at the plant or system to ensure that the relevant requirements are satisfied.

4.2 Signal flow between the components

The signal flow between the interfaces of the blocks, which can directly interact with one another, is shown in the following overview. The additional inputs that are not connected below, should be parameterized according to the description above. However, for reasons of transparency, they are not interconnected in the following overview as they do not exchange information between the blocks, but are individually parameterized for each block.

4.2.1 Automation task

In the following overview of the block interconnections, a hoisting gear is monitored to ensure that it only moves in a defined range.
Either block F_SLP_MONITOR or block F_ENDZONE is used. If F_ENDZONE is used, then there is also the option of only being able to travel into the end zones with a reduced velocity.
The F_SAFE_POS block supplies the safety-relevant position and velocity required for the blocks mentioned above.
Blocks F_SCALE_DINT and SCALE_DINT scale the 32-bit value of the absolute measuring system to SLU unit.
The hoisting gear is monitored for overload and slack cable using F_LOAD_MONITOR.
The F_BRAKE_TEST block is responsible for testing the functioning of the hoisting gear brakes.
If F_LOAD_MONITOR, F_BRAKE_TEST or F_SLP_MONITOR/F_ENDZONE identify that a limit value has been violated, then a signal to initiate a stop response is also generated, and the SLS threshold for retraction is set to the value that can be parameterized at the block, which is internally monitored.
After SS1 has been selected, block F_SBR_MONITOR monitors as to whether the drive brakes with the configured down ramp. If this is not the case, then a signal to initiate STO is generated.
By AND'ing all of the relevant enable signals of the blocks, the signal to initiate a stop response (e.g. SS1) can be generated for the drive.
For the retraction function of blocks F_ENDZONE, F_LOAD_MONITOR and F_SLP_MONITOR, by AND'ing the corresponding MOVE_POSITIVE_OK / MOVE_NEGATIVE_OK signals, only that direction is permitted that allows the axis to move away from the end zone.

For the F_BRAKE_TEST block, using the RELEASE_DIR output, the hoisting gear can be prevented from traveling upwards if the brake test was unsuccessful.

4.3 Response in the case of an error

If a block error occurs when parameterizing the block, or due to an invalid input assignment as a result of the process, then each library block signals this – with the exception of F_MIN_MAX and (F_)SCALE_DINT – using the output ERROR = 1.
In addition, the library blocks – with the exception of F_MIN_MAX and (F_)SCALE_DINT – have a DIAG output; more precise diagnostics is possible based on the error code output at this DIAG output.

