PLCopen - Technical Committee 5
–
Safety Software

Technical Specification

Part 1: Concepts and Function Blocks

Version 1.0 – Official Release

Date: Jan. 31, 2006.

Total number of pages: 149

Concepts and Function Blocks for Safety Functions

The following paper is a document created within the PLCopen Technical Committee 5 – Safety Software.
It summarizes the results of the PLCopen Technical Committee meetings, containing contributions of its members:

Dieter Hess	3S Smart Software Solutions, Kempten, Germany
Leo Schratt	3S Smart Software Solutions, Kempten, Germany
Joachim Greis	Beckhoff, Verl, Germany
Jens Sachs	Beckhoff, Verl, Germany
Josef Papenfort	Beckhoff, Verl, Germany
Franz Kaufleitner	B&R, Eggelsberg, Austria
Andreas Pfeiffer	B&R, Eggelsberg, Austria
Michael Huelke	BGIA, Sankt Augustin, Germany
Jochen Ost	Bosch Rexroth, Lohr, Germany
Reinhold Fischer	Bosch Rexroth, Lohr, Germany
Michael Mühlbauer	Bosch Rexroth, Lohr, Germany
Alfred Moeltner	Elau, Marktheidenfeld, Germany
Thomas Janzer	HIMA, Brühl, Germany
John Joosten	Honeywell SMS, Den Bosch, Netherlands
Michael Sperber	Infoteam Software, Bubenreuth, Germany
Martin Gottschlich	KW Software GmbH, Lemgo, Germany
Steffen Schlette	KW Software GmbH, Lemgo, Germany
Guido Beckmann	Lenze Drive Systems, Hamelen, Germany
Lucian Dold	Omron Europe, Nufringen, Germany
Frank Bauder	Omron Europe, Nufringen, Germany
Olaf Ruth	Phoenix Contact, Blomberg, Germany
Torsten Gast	Phoenix Contact, Blomberg, Germany
Johannes Kalhoff	Phoenix Contact, Blomberg, Germany
Gunther Sälzler	Rockwell Automation, Germany
Boris Süssmann	Schneider Electric, Seligenstadt, Germany
Armin Wenigenrath	Schneider Electric, Seligenstadt, Germany
Andreas Höll	Sick, Waldkirch, Germany
Bernard Mysliwiec	Siemens, Nuremberg, Germany
Martin Gottwald	Siemens, Nuremberg, Germany
Oliver Jäger	SEW Eurodrive, Bruchsal, Germany
Gerd Rabe	TÜV Nord, Hamburg, Germany
Erich Janoschek	TÜV Rheinland, Cologne, Germany
Klaus Kemp	TÜV Rheinland, Cologne, Germany
Eelco van der Wal	PLCopen, Zaltbommel, Netherlands

Change Status List:

Version Number	Date	Change Comment
V 0.1	June 13, 2003	Kick-off meeting at Beckhoff, April 29, including notes and MoM
V 0.2	June 27, 2003	Meeting at Siemens, Erlangen, Germany, June 24. Correct pictures added June 27.
V 0.3	September 30, 2003	Meeting at TÜV Nord, Hamburg, Germany
V 0.4	November 4, 2003	Meeting at Bosch Rexroth
V 0.5	January 26, 2004	Meeting at BIA
V 0.6	March 8 & 9, 2004	Results of the meeting at Beckhoff, Frankfurt
V 0.7	April 1 & 2, 2004	Results of the meeting at Infoteam Software, Bubenreuth
V 0.8	June 23, 2004	Results of the meeting at Bosch Rexroth, June 17 & 18, 2004
V 0.9	July 9	Results of the meeting at Sick AG, July 8 & 9, 2004 - P. Smith
V 0.91	September 2, 2004	Example SAFEBOOL data type added, homework Lenze SafeOpStop, partly Hima Estop, Omron example testable safety sensor, Schneider SF_Start, SafeR-educeSpeed Rexroth
V 0.92	October 15, 2004	Results of meeting at Phoenix Contact. Included Homework SF_EnableSwitch from O. Ruth. Added equivalent and complementary FB from A. Höll. Changed background picture 1 – architectural model 3.
V 0.93	January 19, 2005	As result of the meeting in December 2004 at TÜV Nord
V 0.94	February 2, 2005	As result of the meeting Jan. 2005, and editorial work by MH and EvdW. SF_OutControl added, new version of SF_EnableSwitch (v11), and compliance procedure added.
V 0.95	February 25, 2005	Added Homework FBs in new style. Five blocks missing in total.
V 0.96	March 11, 2005	As result of the meeting in March at Beckhoff
V 0.97	March 22, 2005	Included homework done as result of meeting Feb 10 & 11
V 0.98	March 30, 2005	Meeting at Grand Royal, Arnhem, by M. Hülke and E. v. d. Wal
V 0.99	April 7, 2005	Basic for "Release for Comments". In by August 19, 2005
V 0.99A	June 2, 2005	As result of the meeting at Bosch Rexroth and feedback from Phoenix Contact
V 0.99B	June 10, 2005	Included homework as defined at meeting. Basis for intermediate release.
V 0.99C	Sept. 7, 2005	As result of the meeting near Amsterdam. Includes feedback on V 0.99B.
V 0.99D	Sept. 15, 2005	As result of the telephone conference
V 0.99E	October 5, 2005	As result of the check of the English language, and acceptance of most changes by EvdW
V 0.99F	October 13, 2005	Includes proof reading
V 0.99G	October 19, 2005	As result of the meeting at BGIA
V 0.99H	October 19, 2005	As result of working group, and Ruth and Huelke after the meeting at BGIA
V 0.99I	November 1, 2005	Basis for BGIA and TÜVs to add their changes
“	November 3, 2005	M.Huelke added editorial changes and corrected mistakes identified so far. This version is forwarded to BGIA / TÜV
V 0.99K	December 30, 2005	Reviewed version back from BGIA for feedback, discussion and final voting in working group
V 0.99L	January 26, 2006	As a result of the feedback on V 0.99K during meeting at BGIA
V 0.99M	January 31, 2006	As a result of feedback on V 0.99M from BGIA. Basis for Version 1.0. Internal version. Not released
V 1.0	January 31, 2006	Official release

Table of Contents

1. Introduction

The independent association PLCopen, together with its members and external safety-related organizations, has defined safety-related aspects within the IEC 61131-3 development environments. With this, the safety aspects can be transferred to a software tool, which is integrated into the software development tools. This combination helps developers to integrate safety-related functionality into their systems right from the beginning of the development cycle. Also, it contributes to the overall understanding of safety aspects, as well as certification and approval from independent safety-related organizations.
This document mainly focuses on machine controls and is aimed at both:

a) Suppliers of programmable safety controls
b) Users of programmable safety controls

With this addition, PLCopen merged three environments on one development platform: Logic, Motion, and Safety. This is shown in figure 1.

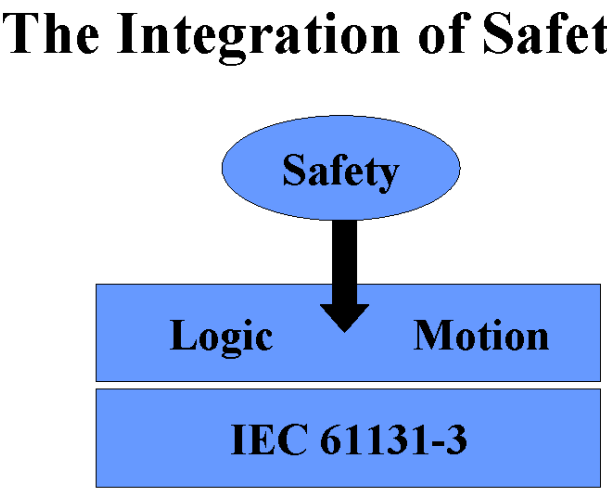


Figure 1: Merging three environments on one platform

1.1. The Rationale of a New Safety Standard

Machine builders are faced with a large set of safety-related standards. This makes it expensive and in some cases unfeasible for machine builders to understand them all fully. Yet in the end they are still responsible for their products and related safety aspects. This risk situation is not very healthy, especially since legislation imposes greater constraints on the equipment suppliers. And their liability increases.
Nowadays there is often a clear separation between the safety-related part and the functional application part. This separation can be made be using different systems for the environments, different tools, and even different people can be involved. This separation often results in the safety aspects being included at the end, and not integrated into the whole system philosophy from the beginning, and often with only limited tests performed. This clearly does not contribute to the overall safety aspects.
Also, the on-going technological innovation now provides safety-approved digital communication buses. This supports the trend away from hard-wired systems towards software-oriented solutions. A parallel can be drawn with the movement away from hard-wired relay logic towards programmable logic controllers, PLCs. Such a trend, of course, involves a change in the mindset. This type of change requires time, widespread support from the industry as a whole, support from educational institutes as well as from certification bodies.
In addition, governmental requirements add to the complexity. For instance, the US-based FDA, Food and Drugs Administration, has set strict regulations that must be complied with. Non-compliance can result in heavy financial penalties, again weakening the sustainability of the organization.

The common basic requirements of a safety application for machine builders within all applicable safety standards are:

- Distinction between safety and non-safety functionalities
- Use of applicable programming languages and language subsets
- Use of validated software blocks
- Use of applicable programming guidelines
- Use of recognized error-reducing measures for the lifecycle of the safety-related software

For users, the effort to fulfill these high requirements should be reduced. This can be done using standardized solutions, which enable typical functionalities to be implemented easily. The standardization of function blocks and integration and support from software tools enables programmers to integrate safety in their applications from the beginning, without adversely affecting their functions and performance, and without adding costs.
To achieve this, PLCopen Committees are working on two levels:

1. Standardization in the look and feel of safety function blocks
2. Integration of standard procedures in the development environment

1: Standardization in the Look and Feel of Safety Function Blocks
In order to help developers use safety-related functionalities, the comfort zone of users must be improved, thus making it easier to accept this way of working. This can be done by standardizing the look and feel of the safety function blocks. In this way the safety functionality can be better recognized and used independently of the applicable system. Re-training is not necessary and the tendency to create dedicated safety functionality is reduced.
In addition, this assists the certification bodies. Specifying and checking the safety software becomes much easier, and therefore quicker, less risky, and less costly.
Providing function blocks at a higher level makes them less dependent on the underlying hardware architecture. Architectures such as hard-wired systems, systems containing safe input and output modules, and network-based systems can be supported with the same function blocks. With this higher-level solution the implementation details can be hidden from users, making the implementation of safety-related software much easier and less costly. This also improves the comfort zone of users.

2: Integration of Standard Procedures
Once the functionalities have been presented in function blocks, the next stage is to determine how to combine them into safety-related programs. At this level the software tool should help the user as much as possible. For this, a new BOOLEAN data type is introduced that is applicable within the safety-related environment, and provides a distinction between safety-related and non-safety-related Boolean variables. This provides the basis for the development tool to identify safety-critical program parts, and guide the user with permissible connections, while preventing incorrect connections. In this way, support can be implemented for the different levels of the various safety standards.
This is combined with a reduction in the functions of the programming languages. In addition, the Function Block Diagram and Ladder Diagram graphical languages are preferred, thus creating program parts that are easier to create and check. This represents a major contribution to the acceptance and use of safety-related functions, thus eliminating several obstacles as they now exist, and are described above, especially for the machine building industry.

1.2. Objectives

The following objectives were identified and met within this Technical Committee:

- Definition of a standard function block (FB) library for standard safety-related functionality
- Combining these FBs with an application program requires an environment that is suitable for safety-related applications. Requirements and restrictions for such an environment are partly dealt with in this standard.
- Accepted concepts and functions by potential certification bodies, providing the basis for certifiable FBs (as objective for Version 1.0)
- Providing an easy-to-use interface to the safety functionality
- Providing a common basis, terminology, and references
- Related to existing safety standards
- Providing a "style guide" for additional/future FBs
- Providing user guidelines/examples
- Application program should be reusable across platforms
- The primary focus of this Technical Committee is safety in machinery
- To include other areas beyond the machine building industry, further additions are expected. These additions can be dealt with in future additions to this document.
- This specification shall be seen as an open framework without hardware dependencies. It provides openness for implementation on different platforms. The actual implementation of the function blocks themselves is outside the

scope of this standard.
- The programming of "safety-related" and "non-safety-related" logic may be possible in the same context.

Based on these objectives, the PLCopen Technical Committee 5 – Safety produced this specification to meet the basic safety requirements. This specification includes:

- Representation of the software architecture
- Definition of the programming languages
- Presentation of safety-related data types
- Definition of language subsets
- Definition of user levels for easy programming and error prevention
- Error handling and diagnostic concept
- Definition of a generic safety-related function block
- The definition of a set of 19 safety-related function blocks
- The definition of a PLCopen compliance procedure combined with the use of the PLCopen Safety logo

This document basically consists of three parts:
1. Reduction in programming languages and functions, to enable safety-related application programs to be created
2. General rules for safety-related function blocks
3. The definition of a set of function blocks with safety-related functions

1.3. Certification

This document provides guidelines, style guides, and basic specifications of function blocks for implementation and use in safety-related environments. The certification bodies confirm by reviewing resulting in a statement to PLCopen that this document, starting with Version 1.0, meets the relevant aspects of IEC 61508 and the related standards and can be used as a part of a specific safety requirement specification. By using the FBs together with the general aspects, the certification procedure of the application becomes much easier and faster. This also applies to the supplier of the software environment with regard to the implementation of this specification. However, this document or a PLCopen certificate does not guarantee that the implementation meets the requirements of the safety standards. Therefore the implementation of the FBs, or their appropriate use, is the responsibility of the supplier and/or user, including safety certification.

In order to meet the requirements set, different kinds of testing and certification are applicable:

1. Testing and certification of the software tool, often part of the control supplier
2. Testing and certification/conformity of the safety application as programmed by the user

Ad 1: Testing and certification of the software tool, often part of the control supplier
The development environment, including the safety-related function blocks, must be certified by the other relevant bodies. In order to be certified, certain regulations such as those described in IEC 61508 are applicable. These requirements are beyond the scope of this document.

Ad 2: Testing and certification/conformity of the safety application as programmed by the user
Within an application, certification includes the safety-related software combined with the infrastructure, such as sensors, switches and actuators, connection schemes, etc. Certification or approvals for these environments are beyond the scope of this document and have to be dealt with by external dedicated organizations.

The use of the PLCopen logo does not give any guarantees as to compliance with or fulfillment of criteria. The use of the logo simply indicates the inclusion of the concepts and guidelines as described in this document, within the relevant software environment, and the availability of this information in more detail in the relevant section of the PLCopen website: www.plcopen.org .

2. General

2.1. Scope

This paper enables conformance with the relevant software-related requirements as specified in IEC 61508 and other basic standards listed in chapter 2.3. As such it provides a basis for the software safety function requirements specification for safety-related function blocks for the implementer, and provides guidance in the software design and coding phases for both the developer/implementer of the FB's and the user of the FB's. This function requirements specification is suitable for applications with required safety integrity levels of SIL 1, SIL 2 and SIL 3. SIL 3 is the highest SIL required for safety of machinery.

The IEC 61508 safety standard includes the description of a safety lifecycle. This contains 16 phases in total, starting with "1. Concept" and ending with "16: Decommissioning or disposal".
This PLCopen document contributes to IEC 61508 "Phase 9: Realisation; Software safety lifecycle 9.1.1; Safety function requirements specification".

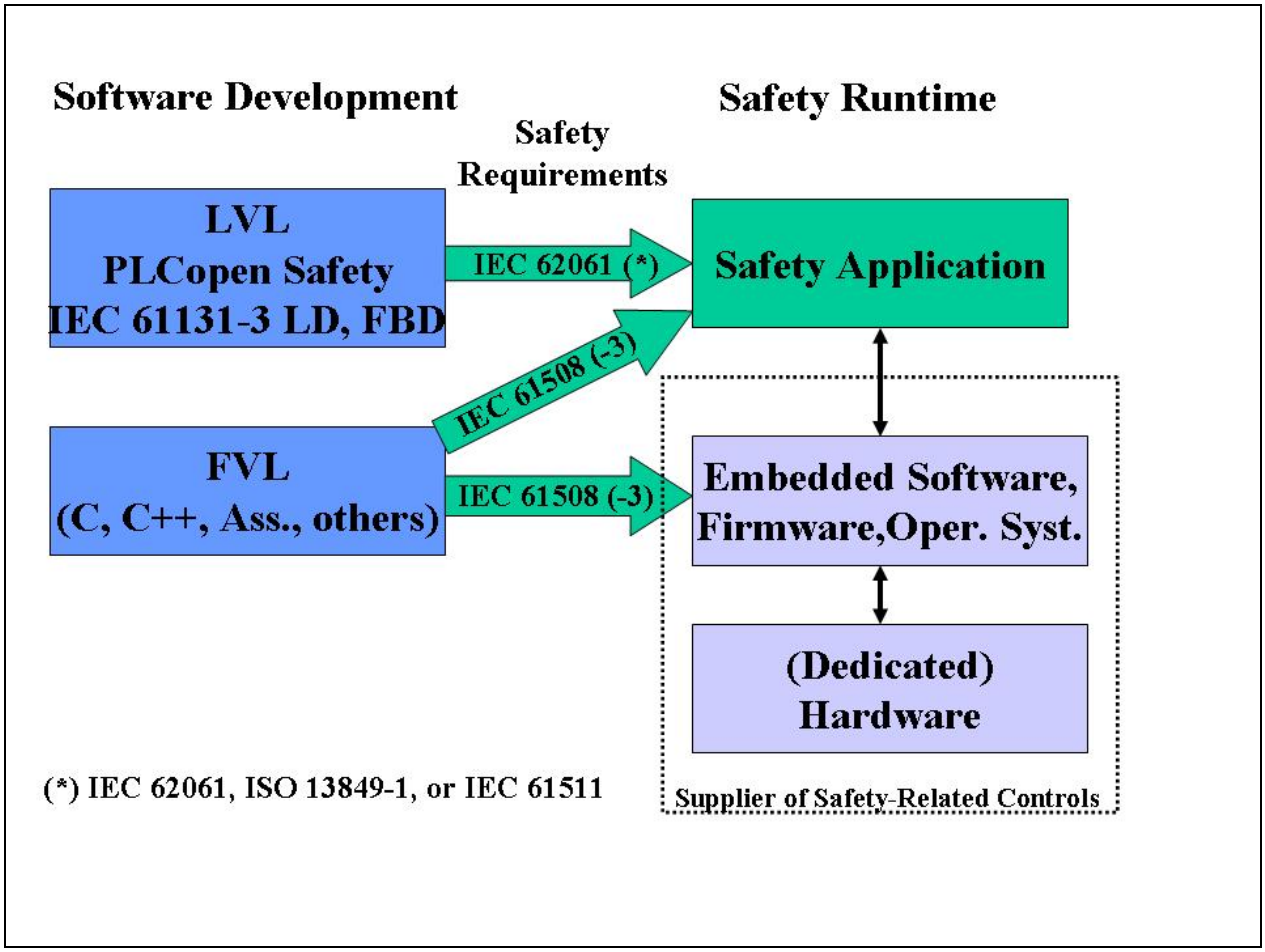


Figure 2: Focus of the work

The relationship between the different standards, the development phases, and the runtime is shown in "Figure 2: Focus of the work". On the left side are the development environments for two levels of software:

1. The embedded software, firmware or operating system, which must comply with the regulations of IEC 61508, especially Part 3. Languages used here can include C, C++, assembler, or others. These are Full Variability Languages (FVL): application-independent languages used by component suppliers for the implementation of (safety) firmware, operating systems or development tools. Rarely used for the safety application itself.
2. The safety application software. If implemented with C, C++, assembler, or others, it is necessary to comply with IEC 61508 as above. They are again based on Full Variability Languages.
   If implemented according to this PLCopen specification, including the reductions in programming languages, instruc-

tions, and certified function blocks, the standards for machinery sector, i.e. IEC 62061 and ISO 13849-1, must be observed by the user at the targeted industries. This simplifies software development and approval dramatically. In this case they can be referred to as Limited Variability Languages (LVL). They are aimed at users in order to create their safety application function blocks. The languages typically used are Ladder Diagram and Function Block Diagram.