4.4 Block interconnection

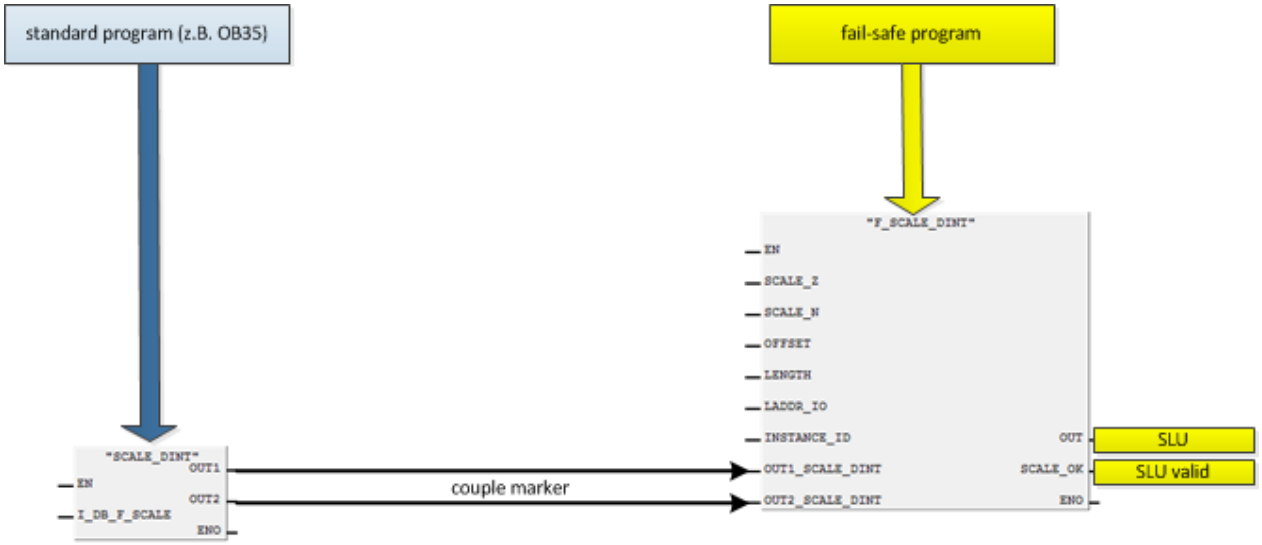


Figure 13: Block interconnection, Part 1

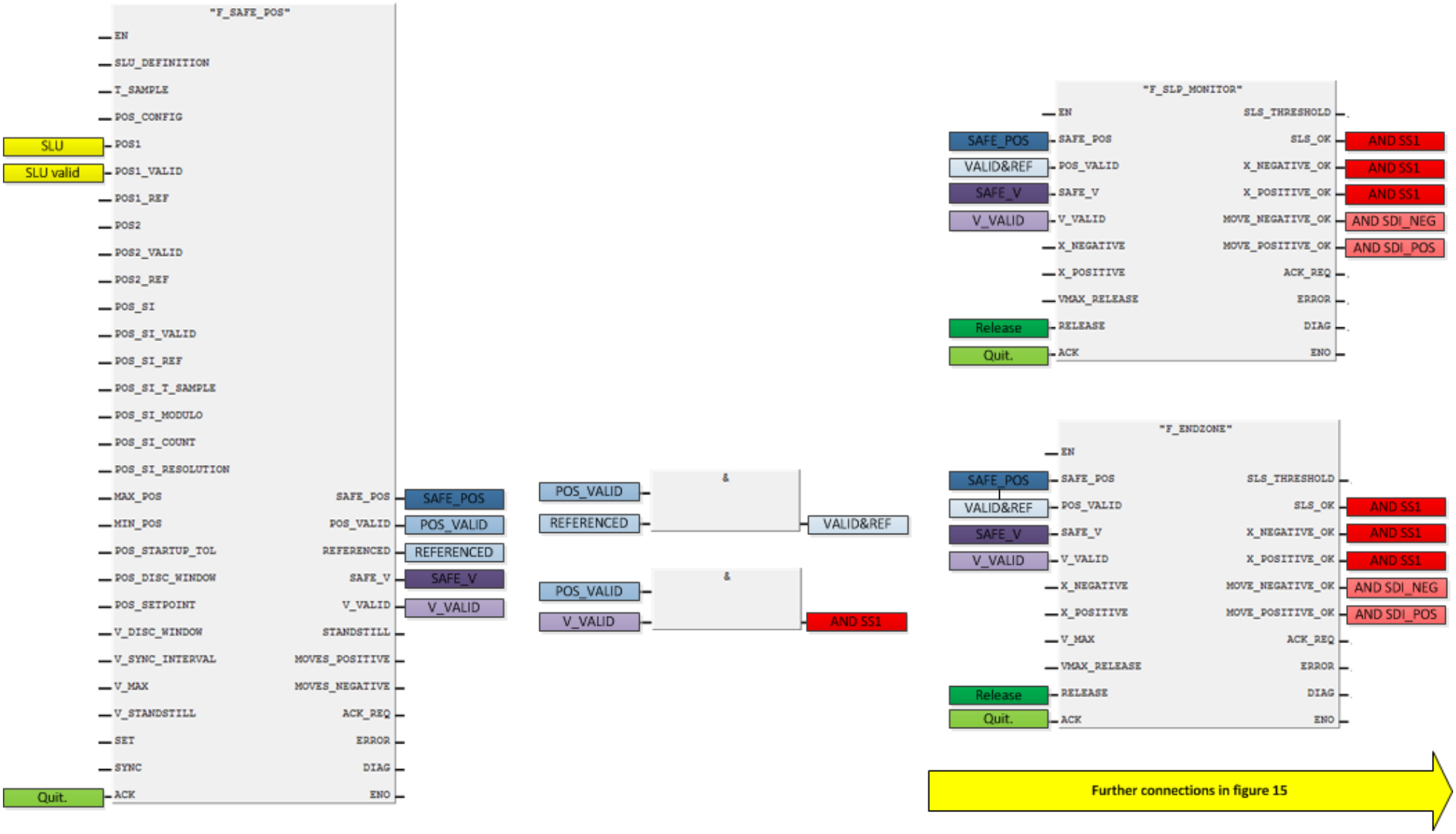


Figure 14: Block interconnection, Part 2

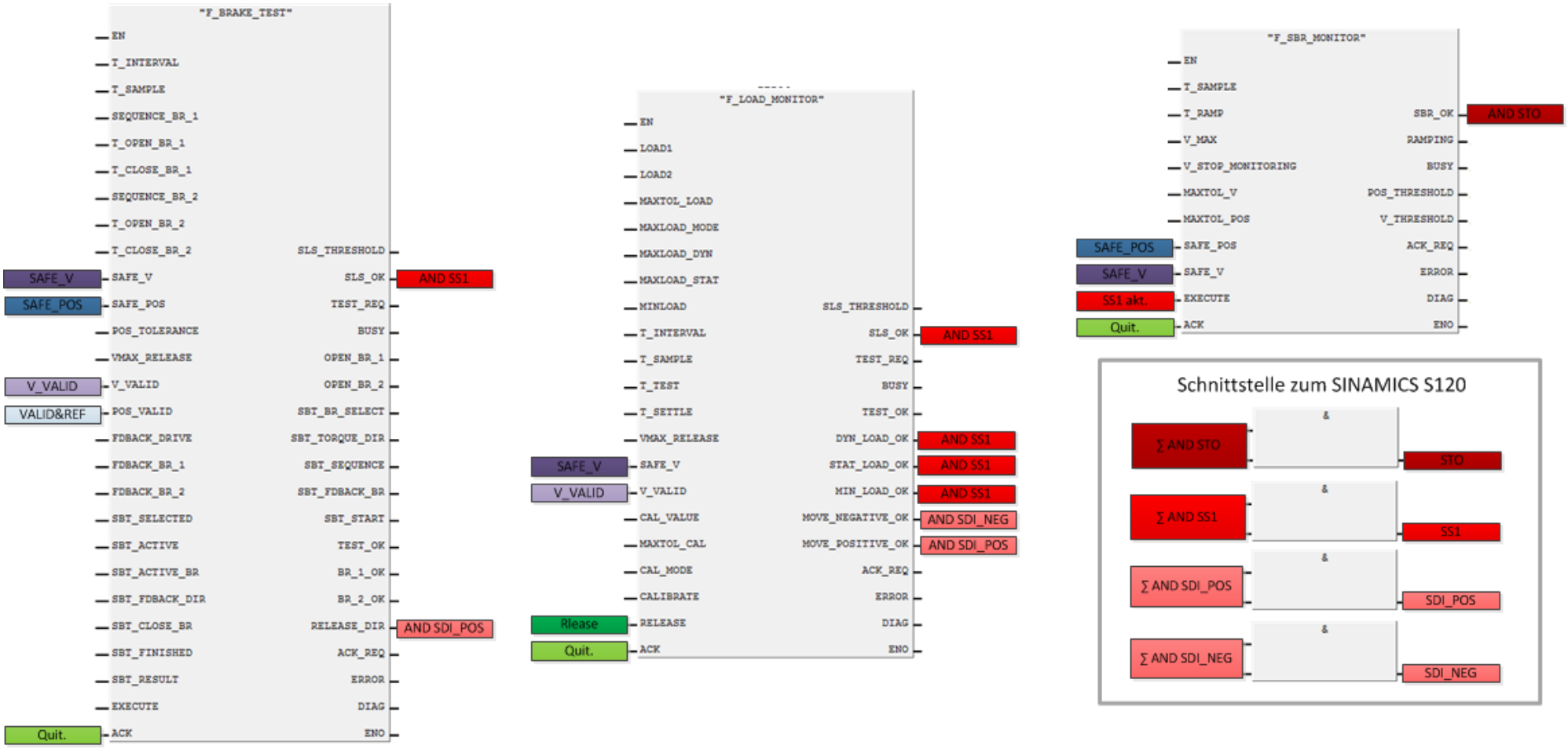


Figure 15: Block interconnection, Part 3

4.4.4 Additionally required blocks

The following blocks, belonging to the Distributed Safety library, are called in the fail-safe function blocks, and must therefore be located in the block folder:

See Chapter 3.1.2

Further, for diagnostics, we recommend that the following organizational blocks are inserted in the block folder, if they are available on the CPU being used:

- Diagnostics alarm – OB 82
- Withdrawing/inserting in operation – OB 83
- Program run error – OB 85
- Module rack failure – OB 86
- Communication error – OB 87
- Programming error – OB 121
- Periphery (I/O) access error – OB 122

4.4.5 Further information

Information about configuring and parameterizing the hardware as well as a description of how to handle STEP7 and the graphic editor (F-FBD or F-LAD) of Distributed Safety and SIMATIC Safety are provided in the manuals listed below:

- S7 Distributed Safety Configuring and Programming
  http://support.automation.siemens.com/WW/view/de/22099875
- SIMATIC Safety - Configuring and Programming
  http://support.automation.siemens.com/WW/view/de/54110126
- SINAMICS S120 Safety Integrated Function Manual
  http://support.automation.siemens.com/WW/view/de/68047679