The function blocks specified here are not to be treated as a "subsystem element" as defined by IEC 62061, but as IEC 61131-3 function blocks. The IEC 62061 definition of a function block differs from that used in IEC 61131-3 in the sense that it can include hardware, providing safety subsystem functionality.

2.2. Terms and Definitions

AOPD	Active opto-electronic protective device
Basic Level	Programming level aimed at safety-application programmers using the certified (or validated) function blocks.
Categories/Cat.	According to EN 954, discrete level for specifying the safety integrity requirements of the safety functions to be allocated to the safety-related systems.
EDM	External device monitoring signal, which reflects the state transition of an actuator.
ESPE	Electro-sensitive protective equipment
Extended Level	Programming level which extends the basic level with the ability to define custom extensions to the specified set of function blocks.
FBD, LD, SFC, ST, IL	Programming Languages according to IEC 61131-3: FBD = Function Block Diagram, LD = Ladder Diagram, SFC = Sequential Function Chart, ST = Structured Text, IL = Instruction List
Function Block (FB)	According to IEC 61131-3, instance of a function block type, where a function block type is a programmable controller programming language element consisting of: 1) The definition of a data structure partitioned into input, output, and internal variables. 2) A set of operations to be performed on the elements of the data structure when an instance of the function block type is invoked.
Functional application software	General part of the application software, which is not directly related to the safety aspects.
FVL	According to IEC 62061, Full Variability Language: type of language that provides the capability to implement a wide variety of functions and applications
LVL	According to IEC 62061, Limited Variability Language: type of language that provides the capability to combine predefined, application specific library functions to implement the safety requirements specifications
MC-related function	Function relating to motion control applications. To be considered in relation to the set of PLCopen standards "Function Blocks for Motion Control".
Muting	Muting is the intended suppression of the safety function. This is required, e.g., when transporting the material into the danger zone.
NC	Break contact. Normally-Closed contacts disconnect the circuit when the relay is activated; the circuit is connected when the relay is inactive.
NO	Make contact. Normally-Open contacts connect the circuit when the relay is activated; the circuit is disconnected when the relay is inactive.
OSSD	Output Signal Switching Device
Performance Level (PL)	According to ISO 13849-1, discrete level for specifying the safety integrity requirements of the safety functions to be allocated to the safety-related systems, where "PL e" has the highest level of safety integrity and "PL a" has the lowest.
PES	Programmable Electronic System (see IEC 61508)
PFD/PFH	According to IEC 61508-1, probability of failure to perform design function on demand (PFD)/probability of a dangerous failure per hour (PFH).
PLC	Programmable Logic Controller
POU	Program organization units ‘Program', ‘Function', and ‘Function Block', as defined in IEC 61131-3
Process control	Control signal from the functional application for process control.
SAFEBOOL	Data type to identify safety-related BOOLEAN signals.
SAFExxxx	Data type to identify safety-related signals of type xxxx (like SAFEINT).
Safety	Freedom from unacceptable risk (IEC 61508-4: 3.1.8/ISO/IEC Guide 51 second edition (1997 draft)).
Safety application software	Part of application software used to implement safety-related control functions within a safety-related system.
Safety demand	Request to the safety-related function block to set the output signal to the Safe state (FALSE).
Safety Integrity Level, SIL	According to IEC 61508-4, discrete level for specifying the safety integrity requirements of the safety functions to be allocated to the E/E/PE safety-related systems.
System Level	Specific programming level aimed at the implementation of the (specified) function blocks by suppliers. This level is not explained further in this document.

2.3. Relation to Other Standards

The following standards are referenced by this specification:

- IEC 61508-3 (1998-12), Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 3: Software requirements
- EN 954-1 (1996-12), Safety of machinery - Safety-related parts of control systems - Part 1: General principles for design
- ISO/DIS 13849-1 (2004-04), Safety of machinery - Safety-related parts of control systems - Part 1: General principles for design
- ISO 13849-2 (2003-08), Safety of machinery - Safety-related parts of control systems - Part 2: Validation
- IEC 62061 (2005-01), Safety of machinery - Functional safety of safety-related electrical, electronic and programmable electronic control systems
- IEC 60204-1, Ed. 5.0 (2003-07), Safety of machinery - Electrical equipment of machines - Part 1: General requirements
- ISO 12100-1 (2003-11), Safety of machinery - Basic concepts, general principles for design - Part 1: Basic terminology, methodology (replaces EN 292-1)
- ISO 12100-2 (2003-11), Safety of machinery - Basic concepts, general principles for design - Part 2: Technical principles (replaces EN 292-2)
- EN 418 (1992-10), Safety of machinery; emergency stop equipment, functional aspects; principles for design / Note: To be replaced by prEN ISO 13850 (2005-01).
- prEN ISO 13850 (2005-01), Safety of machinery - Emergency stop - Principles for design (ISO/DIS 13850:2005) / Note: Intended as replacement for EN 418 (1992-10)
- EN 61496-1 (2004-05), Safety of machinery - Electro-sensitive protective equipment - Part 1: General requirements and tests (IEC 61496-1:2004, modified)
- CD IEC 61800-5-2 (2005), Adjustable speed electrical power drive systems - Part 5-2: Safety Requirements - Functional safety
- EN 1088 (1995-12), Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
- EN 574 (1996-11), Safety of machinery - Two-hand control devices - Functional aspects - Principles for design
- EN 1037 (1995-12), Safety of machinery - Prevention of unexpected start-up

3. Model

3.1. Software Architectural Model

A software architectural model is provided to describe the typical location of the specified safety function blocks within a machinery control system. This model is as generic as possible, so that existing and upcoming safety control systems can be covered by this model. No safety control hardware architecture should be excluded by this software specification.

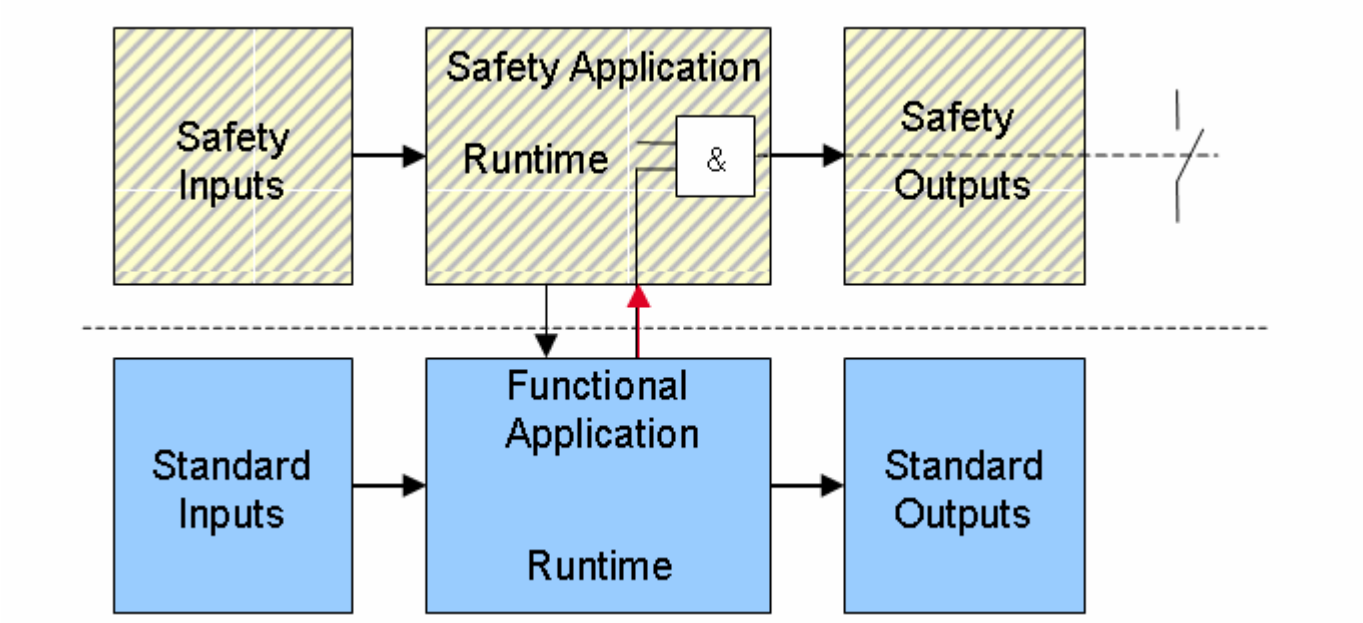


Figure 3: Architectural model

The proposed architectural model differentiates between the functional application part and the safety application part. This is often coupled to two levels of software engineering environments. The objective of PLCopen is to merge these two environments, e.g., a development environment for the functional part with an integrated safety part, including reductions in programming languages and functionality for the safety section.

The two applications could be executed on one device or there could by two or more separate devices which are more or less loosely coupled. The data exchange between the applications, represented by the dashed line, could be via networks, wired I/O or memory transfer within one device. Generally, an important requirement is that there is no undesirable interference from the functional application on the safety application.

On the left side of the model, two sets of inputs are identified, and on the right side two levels of outputs. In the middle, the two environments are shown separately, both coupled to their related inputs and outputs.
The permitted data exchange between the safety and the functional applications is shown in the middle.

- The functional application has read access to the safety inputs and global variables (as indicated by the left arrow).
- The non-safe signals can only be used in the safety application to control program flow and cannot be connected directly to the safe outputs (as indicated by the right arrow and the AND operator).

The same applies to the two sets of outputs.

The model consists of several levels within a safety application, i.e., between the safe inputs, the program with FBs, and the safe outputs. These levels are:

- Safety inputs
- Input level
- Input processing level
- User interface level with function blocks
- Output processing level
- Output level
- Safety outputs

The safe inputs are made available to the software by the system. The details of this are outside the scope of this document. The same applies to the safe outputs. The SAFEBOOL data type is used to identify safe signals, including inputs and outputs within the software – the underlying technology is not part of this specification.

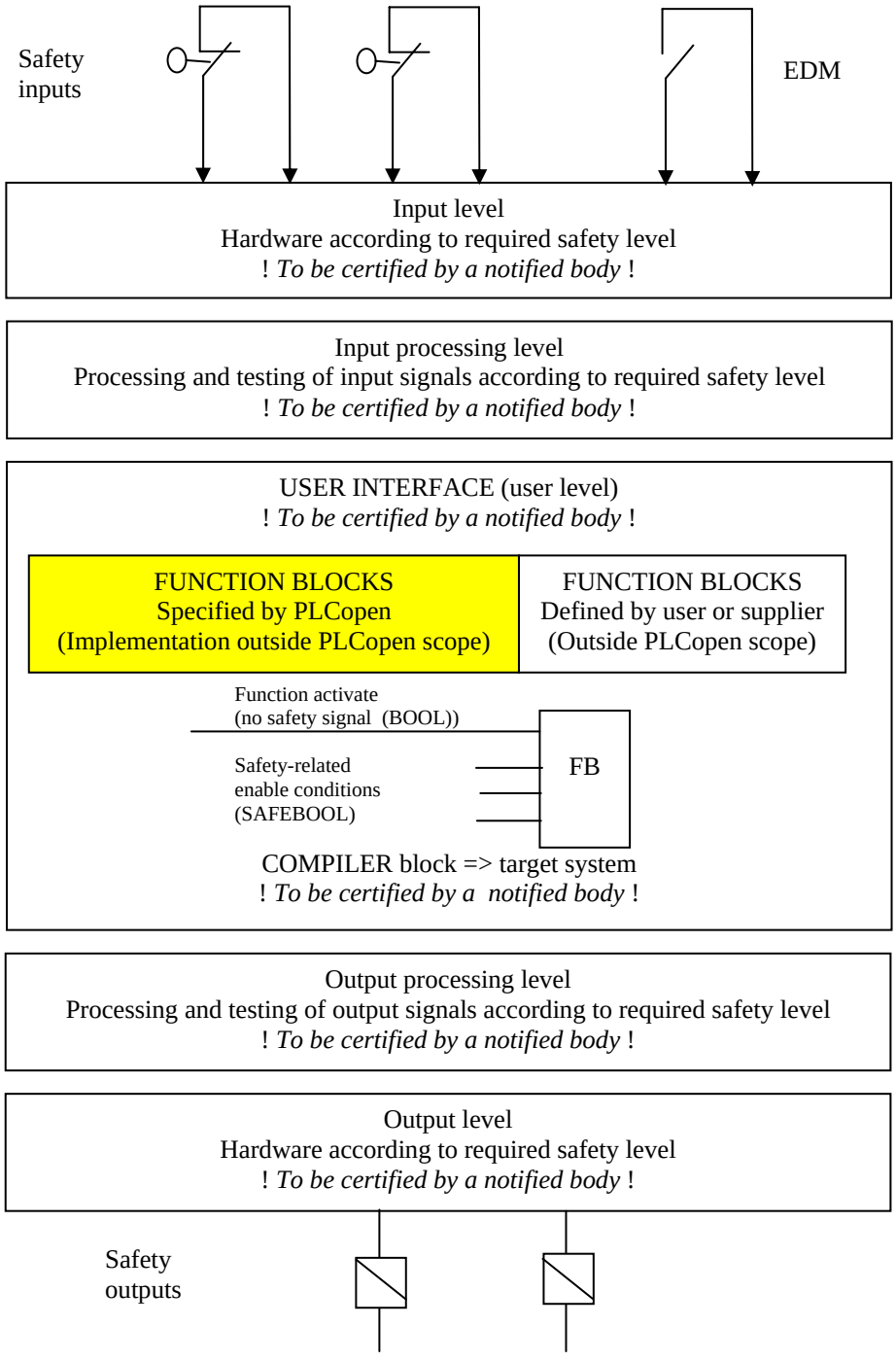


Figure 4: Layers in the architectural model

Notes:

1. The highlighted block in the drawing indicates the scope of this document. The surrounding functionalities are not part of this specification.
2. The number of inputs and outputs do not represent a real application.

3.2. Safe Data Types

In order to differentiate clearly between safety-relevant and standard signals, a new data type with the designation "SAFE" was defined. Thus, the programmer recognizes that the signals are safety-relevant and must be treated with special care. Furthermore, because of this designation the data links can be verified automatically to detect any impermissible links between standard signals and safety-relevant signals. Although the "SAFE" data type cannot guarantee that the signal status is safe (e.g., in the event of incorrectly wired periphery), it is, however, an organizational tool used to minimize errors in the application program. Additionally, when releasing the application program, the safety-relevant signals can be clearly recognized. This simplifies and shortens signal flow verification.
Safe data types are data types applicable within the safety-related environment. These data types shall be used in order to differentiate between safe signals and non-safe signals for ease of validation and certification purposes.

Possible means of supporting safety-related data types in programming environments could be:

- Different means of display/representation of safe data types
- Compiler support of safe data types

SAFEBOOL is a data type that is applicable within the safety-related environment and represents a higher safety integrity level. It differentiates between safety-related and non-safety-related variables. A SAFEBOOL acts as a BOOL within the system, but can contain additional information (attributes) necessary for the safety status and level (could include categories/PL, SILs, PFD/PFH). Such information could be used to calculate the SIL with the programming tool.

The control system guarantees the Safety Integrity Level within the system limits. SAFExx variables are represented as "single-channel", regardless of the internal structure (which can be 1oo1, 1oo2D, 2oo2 or 2oo3). Therefore, such control systems, which execute FB’s with SAFExx inputs and outputs, are to be certified, especially in respect of the generation of SAFExx signals.

Essentially there are (at least) two ways to get a SAFEBOOL variable in the application level:

1. The data is provided as a safe data type by the devices, either by the devices themselves or by the operating system or firmware. This can include a safe network.
2. The data is provided by combining safety inputs in the application itself (such as two safe single-channel inputs).

The safe value for SAFEBOOL must be FALSE. Application designers must ensure that all SAFEBOOL variables result in safe behavior when set to FALSE. SAFEBOOL variables are set to FALSE on initialization and following any faults.

3.3. General Recommendations and Constraints

- Program organization recommendation: The safety application program runs only as a single task. The functional application, which can be executed on a separate processor or device, can contain several tasks.
- The safety program shall not be interrupted by the functional application program.
- When the safety application cycle is started, all relevant input data representation is up-to-date and stable during the cycle.
- The safety-related outputs shall not be changed by the functional application alone.
- In the safety program it is recommended that certified function blocks, as defined in this specification, be used. The user can thus achieve a high level of error prevention.
- The safety function blocks shall be applicable in the FBD and LD IEC 61131-3 languages, while the contents of the function blocks can be implemented in any programming language (e.g., IEC 61131-3 ST, C) or even in firmware or hardware. Therefore the contents are not expected to be portable.
- Every POU/FB in the safety application has accessible information that contains the following: author, date of creation, date of release, version, version history, and functional description (including I/O parameters). This information is visible as a minimum during certification, program design, and program modification. Access to this information may vary depending on the type of use, e.g., can be part of the FB or can be referenced to another source like a web server.
- The software tool should provide support for header information in user-defined POUs.

Note: Safety-related systems are based on "negative" logic. For instance, the physical emergency stop switch is normally closed, so a current flows through the circuit. If the switch is engaged, the contact opens, and so the current flow is stopped. ("Idle current" principle or "Ruhestrom-Prinzip" in the German langauge).

4. Reduction in the Development Environment

4.1. Definition of User Levels

This specification differentiates between three levels:

Basic Level:
A fundamental approach is that the safety program only consists of certified function blocks that can be easily "wired" with one another in graphical form. If, in addition to this, the type of connection is limited, a view adapted to modern technology can be produced, which is similar to the discrete wiring of safety components. The programs have a clear structure and can be easily read. Furthermore, the release time of the program is significantly shortened, as it consists of blocks certified in advance.

Extended Level:
In the case of projects, for which the current status of certified function blocks is not sufficient, the user can create the required blocks (or even the program) in the Extended Level. For this, an extended command range is provided. However, the validation of the functionality for these blocks and programs can be considerably more complex and therefore more time-consuming since the programs underlie the whole verification process. If the blocks have been certified / validated, they can be used in the Basic Level together with the advantages described above.

System Level:
The System Level is provided for suppliers of safety controls. The System Level also enables, e.g., implementations in supplier-specific languages. However, the System Level is not part of the specification.

In any case, the different levels are integrated in the programming tool. Together with an access control they can be assigned to different user groups. The principle described above reduces the effort for the user significantly by simplifying the releasing process.

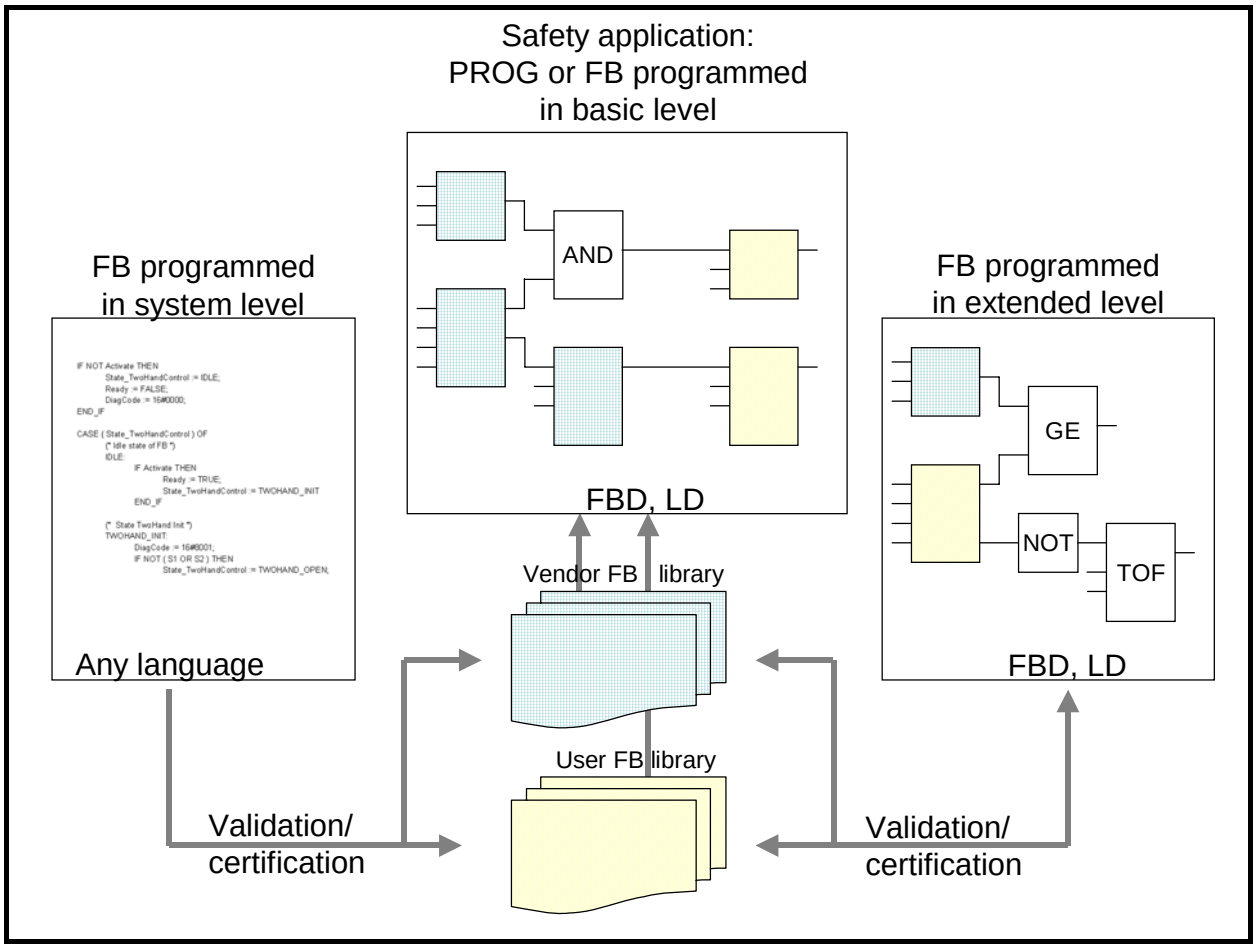


Figure 5: Recommended application scope of the three levels

4.2. Reduction in the Set of Programming Languages

IEC 61508, Part 7, defines a reduction in the preferred programming languages for the different SILs ("Highly Recommended", "Recommended" or "Not Recommended"). Based on this, the preferred languages within this specification are the Function Block Diagram (FBD) and Ladder Diagram (LD) graphical languages with a defined subset of the two. These graphical languages provide a clear overview of the safety program itself, and tool suppliers can implement a much better level of support and guidance for users. This forms the basis for simplified commissioning of the safety-related program. Structured Text (ST), Instruction List (IL), and Sequential Function Chart (SFC) are not dealt with at this time, since higher lifecycle costs are anticipated. More specifically, the testing and validation of applications written in ST or IL is more complex and error-prone then applications written in graphical languages.
This recommendation is specifically aimed at both the Basic Level and the Extended Level. No definitions in terms of languages, functions, and data types are provided here for the System Level (see IEC 61508, Part 7).

4.3. Reduction in Data Types and Declarations

In the tables below, "X" indicates that the item is permitted, "-" indicates that it is not permitted.
Data types other than SAFEBOOL can also have the attribute "safe", e.g., SAFEINT, in order to enable safe data to be tracked automatically.

(See IEC 61131-3; Table 10)

Description	Basic Level	Extended Level	Comments
SAFEBOOL	X	X	A strongly recommended new safety-related data type for binary safety signals only. (For tools where this data type cannot be implemented, the use of BOOL is permitted. However, in that case, data type checking by the compiler is not possible. The user, or tool, is then responsible for ensuring safety and non-safety signals are not mixed up, which may lead to downgrading of the safety integrity level of safety functions.)
BOOL	X	X	For non-safety signals: Exchange with the functional application program only, like an error flag to the operator interface.
INT, DINT	X	X	Basic Level: Only as a constant input parameter to the FB, or when derived from a certified Extended or System level FB. Arithmetic functions are not permitted. Extended Level: Use as a variable permitted. Arithmetic functions are permitted.
REAL	X	X	Same as INT, DINT.
WORD	X	X	Only as an output for diagnostic purposes Extended Level: Use as an internal variable permitted.
TIME	X	X	Only as a constant FB input parameter Extended Level: Use as an internal variable permitted.
Other ANY_BIT	-	-	
Other ANY_INT	-	-	
Other ANY_REAL	-	-	
ANY_DATE	-	-	
STRING	-	-	

Variable Declaration Keywords: (See IEC 61131-3; Table 16)

Description	Basic Level	Extended Level	Comments
VAR	X	X	Only via symbolic declaration.
VAR_INPUT/_OUTPUT	X	X	
VAR_IN_OUT	-	-	
VAR_GLOBAL/EXTERNAL (on FB Level)	-	-	Use of global data may lead to adverse effects and could complicate the analysis of data flow.
VAR_GLOBAL/EXTERNAL (on program level within a single task)	-	X	Restricted use of global data is possible. The use of global data should improve the analysis of data flow.
VAR_ACCESS	-	-	
CONSTANT	X	X	
RETAIN	-	-	

4.4. Reduction in Functions and Function Blocks

Standard Functions: (See IEC 61131-3; Tables 22 - 30)

Description	Basic Level	Extended Level	Comments
AND	X	X	Operation of both BOOL and SAFEBOOL permitted at both levels. Three types of functions are designated to be used: 1) Only SAFEBOOL inputs and one SAFEBOOL output , 2) Only BOOL inputs and one BOOL output, 3) a mix of both for enabling functions: at least one SAFEBOOL input with at least one BOOL input and one SAFEBOOL output
OR	X	X	Basic level: Operation of only SAFEBOOL permitted: Only SAFEBOOL inputs and one SAFEBOOL output Extended level: Operation of both BOOL and SAFEBOOL permitted, but no mixed mode. Two types of functions are designated to be used: 1) Only SAFEBOOL inputs and one SAFEBOOL output , 2) Only BOOL inputs and one BOOL output
XOR, NOT	-	X	
ADD, MUL, SUB, DIV	-	X	No MOD, EXPT, MOVE.
SHL, SHR, ROR, ROL	-	-	Shift functions are not required, as binary information shall not be concatenated to BYTE/WORD.
GT, GE, EQ, LE, LT, NE	-	X	
Selection functions	-	X	
Type conversion functions	X	X	Basic level: Only SAFEBOOL to BOOL conversion permitted. Extended level: For data types that are supported.
String functions	-	-	No STRING available.
Time functions	-	X	Only ADD, SUB, DIV, MUL with TIME operands.
Unary REAL functions	-	-	E.g., SIN, SQRT, LOG.

Standard Function Blocks: (See IEC 61131-3; Tables 34 - 37)

Description	Basic Level	Extended Level	Comments
TON, TOF, TP	X	X	
CTU, CTD, CTUD	X	X	
Bistable FB (SR, RS)	-	X	No semaphores ("SEMA") permitted.
Edge detection	-	X	

4.5. Other Reductions

(See IEC 61131-3; Tables 33, 58)

Description	Basic Level	Extended Level	Comments
Definition of FB	X	X	Basic Level: User Derived FBs for modularization purposes are permitted but shall be encoded only with Basic Level subset.
Directly represented variables	-	-	
STRUCT, ARRAY	-	-	
LD	X	X	See 4.2 Reduction in the Set of Programming Languages with the following restrictions for Basic Level: only power rails, 'normally open' contacts, and (normal) non-negated momentary coils are permitted.
FBD	X	X	See 4.2 Reduction in the Set of Programming Languages with the following restrictions for Basic Level: no negated inputs or outputs are permitted
ST, SFC, IL	-	-	Only permitted on system level. Conforming to IEC 62061.
Other: C, C++, etc.	-	-	Only permitted on system level.
EN/ENO in LD	-	-	
Multiple call of same FB instance	-	-	Every instance must be processed once, and only once, every cycle.
Feedback loop in same network	-	X	The processing order of the FBs must be unique and transparent.
Multiple or conditional return	-	X	Additional return in the event of an error is required and permitted.
Jumps, conditional jumps	-	X	In order to implement the state diagram.
FB declaration features	-	-	See Table 33 of IEC 61131-3.

5. General Rules for Safety-Related Function Blocks

5.1. Function Block-Specific Rules

Default signal	All safety-related Boolean I/O signals have the default safe condition "FALSE".
Signal level	The value of the SAFEBOOL is only applicable as follows: = 0 corresponds to safety as defined at system outputs. = 1 means that the safety aspects of the system are operating correctly, e.g., normal operation is possible. This representation reflects the functionality of the IEC 61131 environments, such as all outputs switch to "0" in the event of an error, as well as default value rules.
Outputs	Every output must be assigned on every cycle.
Missing input/output parameters	Missing parameters are permitted. Default values apply. These default values shall under no circumstances lead to an unsafe state. Default values are specified in the relevant FBs, including their attributes (VARIABLE or CONSTANT).
EN/ENO in LD	Specified FB shall have at least one binary input (i.e., ACTIVATE) and one binary output (i.e., READY), so EN/ENO is not strictly required.
Start behavior	Initially the outputs are set to the default values. After the first call of the function blocks, the outputs are valid. There is a consistent start behavior, so there is no difference in the behavior between cold, warm, and hot start.
Timing diagrams	Timing diagrams, as shown at the FBs, are provided for explanation only. They do not represent the exact timing behavior. The exact timing behavior depends on the implementation (IF versus CASE).
Error handling and diagnostics	All safety-related function blocks have two error-related outputs: Error and DiagCode. These are provided for diagnostic purposes on the user application level, and not for diagnostics on the system/hardware level. The rule for safety-related environments is that the switching of a safety-related function has the highest priority, and following switching there is sufficient time for the diagnostics, either in the functional program or the operator interface.

Table 1: General rules

5.1.1. General Input Parameters

The following tables describe the name, type, and behavior of the generic FB interface:

Input Parameters		
Name	**Type**	**Description**
Activate	BOOL	Variable or constant. Activation of the FB. Initial value is FALSE. This parameter can be connected to the variable, which represents the status (Active or Not Active) of the relevant safety device. This ensures no irrelevant diagnostic information is generated if a device is disabled. If FALSE, all output variables are set to the initial values. If no device is connected, a static TRUE signal must be assigned.
S_<safety-related input name>	SAFExxxx	Every SAFExxxx type input name begins with S_. Only variables may be assigned.
S_StartReset	SAFEBOOL	Variable or constant. FALSE (= initial value): Manual reset when PES is started (warm or cold). TRUE: Automatic reset when PES is started (warm or cold). This function shall only be activated if it is ensured that no hazard can occur at the start of the PES. Therefore the use of the Automatic Circuit Reset feature of the function blocks requires implementation of other system or application measures to ensure that unexpected (or unintended) startup does not occur. It shall be noted in the FB manual that when using a SAFEBOOL variable additional validation of this application is necessary.
S_AutoReset	SAFEBOOL	Variable or constant. FALSE (= initial value): Manual reset when emergency stop button is released.

		TRUE: Automatic reset when emergency stop button is released. This function shall only be activated if it is ensured that no hazard can occur at the start of the PES. Therefore the use of the Automatic Circuit Reset feature of the function blocks requires implementation of other system or application measures to ensure that unexpected (or unintended) startup does not occur. It shall be noted in the FB manual that when using a SAFEBOOL variable additional validation of this application is necessary.
Reset	BOOL	Variable. Initial value is FALSE. Depending on the function, this input can be used for different purposes: • Reset of the state machine, and coupled error and status messages as indicated via DiagCode, when the error cause has been removed. This reset behavior is designed as an error reset. • Manual reset of a "restart interlock" ("Wiederanlaufsperre" in German) by the operator (see EN 954-1). This reset behavior is designed as a functional reset. • Additional FB-specific reset functions. This function is only active on a signal change from FALSE to TRUE. A static TRUE signal causes no further actions, but may be detected as an error in some FBs. The appropriate meaning must be described in every FB. It shall be noted in the FB manual that a SAFEBOOL must be connected instead of a BOOL depending on the safety requirements.

Table 2: Input parameters

5.1.2. General Output Parameters

Output Parameter		
Name	**Type**	**Description**
Ready	BOOL	If TRUE, indicates that the FB is activated and the output results are valid (same as the "POWER" LED of a safety relay). If FALSE, the FB is not active and the program is not executed. Useful in debug mode or to activate/deactivate additional FBs, as well as for further processing in the functional program.
S_<safety-related output name>	SAFExxxx	Every SAFExxxx data type output name begins with S_.
Error	BOOL	Error flag (same as "K1/K2" LED of a safety relay). When TRUE, indicates that an error has occurred, and the FB is in an error state. The relevant error state is mirrored at the DiagCode output. If FALSE, there is no error and the FB is in another state. This again is mirrored by DiagCode (this means that DiagCode must be set in the same cycle as the state change). Useful in debug mode as well as for further processing in the functional program.
DiagCode	WORD	Diagnostic register. All states of the FB (Active, Not Active, and Error) are represented by this register. This information is encoded in hexadecimal format in order to represent more then 16 codes. Only one consistent code is represented at the same time. In the event of multiple errors, the DiagCode output indicates the first detected error. For additional information, see 5.2 Diagnostic Codes. Useful in debug mode as well as for further processing in the functional program.

Table 3: Output parameters

5.2. Diagnostic Codes

A transparent and unique diagnostic concept forms the basis of all function blocks. Thus it is ensured, that, regardless of the supplier's implementation, uniform diagnostic information is available to the user in the form of DiagCode. If no error is present, the internal status of the function block (state machine) is indicated. An error is indicated via a binary output (error). Detailed information about internal or external function block errors can be obtained via DiagCode. The function block must be reset via the different reset inputs.
Suppliers may add additional interfaces via function blocks with supplier-specific diagnostic information.

General Diagnostic Code Ranges	
DiagCode	**Description**
0000_0000_0000_0000$_{bin}$	The FB is not activated or safety CPU is halted.
10xx_xxxx_xxxx_xxxx$_{bin}$	Shows that the activated FB is in an operational state without an error. X = FB-specific code.
11xx_xxxx_xxxx_xxxx$_{bin}$	Shows that the activated FB is in an error state. X = FB-specific code.

Table 4: General diagnostic code ranges

System or Device-Specific Codes	
DiagCode	**Description**
0xxx_xxxx_xxxx_xxxx$_{bin}$	X = System or device-specific message. This information contains the diagnostic information for the system or device, and is mapped directly to the DiagCode output. (Note: 0000hex is reserved)

Table 5: System or device-specific codes

Generic Diagnostic Codes	
DiagCode	**Description**
0000_0000_0000_0000$_{bin}$ **0000$_{hex}$**	The FB is not activated. This code represents the Idle state. For a generic example, the I/O setting for could be: Activate = FALSE S_In = FALSE or TRUE Ready = FALSE Error = FALSE S_Out = FALSE
1000_0000_0000_0000$_{bin}$ **8000$_{hex}$**	The FB is activated without an error or any other condition that sets the safety output to FALSE. This is the default operational state where the S_Out safety output = TRUE in normal operation. For a generic example, the I/O setting for could be: Activate = TRUE S_In = TRUE Ready = TRUE Error = FALSE S_Out = TRUE
1000_0000_0000_0001$_{bin}$ **8001$_{hex}$**	An activation has been detected by the FB and the FB is now activated, but the S_Out safety output is set to FALSE. This code represents the Init state of the operational mode. For a generic example, the I/O setting for could be: Activate = TRUE S_In = FALSE or TRUE Ready = TRUE Error = FALSE S_Out = FALSE

$1000\_0000\_0000\_0010_{bin}$ **8002_{hex}**	The activated FB detects a safety demand ("Sicherheitsanforderung" in German), e.g., S_In = FALSE. The safety output is disabled. This is an operational state where the S_Out safety output = FALSE. For a generic example, the I/O setting for could be: Activate = TRUE S_In = FALSE Ready = TRUE Error = FALSE S_Out = FALSE
$1000\_0000\_0000\_0011_{bin}$ **8003_{hex}**	The safety output of the activated FB has been disabled by a safety demand. The safety demand is now withdrawn, but the safety output remains FALSE until a reset condition is detected. This is an operational state where the S_Out safety output = FALSE. For a generic example, the I/O setting for could be: Activate = TRUE S_In = FALSE => TRUE (continuing with static TRUE) Ready = TRUE Error = FALSE S_Out = FALSE

Table 6: Generic diagnostic codes

Note: If there are more operational states where safety output = TRUE, the next available DiagCode number will be assigned for subsequent states.

5.3. Generic State Diagram

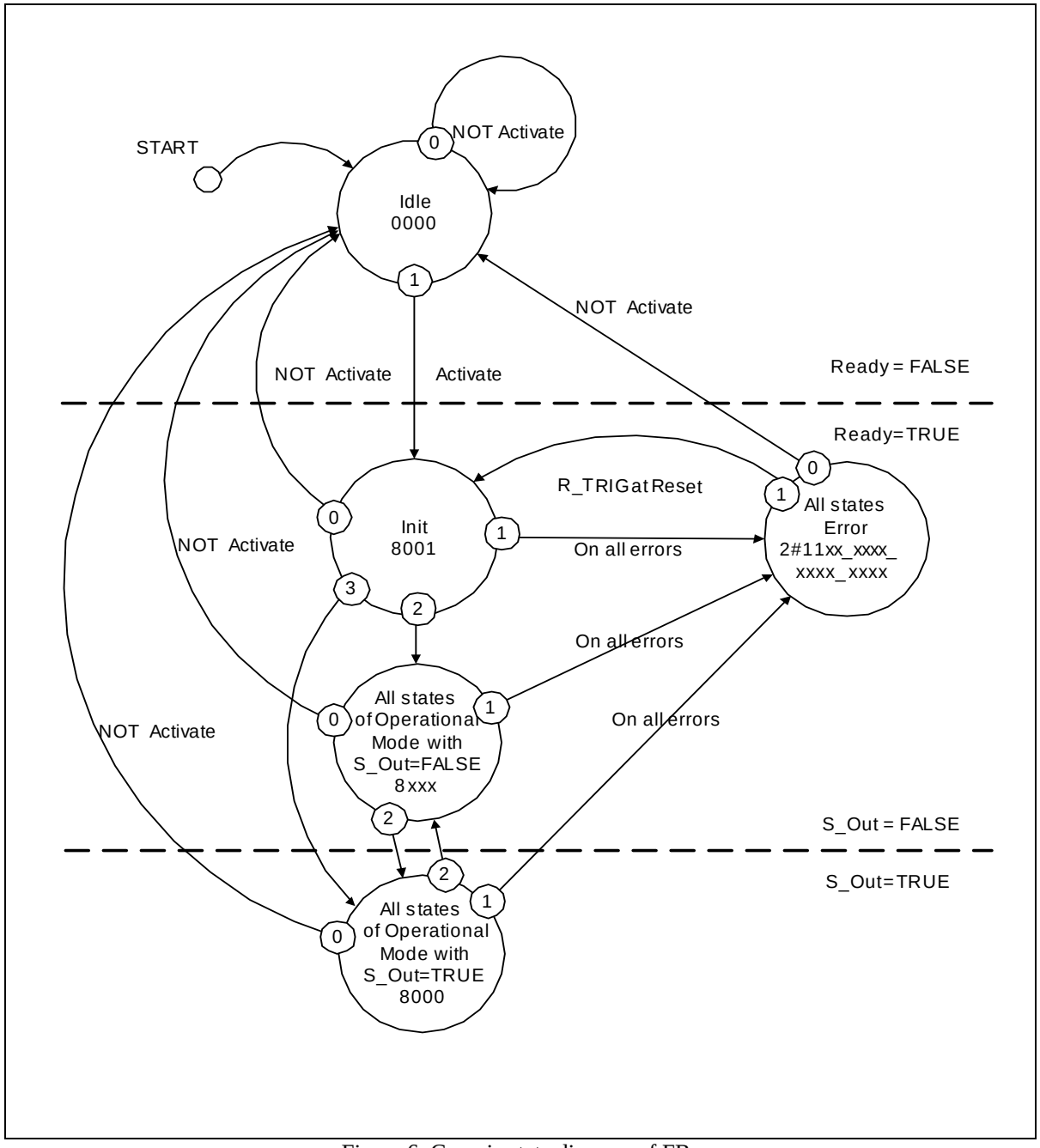


Figure 6: Generic state diagram of FBs

Explanation:

- The above diagram shows a general overview of the states and transitions. Some transitions are not named here, but have a meaning that is FB-specific, and are described with the relevant FBs.
- The diagram shows three areas: At the top the FB is not active and in the Safe state (safe outputs are FALSE), in the middle the FB is active and in the Safe state (safe outputs are FALSE), and at the bottom the FB is in the normal state, i.e., the safe outputs are TRUE.
- The first horizontal line in the state diagram shows the transition from a non-active FB to an active FB.
- The second horizontal line shows the transition from a non-safe state to a safe state of the FB.
- The priorities of possible parallel transitions are indicated by numbers (0 = highest priority).
- State bubbles contain the state name and hexadecimal DiagCode.
- Conditions "OR, AND, XOR" are used as logical operators and "NOT" is used as negation.