5 Abbreviations

CPU	Central Processing Unit
CU	Control Unit
DB	Data Block
DINT	Double Integer; 32-bit data type
DO	Digital Output
DS	Distributed Safety
EPOS	Basic positioner; drive function
F-AI	Fail-safe analog module
F-CPU	Fail-safe central processing unit
FMEA	Failure mode and effect analysis
HTL	Type of incremental encoder
HW	Hardware
I-DB	Instance data block
INT	Integer, 16-bit data type
PL	Performance Level
SBT	Safe Brake Test
SDI	Safe Direction
SI	Safety Integrated
SIL	Safety Integrity Level
SIN/COS	Sine-cosine; type of incremental encoder
SLS	Safely-limited speed safety function
SLU	Safe Length Unit
SOS	Safe Operating Stop
SRS	Safety Requirements Specification
SS1	Safe Stop 1
SS2	Safe Stop 2
SSI	Synchronous serial interface; type of absolute encoder
STARTER	Engineering tool for drives
STO	Safe Torque Off

6 Support

Application Center

Please contact the Application Center in D-91056 Erlangen, Germany, if you have any questions that have not been answered regarding how to use the products described in the manual

mailto:tech.team.motioncontrol@siemens.com

or your Siemens contact person in the local office.

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

Training center

Siemens offers a number of training courses for the S7 automation system. Contact your regional training center or the central training center in D -90327 Nuremberg, Germany.

Phone: +49 (0)911 895–3200

http://www.sitrain.com/

SIMATIC Documentation in the internet/Siemens intranet

• You can find the documentation that is free of charge on the Internet at:

http://support.automation.siemens.com

There, you can use the Knowledge Manager provided to quickly find the documentation that you require.

7 Appendix

7.1 Table with the ASRS block runtimes

	IM151-7 F-CPU	IM151-8F CPU	IM151-8F CPU (ab V3.2)	IM154-8F CPU (ab V3.2)	CPU 315F-2 DP	CPU 315F-2 DP (ab V3.0)	CPU 315F-2 PN/DP	CPU 315F-2 PN/DP (ab V3.1)	CPU 317F-2 DP CPU 317F-2 PN/DP CPU 317TF-2 DP	CPU 317F-2 PN/DP (ab V3.1)	CPU 319F-3 PN/DP	CPU 319F-3 PN/DP (ab V3.2)	CPU 414F-3PN/DP (ab V6)	CPU 416F-2 (V4)	CPU 416F-2 (ab V5) CPU 416F-3PN/DP (ab V5)	WinAC RTX F (ab V4.5)	
																EC31-RTX F (Intel Core Duo 1,2 GHz)	IPC427C RTX F IPC477C RTX F (Intel Core2Duo 1,2 GHz)
F_SAFE_POS	8547 us	6217 us	4615 us	2582 us	8816 us	4049 us	5645 us	2727 us	1685 us	1222 us	287 us	430 us	696 us	995 us	632 us	538 us	830 us
F_SLP_MONITOR	1594 us	1174 us	920 us	524 us	1680 us	942 us	1055 us	630 us	330 us	278 us	58 us	83 us	133 us	197 us	119 us	149 us	235 us
F_ENDZONE	6661 us	4934 us	3707 us	2063 us	6903 us	3293 us	4473 us	2220 us	1365 us	988 us	226 us	343 us	532 us	804 us	499 us	464 us	722 us
F_BRAKE_TEST	8048 us	5854 us	4587 us	2608 us	8353 us	4596 us	5309 us	3089 us	1693 us	1376 us	311 us	419 us	656 us	949 us	584 us	635 us	978 us
F_LOAD_MONITOR	5485 us	4008 us	2963 us	1725 us	5696 us	2924 us	3631 us	1984 us	1112 us	890 us	197 us	282 us	439 us	649 us	403 us	437 us	670 us
F_SBR_MONITOR	3805 us	2756 us	2088 us	1137 us	3939 us	1721 us	2501 us	1159 us	758 us	522 us	126 us	189 us	300 us	452 us	280 us	249 us	390 us
F_SCALE_DINT	408 us	290 us	220 us	115 us	425 us	164 us	264 us	111 us	82 us	52 us	13 us	19 us	30 us	49 us	29 us	28 us	44 us
F_MIN_MAX	1098 us	768 us	580 us	302 us	1131 us	433 us	705 us	297 us	228 us	145 us	35 us	53 us	79 us	130 us	78 us	65 us	100 us

8 Notes