- The complete generic state diagram is omitted from the FB description. Within the FB description, the starting state is Idle, with the transitions to operational states via the Init state.
- The transition from any state due to Activate = FALSE, changes to Idle state (0 = highest priority reserved for Activate = FALSE) – for greater clarity, these transitions are not shown in each FB-related state diagram but are mentioned as a footnote to each state diagram.
- For reasons of clarity, the output setting is not described in the state diagram; an explicit truth table containing the "FB states to output(s)" information is part of each FB specification with the FB-specific error and status codes.

DiagCode	**State Name**	**State Description and Output Setting**

FB-specific error codes:

Cxxx	Error	Ready S_Out Error	= TRUE = FALSE = TRUE

FB-specific status codes (no error):

0000	Idle	Ready S_Out Error	= FALSE = FALSE = FALSE
8001	Init state of operational mode	Ready S_Out Error	= TRUE = FALSE = FALSE
8xxx	All states of operational mode where S_Out = FALSE	Ready S_Out Error	= TRUE = FALSE = FALSE
8000	All states of operational mode where S_Out = TRUE	Ready S_Out Error	= TRUE = TRUE = FALSE

Table 7: Function block codes of generic FBs

6. Safety Function Blocks

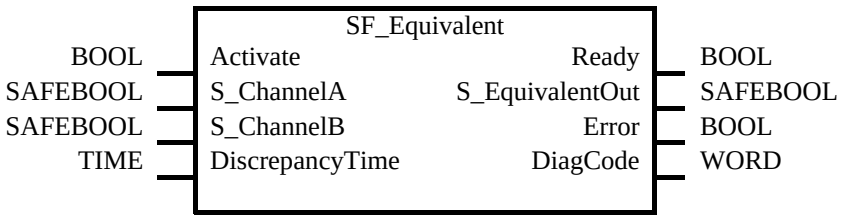
6.1. Equivalent

6.1.1. Applicable Safety Standards

Standards	**Requirements**
EN 954-1: 1996	6.2 General safety priciples, Idle current 6.2 Error detection for category 3 und 4

6.1.2. Interface Description

FB Name	**SF_Equivalent**		
This function block converts two equivalent SAFEBOOL inputs (both NO or NC) to one SAFEBOOL output, including discrepancy time monitoring. This FB should not be used stand-alone since it has no restart interlock. It is required to connect the output to other safety related functionalities.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_ChannelA	SAFEBOOL	FALSE	Variable. Input A for logical connection. FALSE: Contact A open TRUE: Contact A closed.
S_ChannelB	SAFEBOOL	FALSE	Variable. Input B for logical connection. FALSE: Contact B open TRUE: Contact B closed.
DiscrepancyTime	TIME	T#0ms	Constant. Maximum monitoring time for discrepancy status of both inputs.
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_EquivalentOut	SAFEBOOL	FALSE	Safety related output FALSE: Minimum of one input signal = "FALSE" or status change outside of monitoring time. TRUE: Both input signals "active" and status change within monitoring time.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: --			

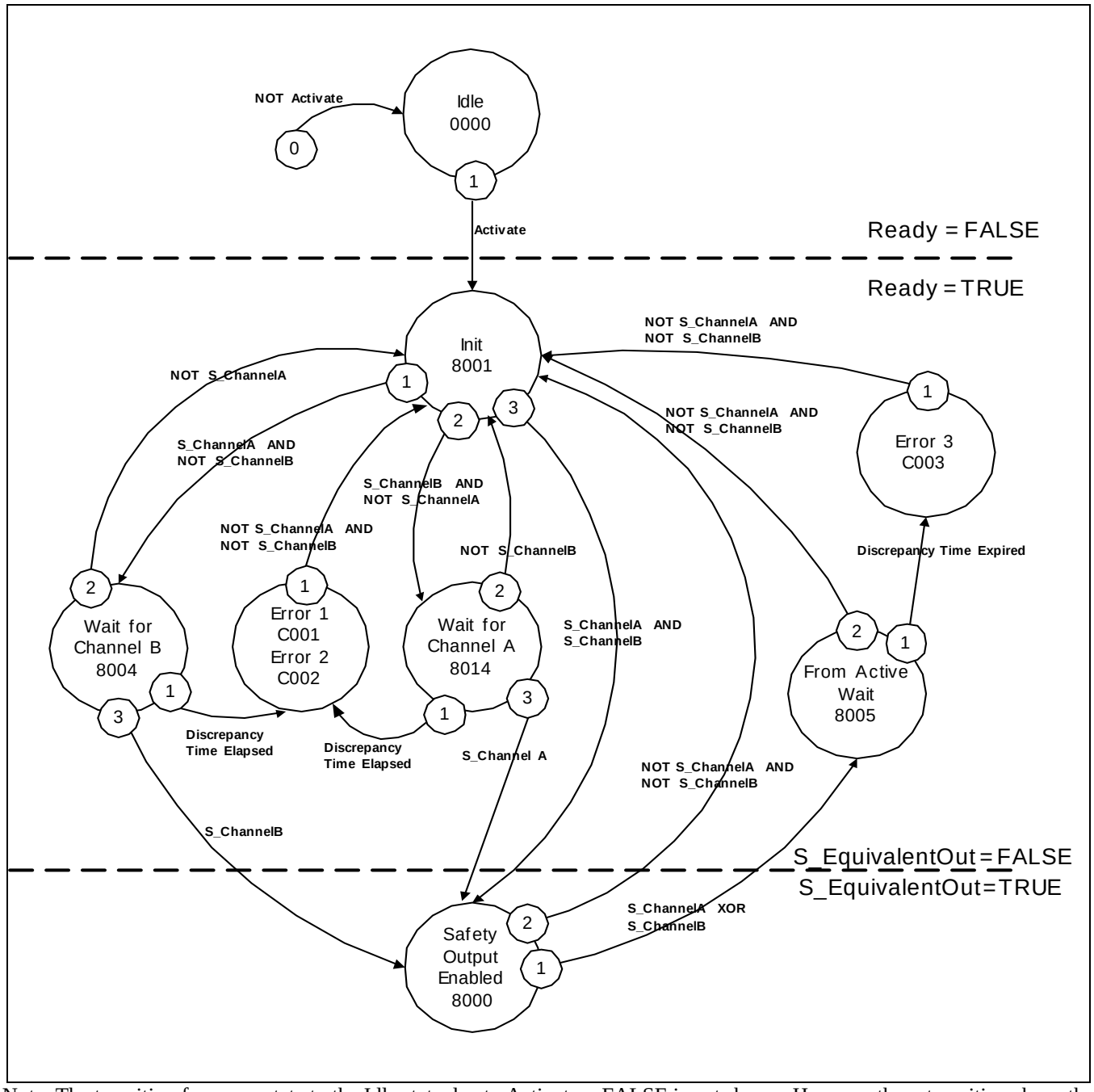


6.1.3. Functional Description

This function block converts two equivalent SAFEBOOL inputs to one SAFEBOOL output with discrepancy time monitoring. Both input Channels A and B are interdependent. The function block output shows the result of the evaluation of both channels.
If one channel signal changes from TRUE to FALSE the output immediately switches off (FALSE) for safety reasons.

Discrepancy time monitoring: The discrepancy time is the maximum period during which both inputs may have different states without the function block detecting an error. Discrepancy time monitoring starts when the status of an input changes. The function block detects an error when both inputs do not have the same status once the discrepancy time has elapsed.
The inputs must be switched symmetrically. This means that monitoring is performed for both the switching on process as well as the switching off process.

State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 7: State diagram for SF_Equivalent

Typical Timing Diagrams

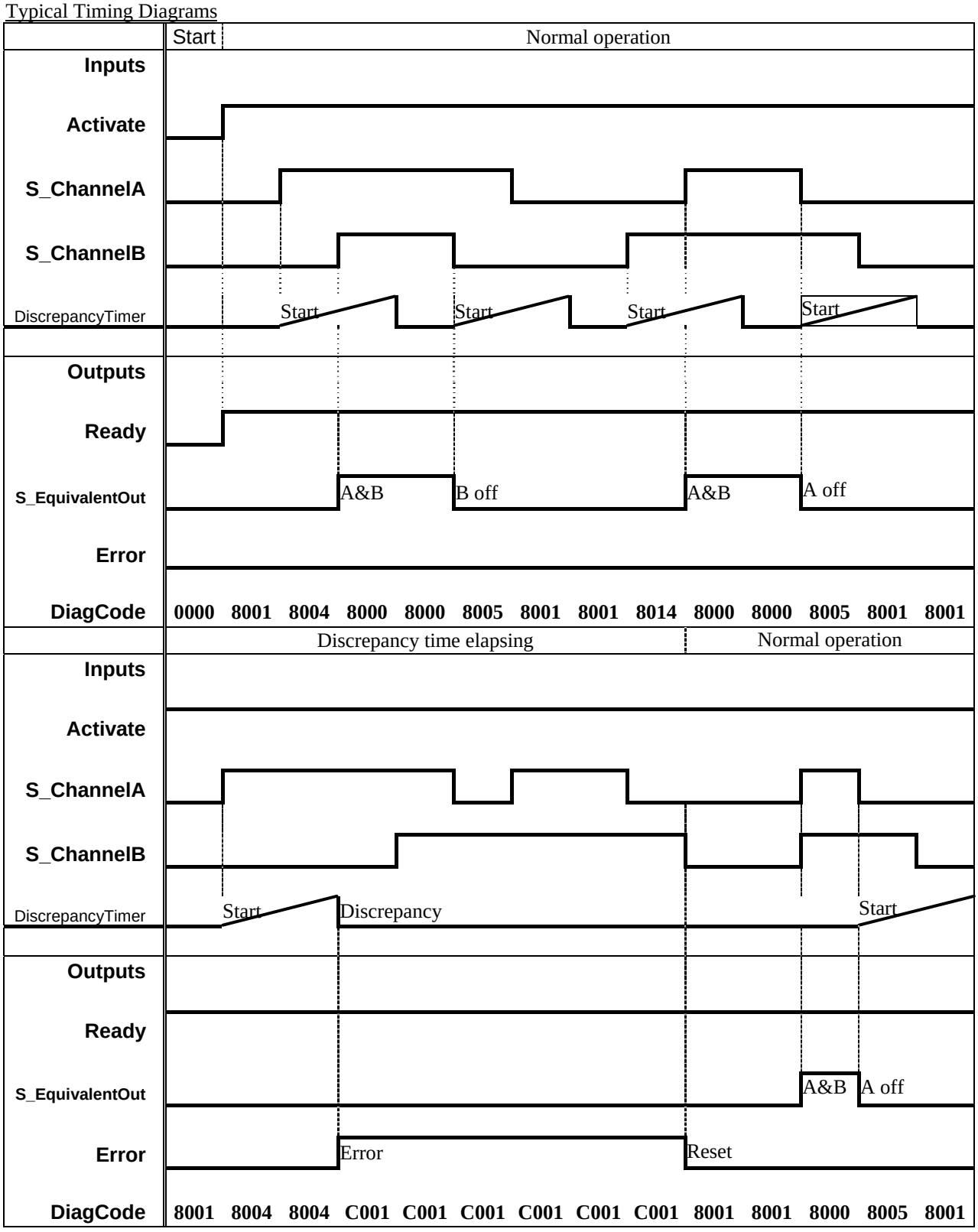


Figure 8: Timing diagrams for SF_Equivalent

6.1.4. Error Detection

The function block monitors the discrepancy time between Channel A and B, when switching to TRUE and also when switching to FALSE.

6.1.5. Error Behavior

S_EquivalentOut is set to FALSE. Error is set to TRUE. DiagCode indicates the Error states. There is no Reset defined as an input coupled with the reset of an error. If an error occurs in the inputs, a new set of inputs with correct S_EquivalentOut must be able to reset the error flag. (Example: if a switch is faulty and replaced, using the switch again results in a correct output)

6.1.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting
FB-specific error codes:		
C001	Error 1	Discrepancy time elapsed in state 8004. Ready = TRUE S_EquivalentOut = FALSE Error = TRUE
C002	Error 2	Discrepancy time elapsed in state 8014. Ready = TRUE S_EquivalentOut = FALSE Error = TRUE
C003	Error 3	Discrepancy time elapsed in state 8005. Ready = TRUE S_EquivalentOut = FALSE Error = TRUE
FB-specific status codes (no error):		
0000	Idle	The function block is not active (initial state). Ready = FALSE S_EquivalentOut = FALSE Error = FALSE
8001	Init	An activation has been detected by the FB and the FB is now activated. Ready = TRUE S_EquivalentOut = FALSE Error = FALSE
8000	Safety Output Enabled	The inputs switched to TRUE in equivalent mode. Ready = TRUE S_EquivalentOut = TRUE Error = FALSE
8004	Wait for Channel B	Channel A has been switched to TRUE - waiting for Channel B; discrepancy timer started. Ready = TRUE S_EquivalentOut = FALSE Error = FALSE
8014	Wait for Channel A	Channel B has been switched to TRUE - waiting for Channel A; discrepancy timer started. Ready = TRUE S_EquivalentOut = FALSE Error = FALSE
8005	From Active Wait	One channel has been switched to FALSE; waiting for the second channel to be switched to FALSE, discrepancy timer started. Ready = TRUE S_EquivalentOut = FALSE Error = FALSE

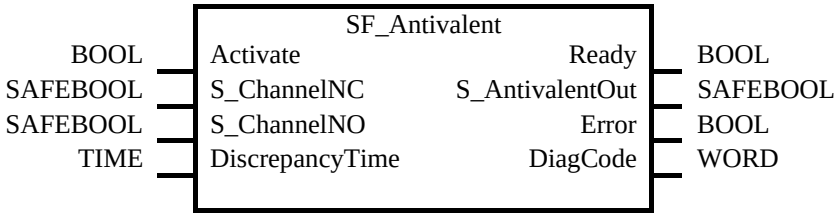
6.2. Antivalent

6.2.1. Applicable Safety Standards

Standards	**Requirements**
EN 954-1: 1996	6.2 General safety priciples, Idle current 6.2 Error detection for category 3 und 4

6.2.2. Interface Description

FB Name	**SF_Antivalent**		
This function block converts two antivalent SAFEBOOL inputs (NO/NC pair) to one SAFEBOOL output with discrepancy time monitoring. This FB should not be used stand-alone since it has no restart interlock. It is required to connect the output to other safety related functionalities.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_ChannelNC	SAFEBOOL	FALSE	Variable. NC stands for Normally Closed. Input for NC connection. FALSE: NC contact open. TRUE: NC contact closed.
S_ChannelNO	SAFEBOOL	TRUE	Variable. NO stands for Normally Open. Input for NO connection. FALSE: NO contact open TRUE: NO contact closed
DiscrepancyTime	TIME	T#0ms	Constant. Maximum monitoring time for discrepancy status of both inputs.
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_AntivalentOut	SAFEBOOL	FALSE	Safety related output FALSE: Minimum of one input signal "not active" or status change outside of monitoring time. TRUE: Both inputs signals "active" and status change within monitoring time.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: "Antivalent" means that during normal operation, the two inputs are in opposite states at the same time. This is sometimes called "complementary" or "non-equivalent".			

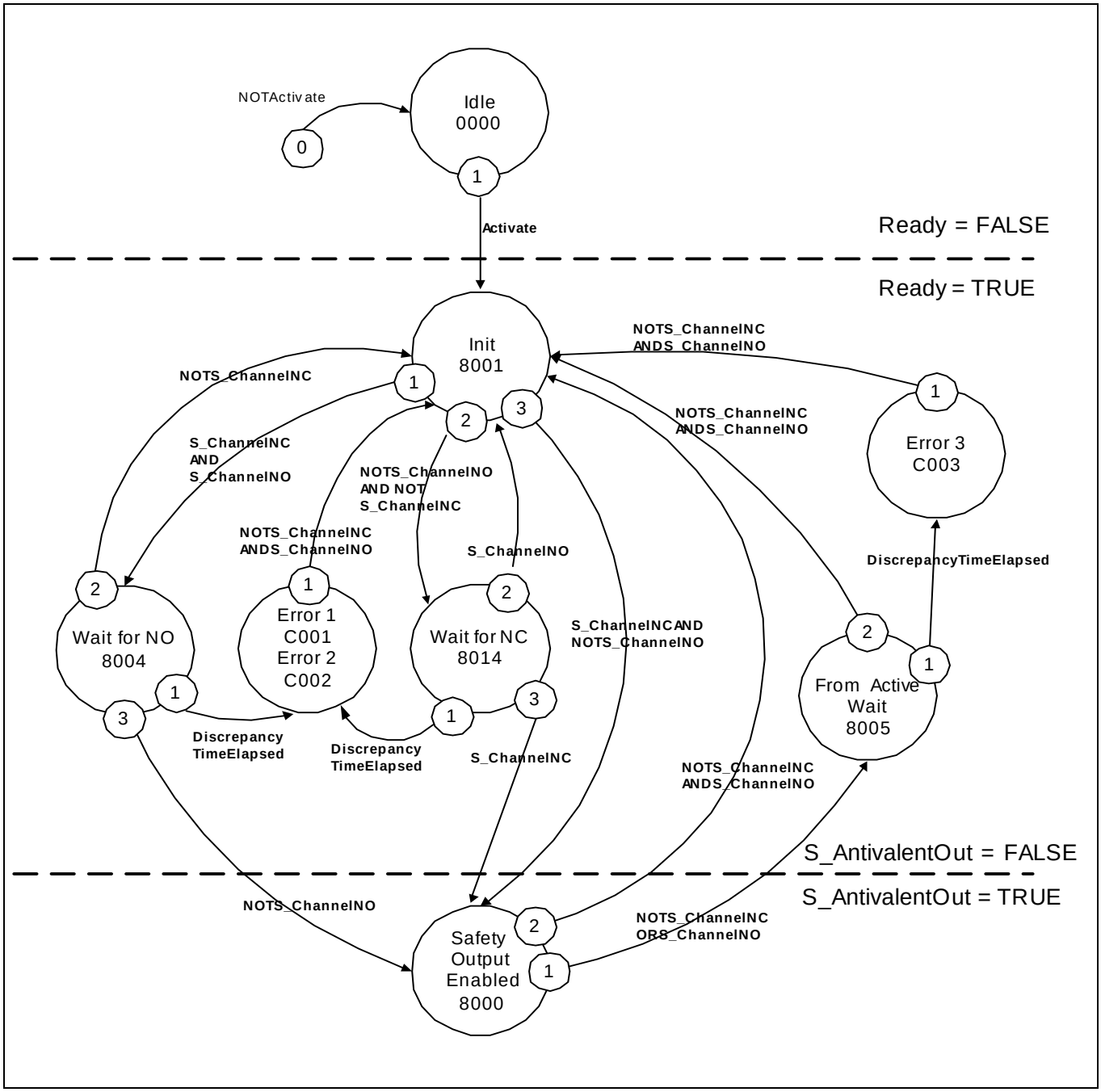


6.2.3. Functional Description

This function block converts two antivalent SAFEBOOL inputs to one SAFEBOOL output with discrepancy time monitoring. Both input channels are interdependent. The function block output shows the result of the evaluation of both channels.
If S_AntivalentOut = TRUE and one of the safety related inputs changes, the output immediately switches to FALSE.
Discrepancy time monitoring: The discrepancy time is the maximum period during which both inputs may have the same states (i.e., both inputs are either TRUE or FALSE) without the function block detecting an error. Discrepancy time monitoring starts when the status of an input changes. The function block detects an error when both inputs do not have antivalent values once

the discrepancy time has elapsed.
The inputs must be switched symmetrically. This means that monitoring is performed for both the switching on process as well as the switching off process.

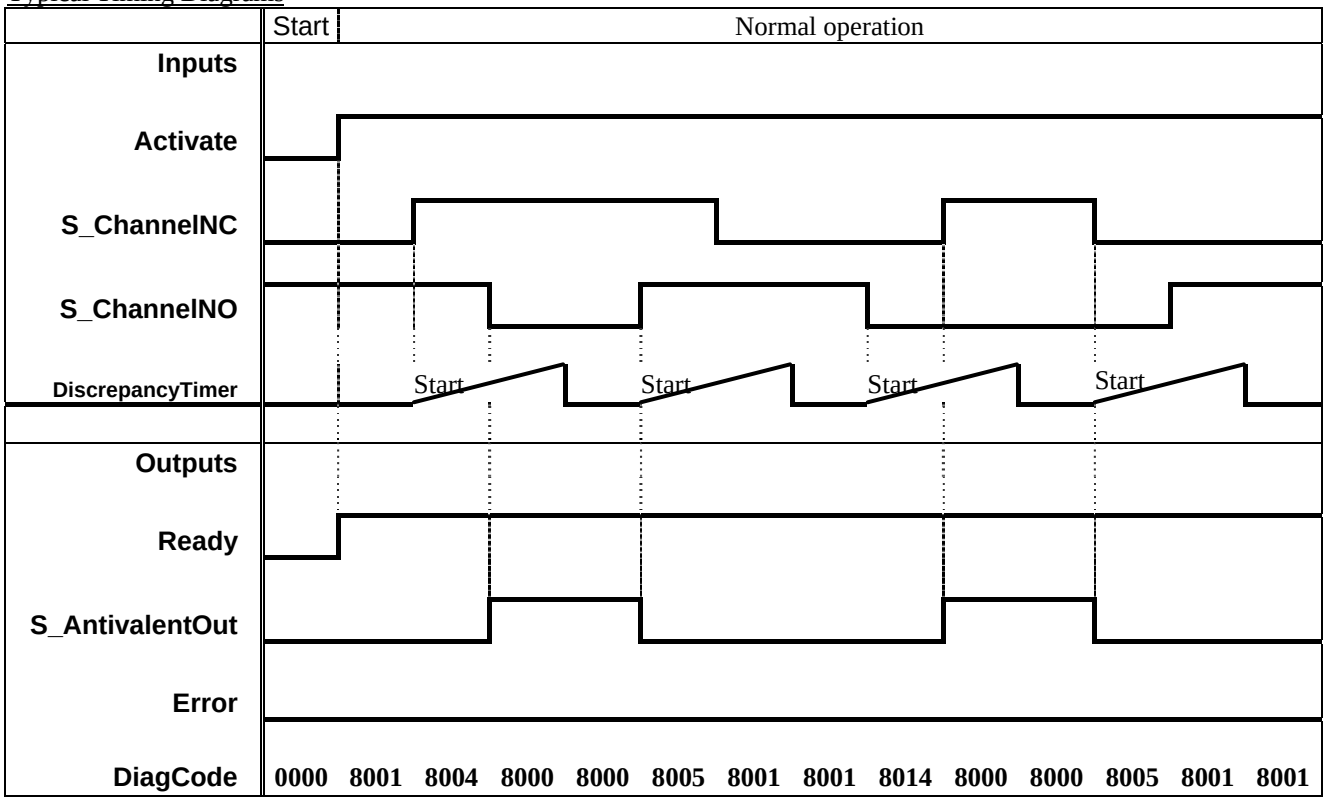
State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 9: State diagram for SF_Antivalent

Typical Timing Diagrams



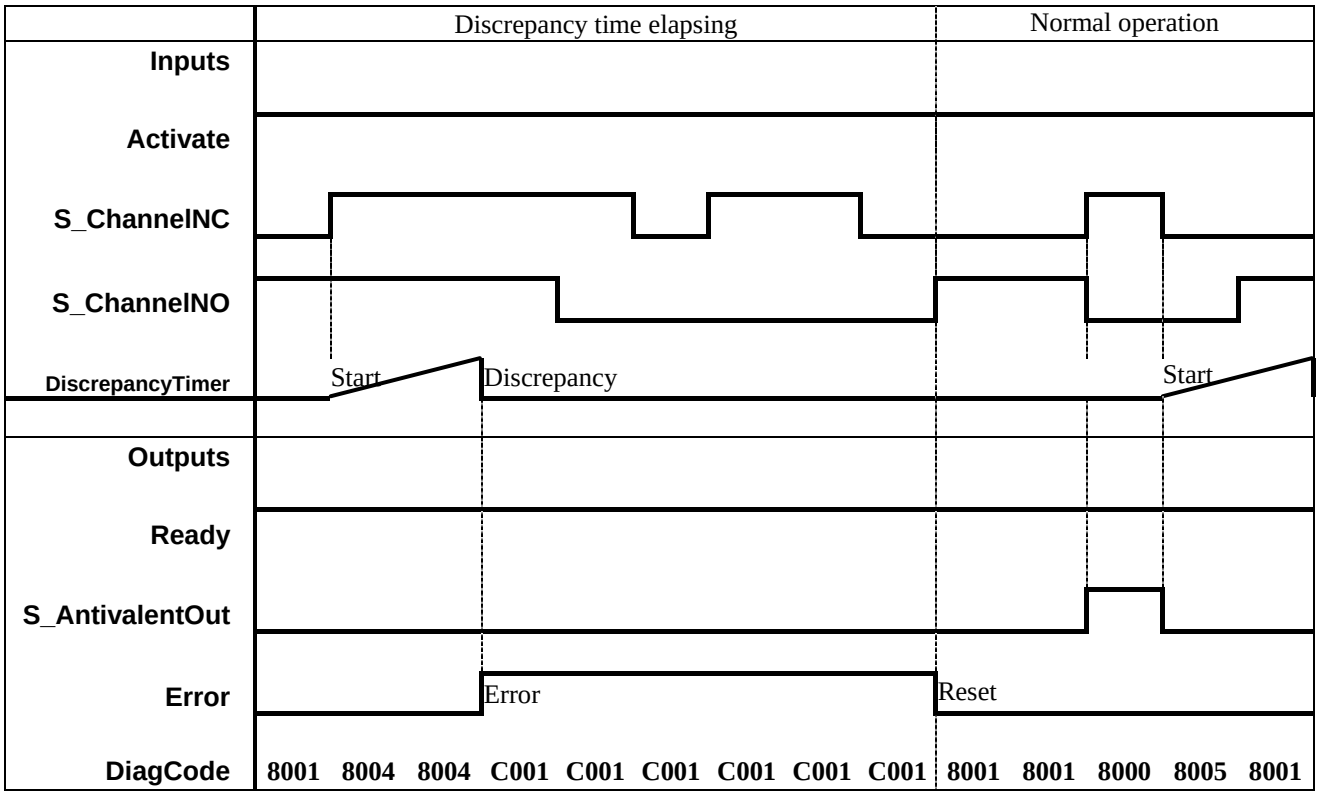


Figure 10: Timing diagrams for SF_Antivalent

6.2.4. Error Detection

The function block monitors the discrepancy time between Channel NO and Channel NC.

6.2.5. Error Behavior

The output S_AntivalentOut is set to FALSE. Error is set to TRUE. DiagCode indicates the Error states.

There is no Reset defined as an input coupled with the reset of an error. If an error occurs in the inputs, one new set of inputs with the correct value must be able to reset the error flag. (Example: if a switch is faulty and replaced, using the switch again results in a correct output)

6.2.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting
FB-specific error codes:		
C001	Error 1	Discrepancy time elapsed in state 8004. Ready = TRUE S_AntivalentOut = FALSE Error = TRUE
C002	Error 2	Discrepancy time elapsed in state 8014. Ready = TRUE S_AntivalentOut = FALSE Error = TRUE
C003	Error 3	Discrepancy time elapsed in state 8005. Ready = TRUE S_AntivalentOut = FALSE Error = TRUE
FB-specific status codes (no error):		
0000	Idle	The function block is not active (initial state). Ready = FALSE S_AntivalentOut = FALSE Error = FALSE
8001	Init	An activation has been detected by the FB and the FB is now activated. Ready = TRUE S_AntivalentOut = FALSE Error = FALSE
8000	Safety Output Enabled	The inputs switched to the Active state in antivalent mode. Ready = TRUE S_AntivalentOut = TRUE Error = FALSE
8004	Wait for NO	ChannelNC has been switched to TRUE - waiting for ChannelNO to be switched to FALSE; discrepancy timer started. Ready = TRUE S_AntivalentOut = FALSE Error = FALSE
8014	Wait for NC	ChannelNO has been switched to FALSE - waiting for ChannelNC to be switched to TRUE; discrepancy timer started. Ready = TRUE S_AntivalentOut = FALSE Error = FALSE
8005	From Active Wait	One channel has been switched to inactive; waiting for the second channel to be switched to inactive too. Ready = TRUE S_AntivalentOut = FALSE Error = FALSE

6.3. Mode Selector

6.3.1. Applicable Safety Standards

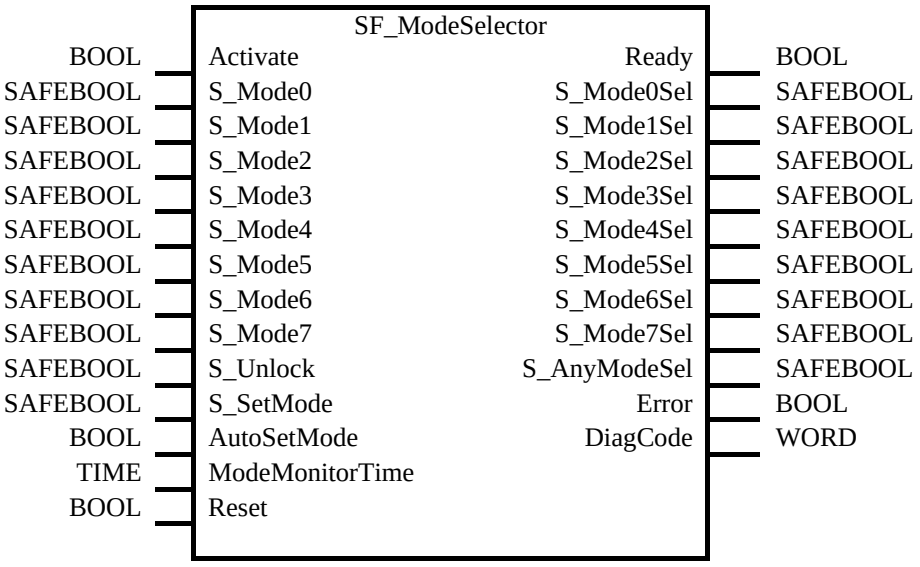
Standards	**Requirements**
MRL 98/37/EC, Annex I	1.2.3. Starting ... It must be possible to start machinery only by voluntary actuation of a control provided for the purpose.... The same requirement applies:... - when effecting a significant change in the operating conditions.... 1.2.5 ... mode selector which can be locked in each position. Each position of the selector must correspond to a single operating or control mode....
EN ISO 12100-2: 2003	4.11.10 Selection of Control and Operating Modes … shall be fitted with a mode selector which can be locked in each position. Each position of the selector shall be clearly identifiable and shall exclusively enable one control or operating mode to be selected…
IEC 60204-1, Ed. 5.0 : 2003	9.2.3 Operating Modes ...When a hazardous condition can result from a mode selection, unauthorized and/or inadvertent selection shall be prevented by suitable means (e.g. key operated switch, access code). Mode selection by itself shall not initiate machine operation. A separate action by the operator shall be required. ...Indication of the selected operating mode shall be provided...
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.3.2. Interface Description

FB Name	**SF_ModeSelector**		
This function block selects the system operation mode, such as manual, automatic, semi-automatic, etc.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Table 5.1.1 General Input Parameters
S_Mode0	SAFEBOOL	FALSE	Variable or constant. Input 0 from mode selector switch FALSE: Mode 0 is not requested by operator. TRUE: Mode 0 is requested by operator.
S_Mode1	SAFEBOOL	FALSE	Variable or constant. Input 1 from mode selector switch FALSE: Mode 1 is not requested by operator. TRUE: Mode 1 is requested by operator.
S_Mode2	SAFEBOOL	FALSE	Variable or constant. Input 2 from mode selector switch FALSE: Mode 2 is not requested by operator. TRUE: Mode 2 is requested by operator.
S_Mode3	SAFEBOOL	FALSE	Variable or constant. Input 3 from mode selector switch FALSE: Mode 3 is not requested by operator. TRUE: Mode 3 is requested by operator.
S_Mode4	SAFEBOOL	FALSE	Variable or constant. Input 4 from mode selector switch FALSE: Mode 4 is not requested by operator. TRUE: Mode 4 is requested by operator.
S_Mode5	SAFEBOOL	FALSE	Variable or constant. Input 5 from mode selector switch FALSE: Mode 5 is not requested by operator. TRUE: Mode 5 is requested by operator.
S_Mode6	SAFEBOOL	FALSE	Variable or constant. Input 6 from mode selector switch FALSE: Mode 6 is not requested by operator. TRUE: Mode 6 is requested by operator.

S_Mode7	SAFEBOOL	FALSE	Variable or constant. Input 7 from mode selector switch FALSE: Mode 7 is not requested by operator. TRUE: Mode 7 is requested by operator.
S_Unlock	SAFEBOOL	FALSE	Variable or constant. Locks the selected mode FALSE: The actual S_ModeXSel output is locked therefore a change of any S_ModeX input does **not** lead to a change in the S_ModeXSel output even in the event of a rising edge of Set-Mode. TRUE: The selected S_ModeXSel is not locked; a mode selection change is possible.
S_SetMode	SAFEBOOL	FALSE	Variable (or constant FALSE, if AutoSetMode = TRUE) Sets the selected mode Operator acknowledges the setting of a mode. Any change to new S_ModeX = TRUE leads to S_AnyModeSel/S_ModeXSel = FALSE, only a rising SetMode trigger then leads to new S_ModeXSel = TRUE.
AutoSetMode	BOOL	FALSE	Constant. Parameterizes the acknowledgement mode FALSE: A change in mode must be acknowledged by the operator via SetMode. TRUE: A valid change of the S_ModeX input to another S_ModeX automatically leads to a change in S_ModeXSel without operator acknowledgment via SetMode (as long as this is not locked by S_Unlock).
ModeMonitorTime	TIME	T#0	Constant. Maximum permissible time for changing the selection input.
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_Mode0Sel	SAFEBOOL	FALSE	Indicates that mode 0 is selected and acknowledged. FALSE: Mode 0 is not selected or not active. TRUE: Mode 0 is selected and active.
S_Mode1Sel	SAFEBOOL	FALSE	Indicates that mode 1 is selected and acknowledged. FALSE: Mode 1 is not selected or not active. TRUE: Mode 1 is selected and active.
S_Mode2Sel	SAFEBOOL	FALSE	Indicates that mode 2 is selected and acknowledged. FALSE: Mode 2 is not selected or not active. TRUE: Mode 2 is selected and active.
S_Mode3Sel	SAFEBOOL	FALSE	Indicates that mode 3 is selected and acknowledged. FALSE: Mode 3 is not selected or not active. TRUE: Mode 3 is selected and active.
S_Mode4Sel	SAFEBOOL	FALSE	Indicates that mode 4 is selected and acknowledged. FALSE: Mode 4 is not selected or not active. TRUE: Mode 4 is selected and active.
S_Mode5Sel	SAFEBOOL	FALSE	Indicates that mode 5 is selected and acknowledged. FALSE: Mode 5 is not selected or not active. TRUE: Mode 5 is selected and active.
S_Mode6Sel	SAFEBOOL	FALSE	Indicates that mode 6 is selected and acknowledged. FALSE: Mode 6 is not selected or not active. TRUE: Mode 6 is selected and active.
S_Mode7Sel	SAFEBOOL	FALSE	Indicates that mode 7 is selected and acknowledged. FALSE: Mode 7 is not selected or not active. TRUE: Mode 7 is selected and active.
S_AnyModeSel	SAFEBOOL	FALSE	Indicates that any of the 8 modes is selected and acknowledged. FALSE: No S_ModeX is selected. TRUE: One of the 8 S_ModeX is selected and active.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters

DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: The X in parameter names "S_ModeX" or "S_ModeXSel" is a placeholder for digits 0 to 7.			



6.3.3. Functional Description

This function block selects the system operation mode, such as manual, automatic, semi-automatic, etc. On controller startup, it should be assumed that the machine is in safe mode. On machine startup, the transition to the mode set by the mode selector switch must be initiated by a function block input (e.g., machine START button).

The default state following activation of the FB is the ModeChanged state. This is also the safe state of the FB, where all S_ModeXSel and S_AnyModeSel are FALSE.

If the FB is in the ModeChanged state:

- The new S_ModeX input must be acknowledged by a rising S_SetMode trigger (if AutoSetMode = FALSE), which leads to a new S_ModeXSel output.
- The new S_ModeX input automatically leads to a new S_ModeXSel output (if AutoSetMode = TRUE).
- Such a transition from state 8005 to 8000 is only valid, if one S_ModeX input is TRUE. As long as all S_ModeX are FALSE, the FB remains in state 8005, even if the S_SetMode triggers.

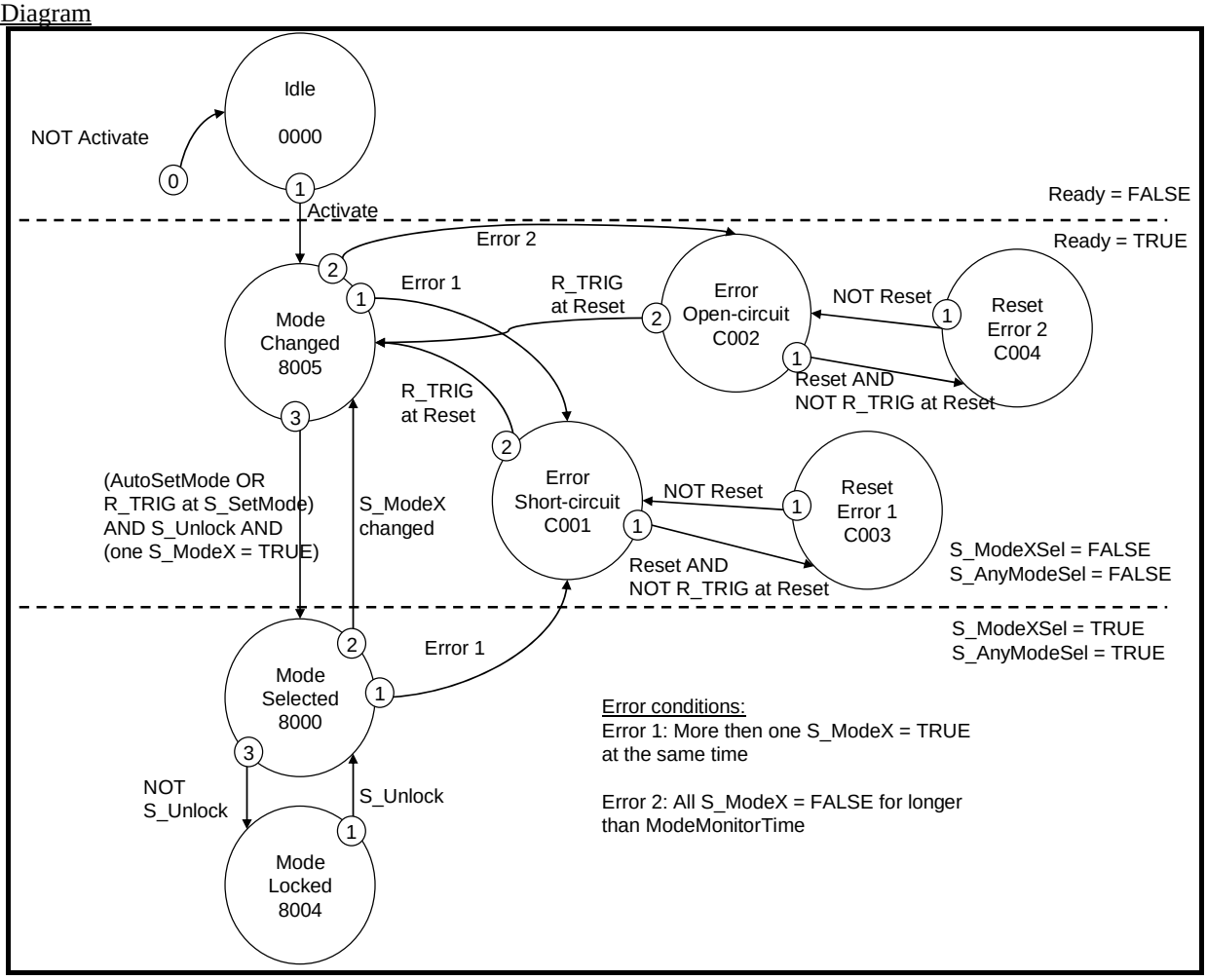
The transition from the ModeChanged to ModeSelected state, i.e., S_SetMode set by the operator, is not monitored by a timer.

If the FB is in the ModeSelected state, the simultaneous occurrence of a new S_ModeX input (higher priority) and the NOT S_Unlock signal (lower priority) leads to the ModeChanged state.

The S_ModeX input parameters, which are not used for mode selection, should be called with the default value FALSE to simplify program verification.

The AutoSetMode input shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 11: State diagram for SF_ModeSelector

Typical Timing Diagrams

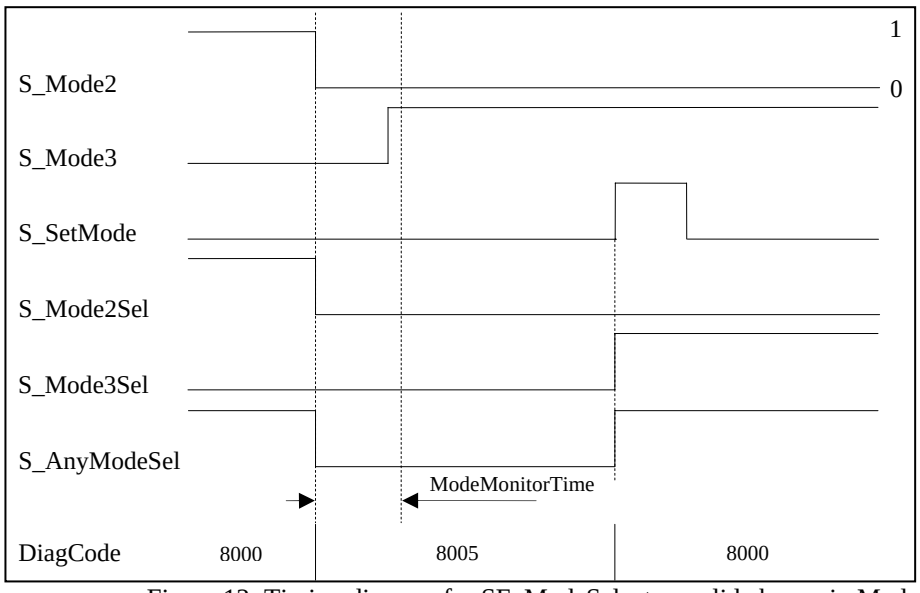


Figure 12: Timing diagram for SF_ModeSelector, valid change in Mode input with acknowledgment

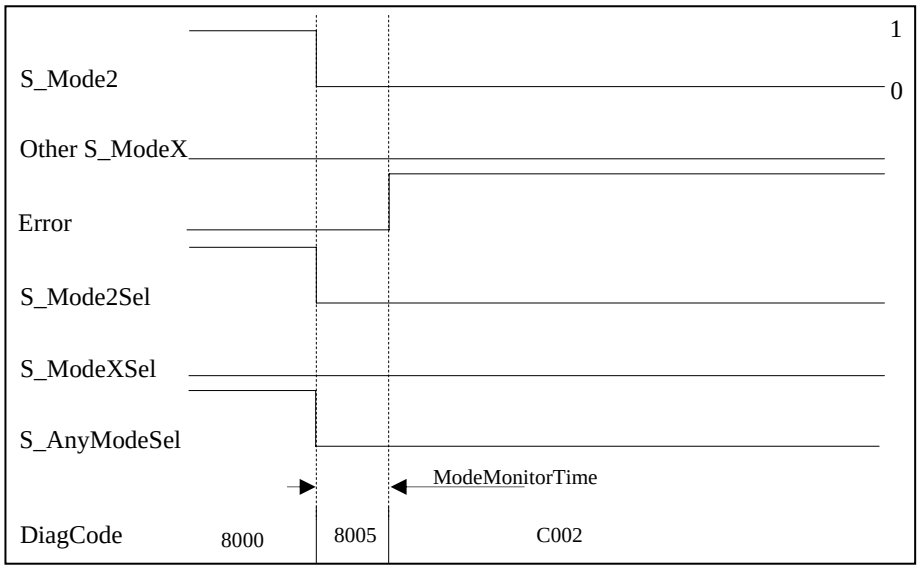


Figure 13: Timing diagram for SF_ModeSelector, error condition 2 at Mode inputs

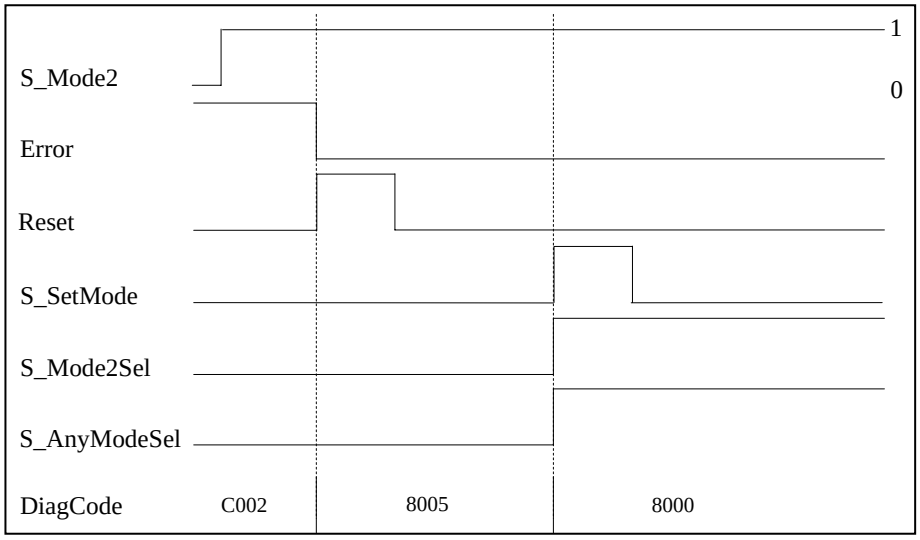


Figure 14: Timing diagram for SF_ModeSelector, reset of error condition

6.3.4. Error Detection

The FB detects whether none of the mode inputs is selected. This invalid condition is detected after ModeMonitorTime has elapsed:

- Which restarts with each falling trigger of an S_ModeX switched mode input
- Which is then in the ModeChanged state following activation of the FB

In contrast, the FB directly detects whether more than one S_ModeX mode input is selected at the same time.
A static reset condition is detected when the FB is either in Error state C001 or C002.

6.3.5. Error Behavior

In the event of an error, the S_ModeXSel and S_AnyModeSel outputs are set to safe state = FALSE. The DiagCode output indicates the relevant error code and the Error output is set to TRUE.
An error must be acknowledged with the rising trigger of the Reset BOOL input. The FB changes from an error state to the ModeChanged state.

6.3.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C001	Error Short-circuit	The FB detected that two or more S_ModeX are TRUE, e.g., short-circuit of cables. Ready = TRUE Error = TRUE S_AnyModeSel = FALSE All S_ModeXSel = FALSE
C002	Error Open-circuit	The FB detected that all S_ModeX are FALSE: The period following a falling S_ModeX trigger exceeds ModeMonitorTime, e.g., open-circuit of cables. Ready = TRUE Error = TRUE S_AnyModeSel = FALSE All S_ModeXSel = FALSE
C003	Reset Error 1	Static Reset signal detected in state C001. Ready = TRUE Error = TRUE S_AnyModeSel = FALSE All S_ModeXSel = FALSE
C004	Reset Error 2	Static Reset signal detected in state C002. Ready = TRUE Error = TRUE S_AnyModeSel = FALSE All S_ModeXSel = FALSE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE Error = FALSE S_AnyModeSel = FALSE All S_ModeXSel = FALSE
8005	ModeChanged	State after activation or when S_ModeX has changed (unless locked) or after Reset of an error state. Ready = TRUE Error = FALSE S_AnyModeSel = FALSE All S_ModeXSel = FALSE
8000	ModeSelected	Valid mode selection, but not yet locked. Ready = TRUE Error = FALSE S_AnyModeSel = TRUE S_ModeXSel = Selected X is TRUE, others are FALSE.
8004	ModeLocked	Valid mode selection is locked. Ready = TRUE Error = FALSE S_AnyModeSel = TRUE S_ModeXSel = Selected X is TRUE, others are FALSE.

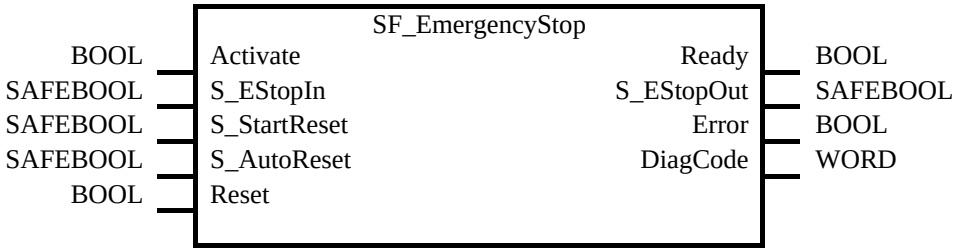
6.4. Emergency Stop

6.4.1. Applicable Safety Standards

Standards	**Requirements**
EN 418: 1992	3. Definitions 4.1.12 ... Resetting the control device shall not by itself cause a restart command. .
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart
EN 16204-1, 1997	9.2.2. Stop Functions

6.4.2. Interface Description

FB Name	**SF_EmergencyStop**		
This function block is a safety-related function block for monitoring an emergency stop button. This FB can be used for emergency switch off functionality (stop category 0), or - with additional peripheral support - as emergency stop (stop category 1 or 2)			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_EStopIn	SAFEBOOL	FALSE	Safety demand input. Variable. FALSE: Demand for safety-related response (e.g., emergency stop button is engaged). TRUE: No demand for safety-related response (e.g., emergency stop button not engaged).
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
S_AutoReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_EStopOut	SAFEBOOL	FALSE	Output for the safety-related response. FALSE: Safety output disabled. Demand for safety-related response (e.g., emergency stop button engaged, reset required or internal errors active) TRUE: Safety output enabled. No demand for safety-related response (e.g., emergency stop button not engaged, no internal errors active).
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: The following requirements as defined in EN 418: 1992 have to be fulfilled by the user: Ch. 4.1.4 After activation of the actuator, the emergency stop equipment shall operate in such a way that the hazard is averted or reduced automatically in the best possible manner. 4.1.7 The emergency stop command shall override all other commands. 4.1.12 Resetting the control device shall only be possible as the result of a manual action on the control device itself....It shall not be possible to restart the machine until all control devices which have been actuated are reset manually, individually and intentionally.			



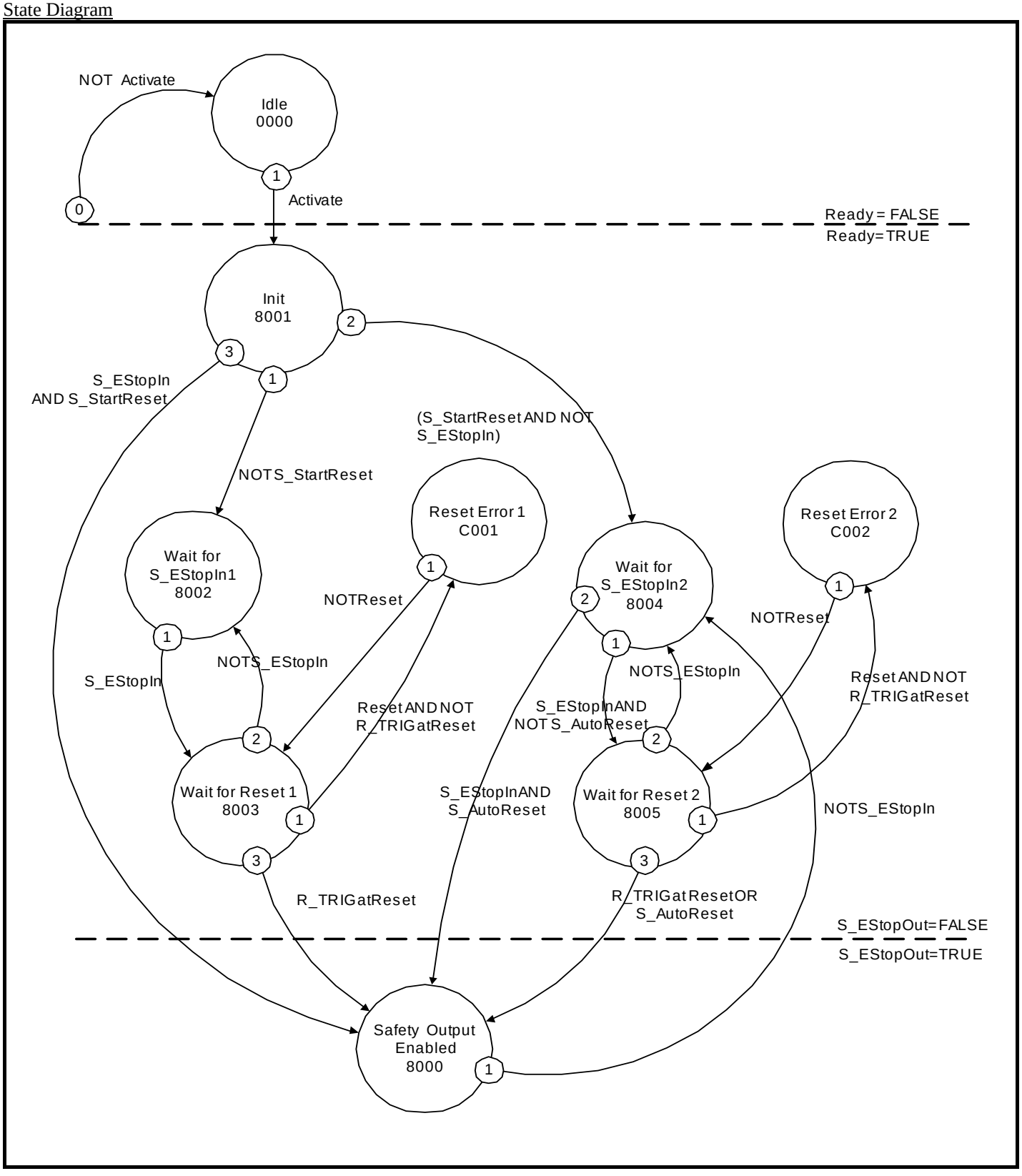
6.4.3. Functional Description

The S_EStopOut enable signal is reset to FALSE as soon as the S_EStopIn input is set to FALSE. The S_EStopOut enable signal is reset to TRUE only if the S_EStopIn input is set to TRUE and a reset occurs. The enable reset depends on the defined S_StartReset, S_AutoReset, and Reset inputs.
If S_AutoReset = TRUE, acknowledgment is automatic.
If S_AutoReset = FALSE, a rising trigger at the Reset input must be used to acknowledge the enable.
If S_StartReset = TRUE, acknowledgment is automatic the fist time the PES is started.
If S_StartReset = FALSE, a rising trigger at the Reset input must be used to acknowledge the enable.

The S_StartReset and S_AutoReset inputs shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

SF_EmergencyStop can be used to monitor both single and two-channel emergency stop buttons. For example, for two-channel applications, the additional function blocks SF_Equivalent can be used to detect whether the contact synchronization has been exceeded. The category classification in accordance with EN 954-1 will depend on the final elements that are used.
The SF_EmergencyStop automatically detects a static TRUE on Reset. Further error detection, e.g., wire break, short circuit depends on the dedicated hardware that is used.

State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 15: State diagram for SF_EmergencyStop

Typical Timing Diagrams

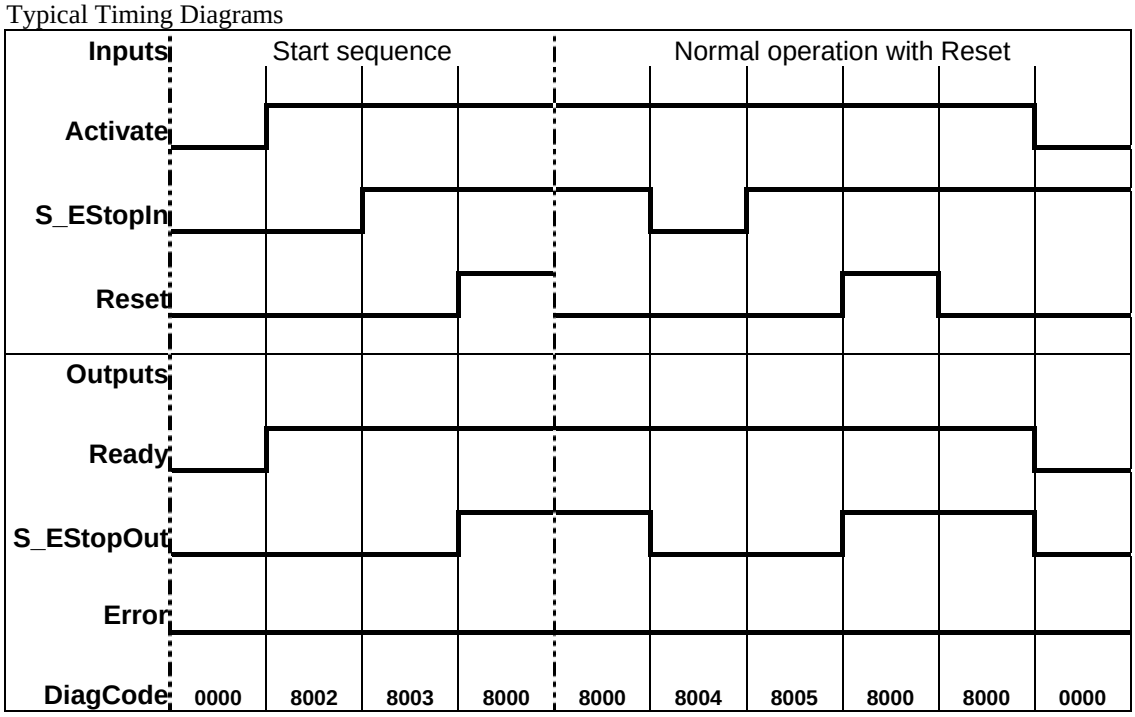


Figure 16: Timing diagram for SF_EmergencyStop: S_StartReset = FALSE; S_AutoReset = FALSE; Start, reset, normal operation, safety demand, restart

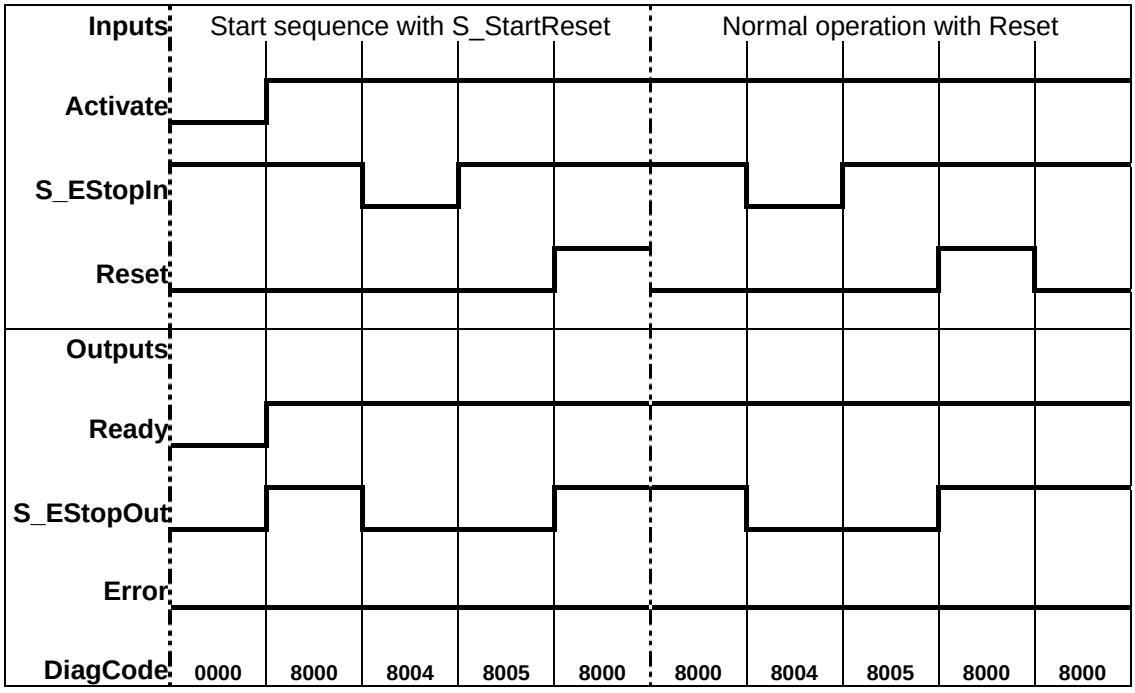


Figure 17: Timing diagram for SF_EmergencyStop: S_StartReset = TRUE, S_AutoReset = FALSE; Start, normal operation, safety demand, restart

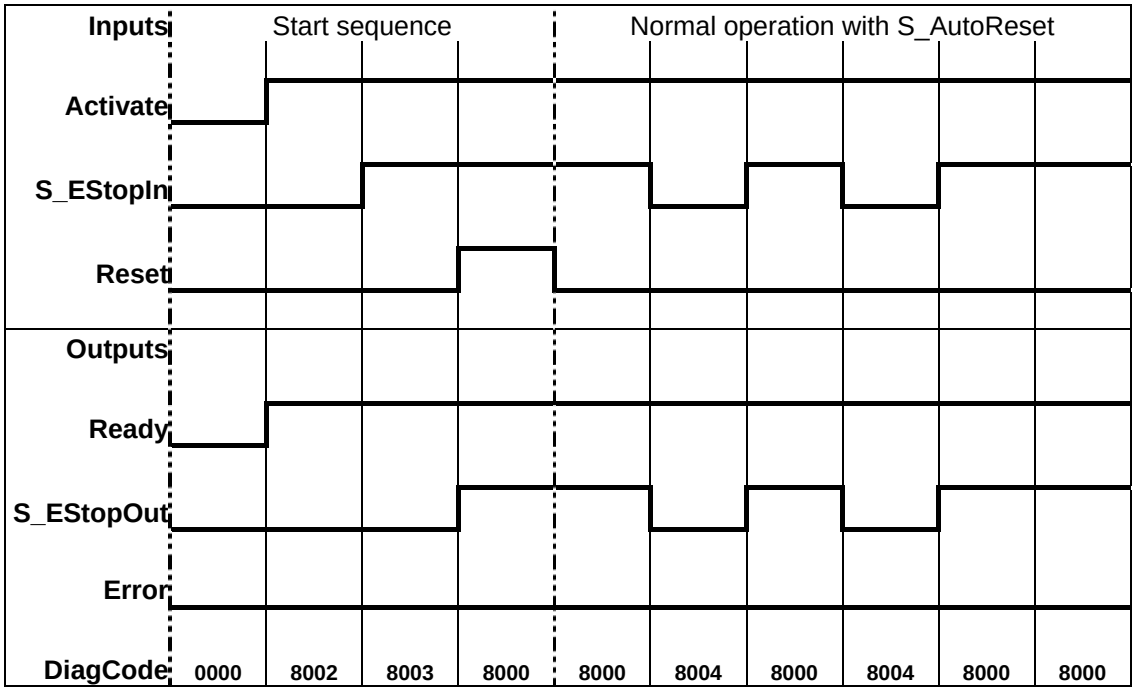


Figure 18: Timing diagram for SF_EmergencyStop: S_StartReset = FALSE, S_AutoReset = TRUE, Start, normal operation, safety demand, restart

6.4.4. Error Detection

The function block detects a static TRUE signal at Reset input.

6.4.5. Error Behavior

S_EStopOut is set to FALSE. In case of a static TRUE signal at the Reset input, the DiagCode output indicates the relevant error code and the Error output is set to TRUE.
To leave the error states, the Reset must be set to FALSE.

6.4.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C001	Reset Error 1	Reset is TRUE while waiting for S_EStopIn = TRUE. Ready = TRUE S_EStopOut = FALSE Error = TRUE
C002	Reset Error 2	Reset is TRUE while waiting for S_EStopIn = TRUE. Ready = TRUE S_EStopOut = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_EStopOut = FALSE Error = FALSE
8001	Init	Activation is TRUE. The function block was enabled. Check if S_StartReset is required. Ready = TRUE S_EStopOut = FALSE Error = FALSE

8002	Wait for S_EstopIn 1	Activation is TRUE. Check if Reset is FALSE and wait for S_EStopIn = TRUE. Ready = TRUE S_EStopOut = FALSE Error = FALSE
8003	Wait for Reset 1	Activation is TRUE. S_EStopIn = TRUE. Wait for rising trigger of Reset. Ready = TRUE S_EStopOut = FALSE Error = FALSE
8004	Wait for S_EstopIn 2	Activation is TRUE. Safety demand detected. Check if Reset is FALSE and wait for S_EStopIn = TRUE. Ready = TRUE S_EStopOut = FALSE Error = FALSE
8005	Wait for Reset 2	Activation is TRUE. S_EStopIn = TRUE. Check for S_AutoReset or wait for rising trigger of Reset. Ready = TRUE S_EStopOut = FALSE Error = FALSE
8000	Safety Output Enabled	Activation is TRUE. S_EStopIn = TRUE. Functional mode with S_EStopOut = TRUE. Ready = TRUE S_EStopOut = TRUE Error = FALSE

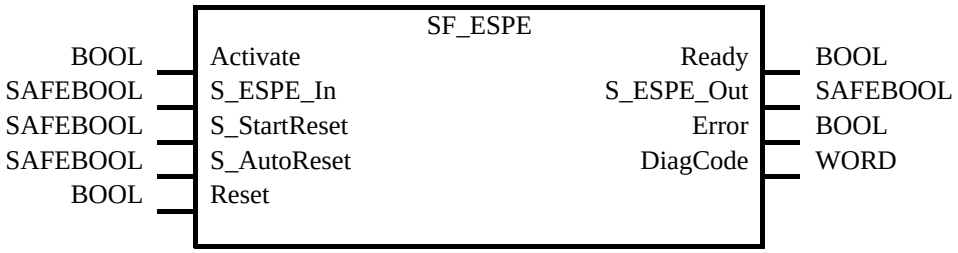
6.5. Electro-Sensitive Protective Equipment (ESPE)

6.5.1. Applicable Safety Standards

Standards	**Requirements**
EN IEC 61496-1: 2004	A.5.1 Start Interlock: The start interlock shall prevent the OSSD(s) going to the ON-state when the electrical supply is switched on, or is interrupted and restored. A.5.2: A failure of the start interlock which causes it to go to, or remain in a permanent ON-state shall cause the ESPE to go to, or to remain in the lock-out condition. A.6.1 Restart interlock: … The interlock condition shall continue until the restart interlock is manually reset. However, it shall not be possible to reset the restart interlock whilst the sensing device is actuated.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.5.2. Interface Description

FB Name	**SF_ESPE**		
This function block is a safety-related function block for monitoring electro-sensitive protective equipment (ESPE).			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_ESPE_In	SAFEBOOL	FALSE	Safety demand input. Variable. FALSE: ESPE actuated, demand for safety-related response. TRUE: ESPE not actuated, no demand for safety-related response. Safety control system must be able to detect a very short interruption of the sensor (which is specified in 61496-1: minimum 80 ms), when the ESPE is used in applications as a trip device
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
S_AutoReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_ESPE_Out	SAFEBOOL	FALSE	Output for the safety-related response. FALSE: Safety output disabled. Demand for safety-related response (e.g., reset required or internal errors active). TRUE: Safety output enabled. No demand for safety-related response.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes:			



6.5.3. Functional Description

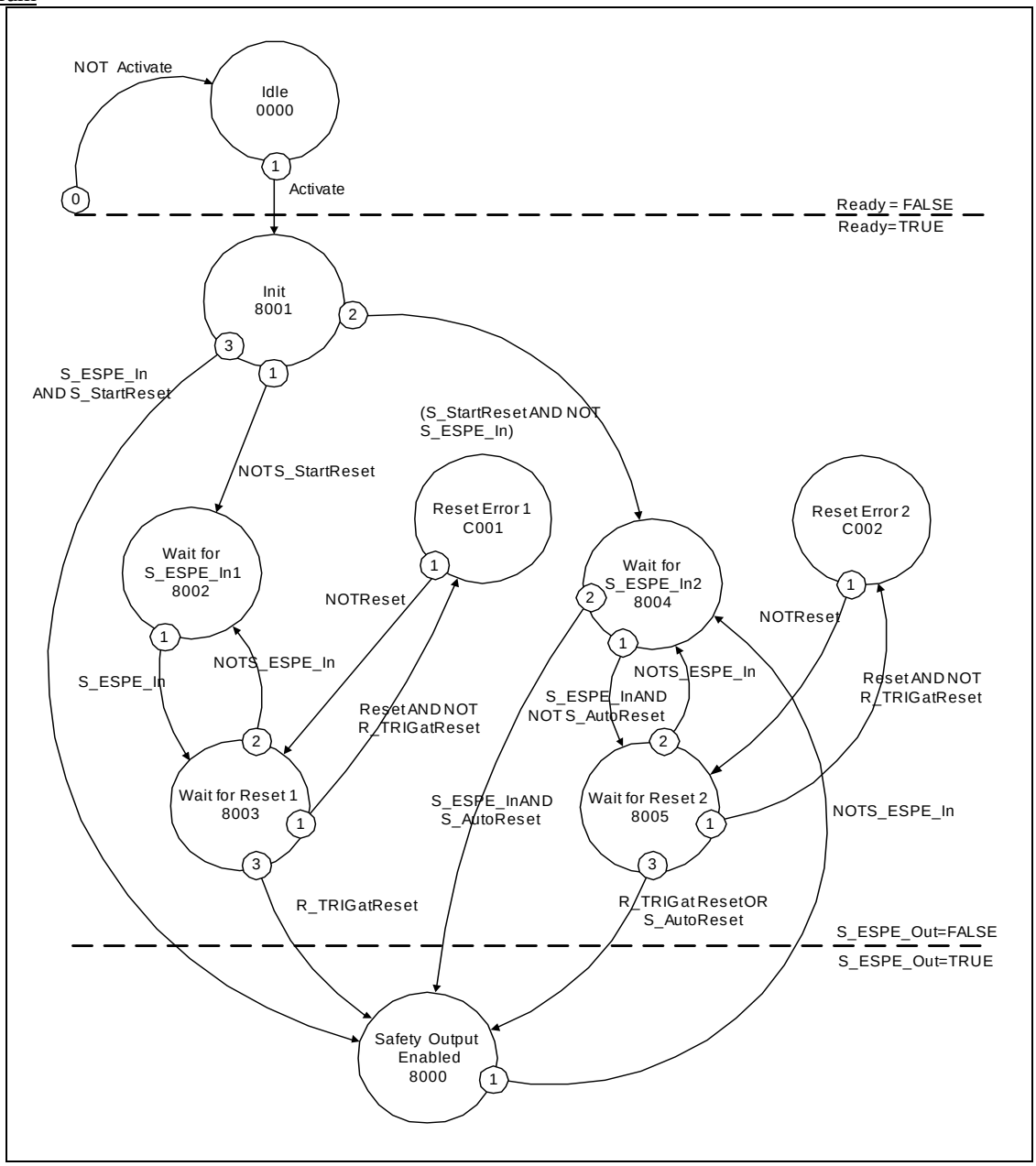
This function block is a safety-related function block for monitoring electro-sensitive protective equipment (ESPE). The func-

tion is identical to SF_EmergencyStop. The S_ESPE_Out output signal is set to FALSE as soon as the S_ESPE_In input is set to FALSE. The S_ESPE_Out output signal is set to TRUE only if the S_ESPE_In input is set to TRUE and a reset occurs. The enable reset depends on the defined S_StartReset, S_AutoReset, and Reset inputs.
If S_AutoReset = TRUE, acknowledgment is automatic.
If S_AutoReset = FALSE, a rising trigger at the Reset input must be used to acknowledge the enable.
If S_StartReset = TRUE, acknowledgment is automatic the PES is started the first time.
If S_StartReset = FALSE, a rising trigger at the Reset input must be used to acknowledge the enable.

The S_StartReset and S_AutoReset inputs shall only be activated if it is ensured, that no hazardous situation can occur when the PES is started.

The ESPE must be selected in respect of the product standards EN IEC 61496-1, -2 and -3 and the required categories according EN 954-1.

State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 19: State diagram for SF_ESPE

Typical Timing Diagrams

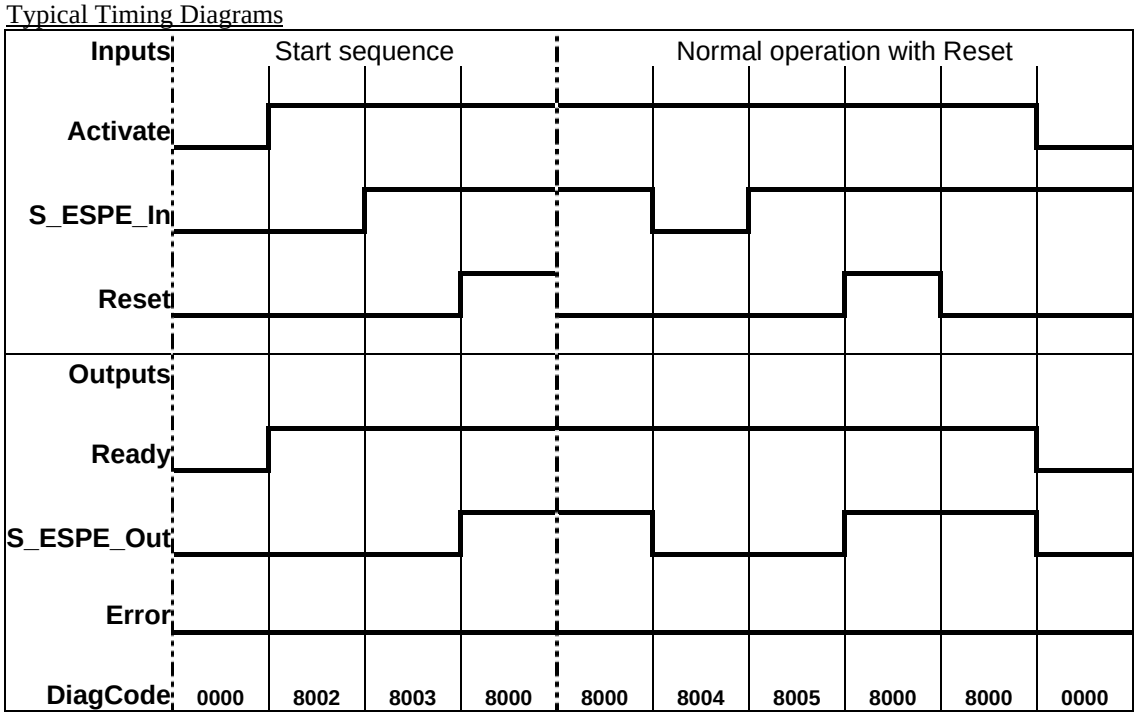


Figure 20: Timing diagram for SF_ESPE: S_StartReset = FALSE; S_AutoReset = FALSE; Start, reset, normal operation, safety demand, restart

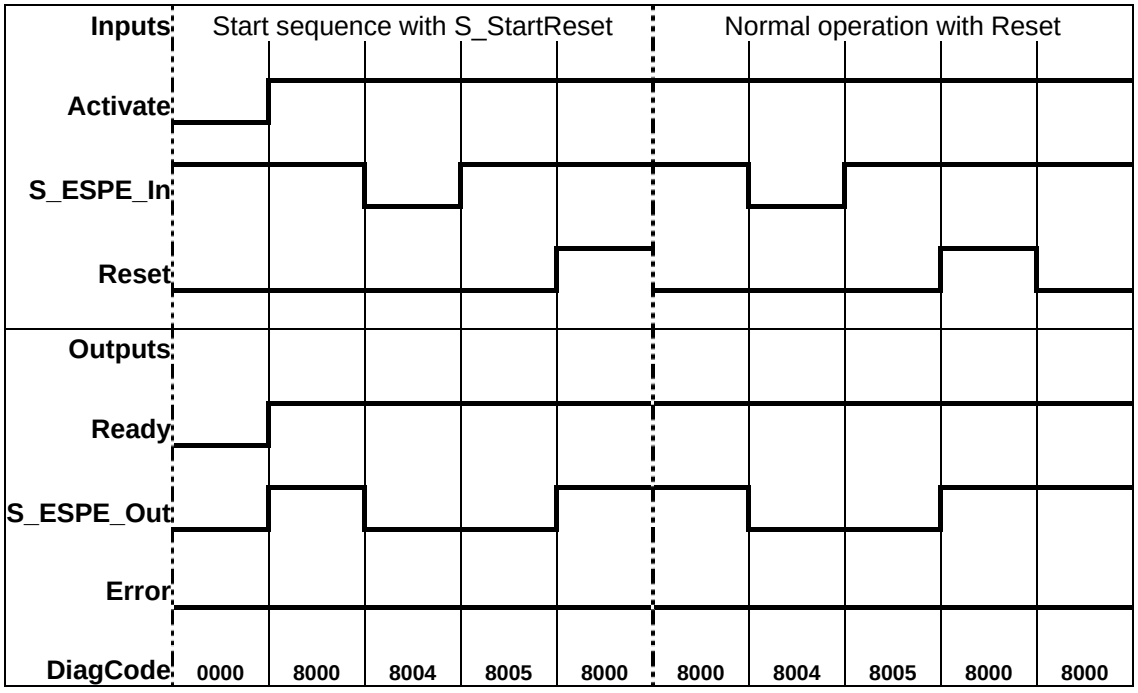


Figure 21: Timing diagram for SF_ESPE: S_StartReset = TRUE, S_AutoReset = FALSE; Start, normal operation, safety demand, restart

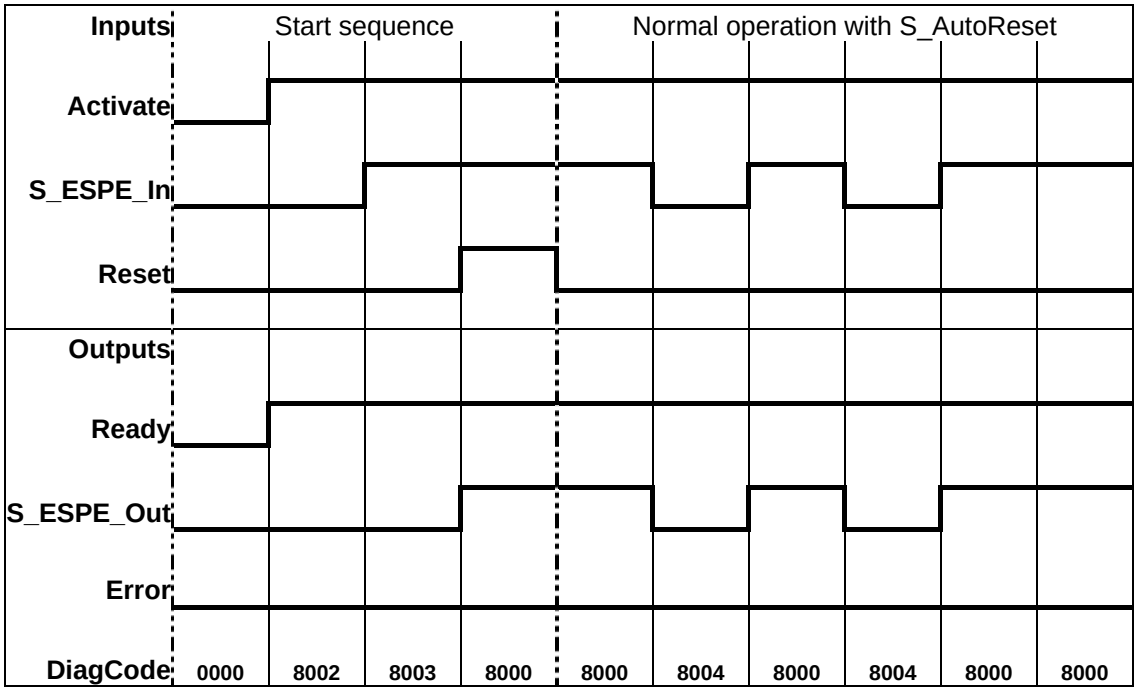


Figure 22: Timing diagram for SF_ESPE: S_StartReset = FALSE, S_AutoReset = TRUE, Start, normal operation, safety demand, restart

6.5.4. Error Detection

The function block detects a static TRUE signal at Reset input.

6.5.5. Error Behavior

S_ESPE_Out is set to FALSE. In case of a static TRUE signal at the Reset input, the DiagCode output indicates the relevant error code and the Error output is set to TRUE.
To leave the error states, the the Reset must be set to FALSE.

6.5.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C001	Reset Error 1	Reset is TRUE while waiting for S_ESPE_In = TRUE. Ready = TRUE S_ESPE_Out = FALSE Error = TRUE
C002	Reset Error 2	Reset is TRUE while waiting for S_ESPE_In = TRUE. Ready = TRUE S_ESPE_Out = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_ESPE_Out = FALSE Error = FALSE
8001	Init	Activation is TRUE. The function block was enabled. Check if S_StartReset is required. Ready = TRUE S_ESPE_Out = FALSE Error = FALSE

8002	Wait for S_ESPE_In 1	Activation is TRUE. Check if Reset is FALSE and wait for S_ESPE_In = TRUE. Ready = TRUE S_ESPE_Out = FALSE Error = FALSE
8003	Wait for Reset 1	Activation is TRUE. S_ESPE_In = TRUE. Wait for rising trigger of Reset. Ready = TRUE S_ESPE_Out = FALSE Error = FALSE
8004	Wait for S_ESPE_In 2	Activation is TRUE. Safety demand detected. Check if Reset is FALSE and wait for S_ESPE_In = TRUE. Ready = TRUE S_ESPE_Out = FALSE Error = FALSE
8005	Wait for Reset 2	Activation is TRUE. S_ESPE_In = TRUE. Check for S_AutoReset or wait for rising trigger of Reset. Ready = TRUE S_ESPE_Out = FALSE Error = FALSE
8000	Safety Output Enabled	Activation is TRUE. S_ESPE_In = TRUE. Functional mode with S_ESPE_Out = TRUE. Ready = TRUE S_ESPE_Out = TRUE Error = FALSE

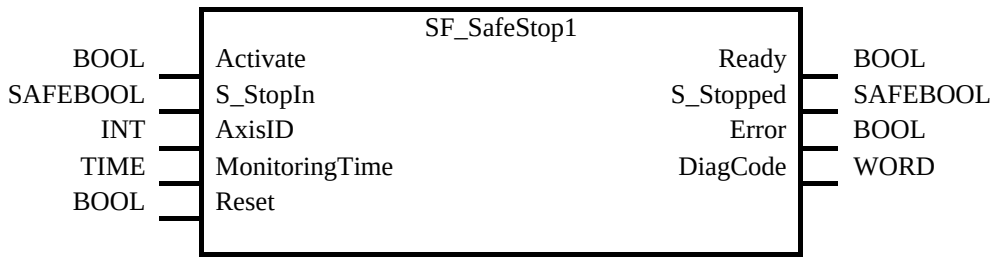
6.6. SafeStop1

6.6.1. Applicable Safety Standards

Standards	**Requirements**
CD IEC 61800-5-2: 2005	5.2.1.3 Safe Stop 1 (SS1) ..initiates and monitors (or controls) .. deceleration .. within set limits to stop the motor and initiates the STO function when the motor has stopped; or initiates the motor deceleration and initiates the STO function after an application specific time delay References to IEC 60204-1 (controlled stop in accordance with category 1) 5.2.1.2 Safe Torque Off (STO). Power, that can cause movement ... is not applied to the motor or has been removed. References to IEC 60204-1 (uncontrolled stop in accordance with category 0).
IEC 60204-1Ed. 5: 2003	9.2.2 Stop Functions Category 1: a controlled stop with power available to the machine actuators to achieve the stop and then removal of power when the stop is achieved;
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.6.2. Interface Description

FB Name	**SF_SafeStop1**		
This FB initiates a controlled stop of an electrical drive in accordance with category 1 of IEC 60204-1.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_StopIn	SAFEBOOL	FALSE	Variable. Input to request a safe stop, deriving from a safety FB. These preceeding FB's must ensure the restart inter-lock. FALSE: Stop requested. TRUE: Stop not requested.
AxisID	INT	0	Constant: Drive address. Its range is supplier specific.
MonitoringTime	TIME	T#0s	Constant: Time until the drive shall be stopped.
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_Stopped	SAFEBOOL	FALSE	Safety output indicating the motion status of the drive. FALSE: Drive is not stopped. TRUE: Drive is stopped.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: This FB provides the functionality of both Safe Stop 1 (SS1) and Safe Torque Off (STO) from standard IEC 61800-5-2.			



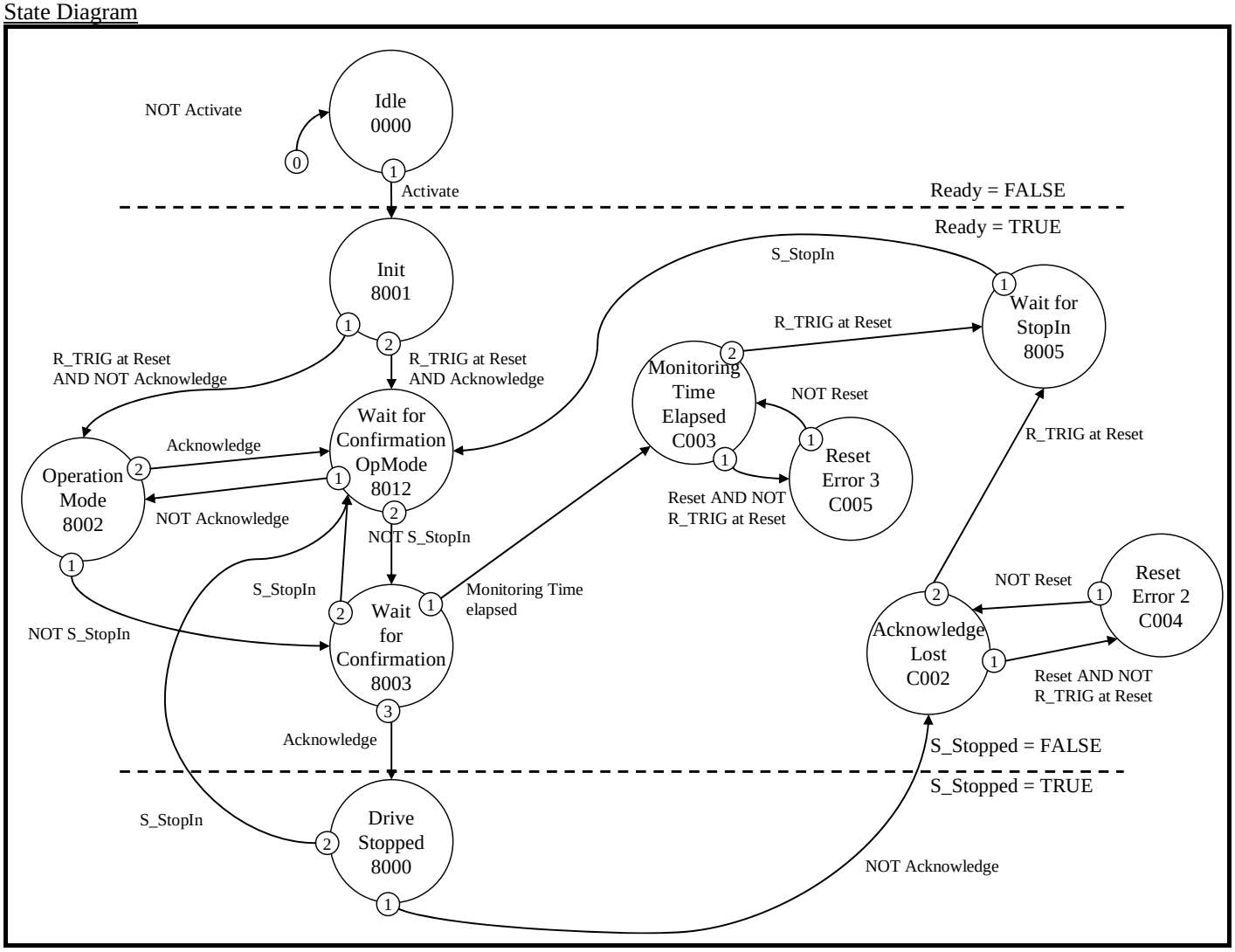
6.6.3. Functional Description

The FB initiates a controlled stop of an electrical drive in accordance with stop category 1 - with power available to the machine actuators to achieve the stop and then removal of the power when the stop is achieved.

The safety function will be provided by the drive system itself. However, the deceleration of the drive can be controlled by motion control or by the drive itself. Therefore the FB only initiates the stop, monitors it, and sets the output when the drive system acknowledges that the drive is stopped by an internal signal on the system level. This will be indicated with the "S_Stopped" output. The drive switches off the power supply via an internal switching device. The drive system may provide different stop modes (speed-controlled, torque-controlled, etc.). Therefore the drive provides parameters, which have to be set in the drive system. If the drive does not acknowledge the controlled stop within the MonitoringTime, it should be assumed that the drive system is not able to carry out the request for some reason and may be on any speed level (S_Stopped remains FALSE). However, the drive system itself may initiate a supplier-specific failure response, which may result in switching off the power supply as a last resort. (Depending on the risk analysis further measures may be required.)

Note:
a) Drives which do not support this functionality shall not be controlled by this FB. For these drives the FB SF_SafetyRequest is provided.
b) Category 0 and/or category 1 and/or category 2 stops shall be provided as indicated by the risk assessment and the functional requirements of the machine. Additional protective devices or interlocks may be required.
c) The drive must ensure that the stop function overrides any related start functions.
d) Emergency stop: The restart interlock must be handled by the SF_EmergencyStop function block. The drive must ensure that the stop function overrides all other functions.
e) It shall not be possible to restart the machine until all emergency stop commands have been reset. This must be locked by the application program.
f) The application program is responsible for indicating to the supervisory control and/or machines that a stop condition exists.

State Diagram



Note 1: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).
Note 2: The signal “Acknowledge” means the internal acknowledge from the drive.
Figure 23: State diagram for SF_SafeStop1

Typical Timing Diagrams

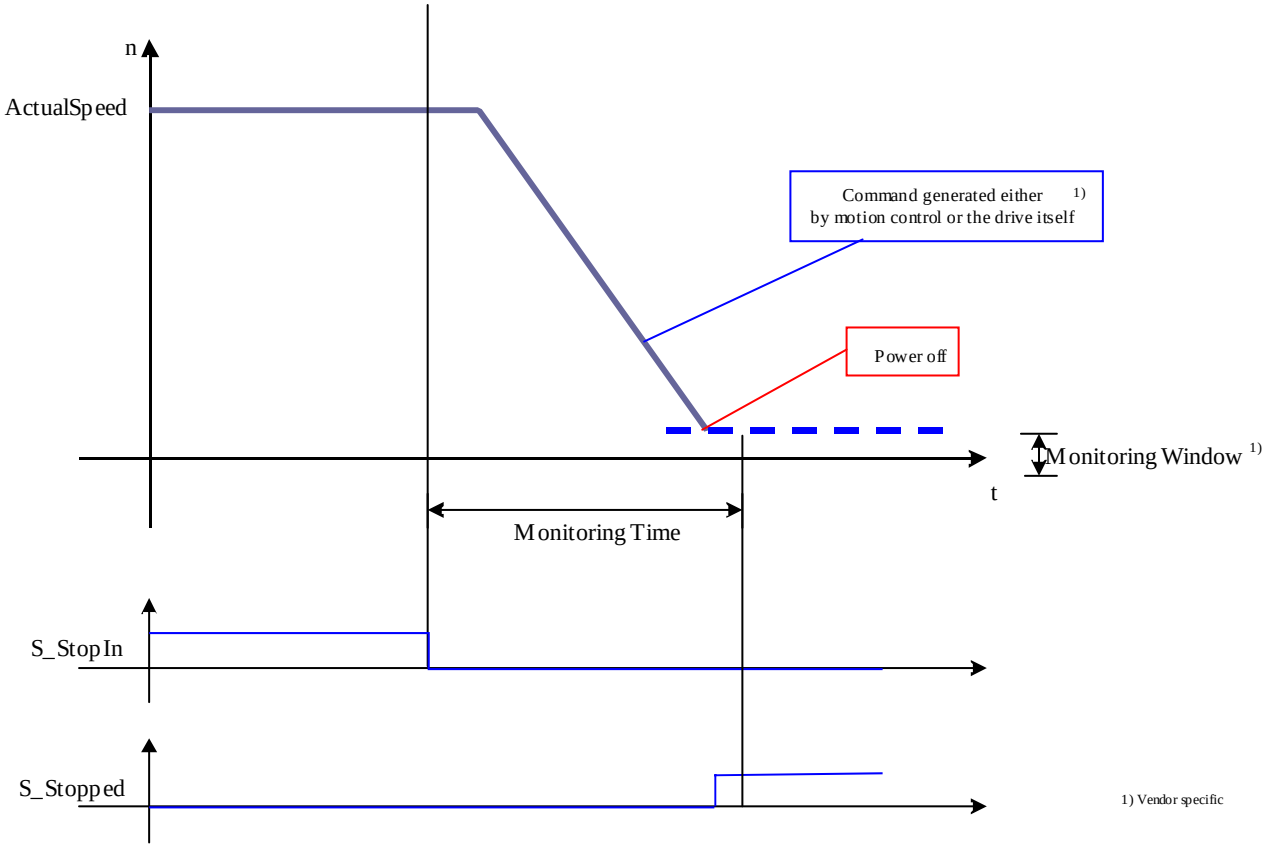


Figure 24: Process timing diagram for SF_SafeStop1

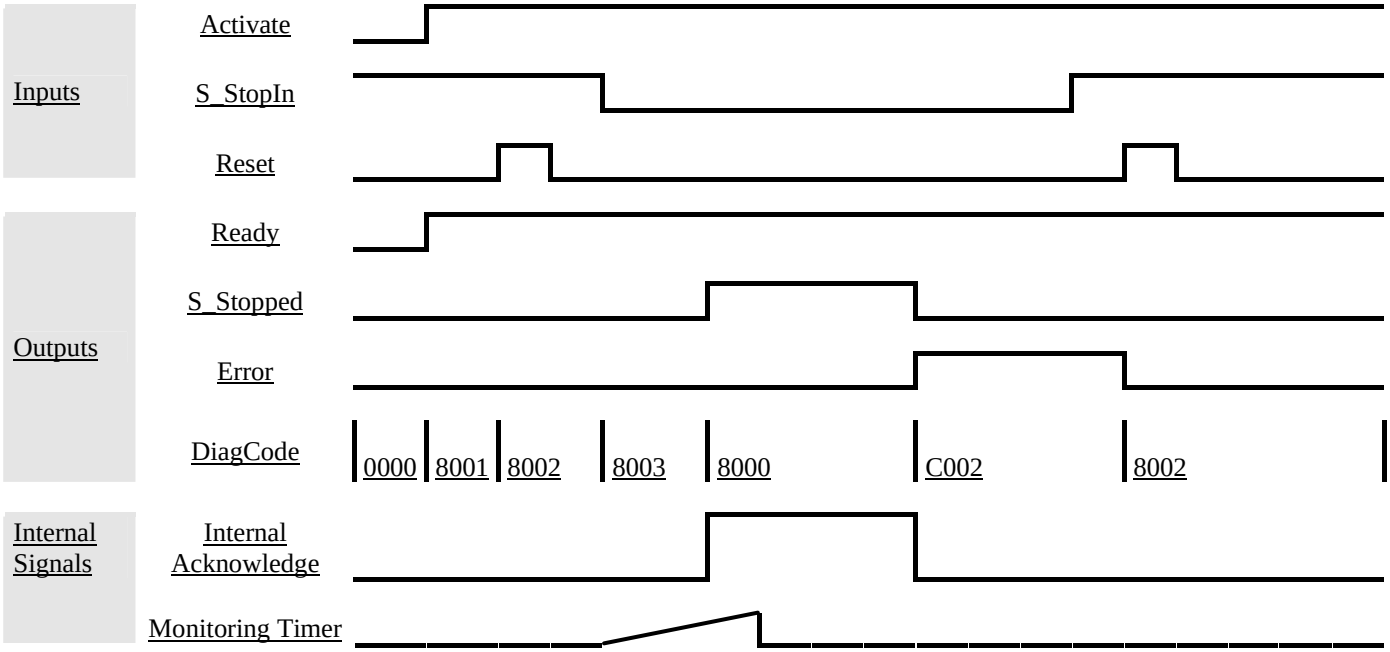


Figure 25: Timing diagram for SF_SafeStop1

6.6.4. Error Detection

The FB detects whether the drive is not stopped within the monitoring time.
The FB detects whether the acknowledge signal is lost while the request is still active.
The FB detects a static Reset signal.

6.6.5. Error Behavior

In the event of an error, the S_Stopped output is set to FALSE.
An error must be acknowledged by a rising trigger at the Reset input. To continue the FB after this reset, the S_StopIn request must be set to TRUE.

6.6.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C002	Acknowledge Lost	Drive was stopped and stop is still requested, however the Acknowledge signal is lost. Ready = TRUE S_Stopped = FALSE Error = TRUE
C003	Monitoring Time Elapsed	The drive was not stopped within the monitoring time. Ready = TRUE S_Stopped = FALSE Error = TRUE
C004	Reset Error 2	Static Reset detected in state C002. Ready = TRUE S_Stopped = FALSE Error = TRUE
C005	Reset Error 3	Static Reset detected in state C003. Ready = TRUE S_Stopped = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_Stopped = FALSE Error = FALSE
8000	Drive Stopped	Drive is stopped. Ready = TRUE S_Stopped = TRUE Error = FALSE
8001	Init	State after Activate is set to TRUE. Ready = TRUE S_Stopped = FALSE Error = FALSE
8002	Operation Mode	Any drive mode (operation or any other safe mode). Ready = TRUE S_Stopped = FALSE Error = FALSE
8012	Wait for Confirmation OpMode	Wait for Acknowledge that the drive is in operation mode. Ready = TRUE S_Stopped = FALSE Error = FALSE
8003	Wait for Confirmation	Waiting for confirmation from the drive (system interface). Ready = TRUE S_Stopped = FALSE Error = FALSE

8005	Wait for StopIn	Error was reset. However, S_StopIn must be set to TRUE before the FB can be initialized. Ready = TRUE S_Stopped = FALSE Error = FALSE

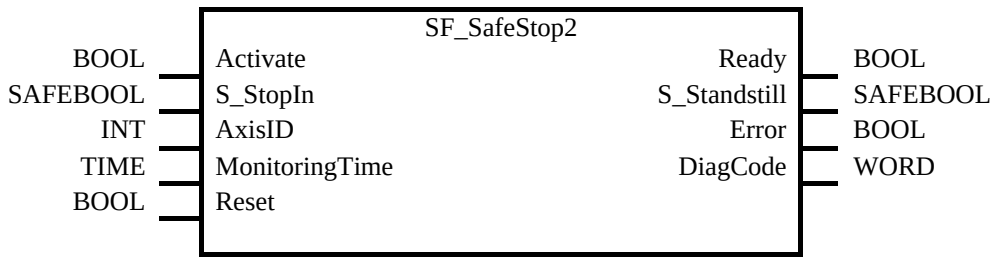
6.7. SafeStop2

6.7.1. Applicable Safety Standards

Standards	**Requirements**
CD IEC 61800-5-2: 2005	5.2.1.4 Safe Stop 2 (SS2) ..initiates and monitors (or controls) .. deceleration .. within set limits to stop the motor and initiates the Safe Operating Stop (SOS) function when the motor has stopped; or initiates the motor deceleration and initiates the SOS function after an application specific time delay 5.2.2.1 Safe Operating Stop (SOS) Ensures that the motor remains stopped by resisting external forces Safe Stop 2 in combination with Safe Operating Stop corresponds to IEC 60204-1 (controlled stop in accordance with category 2)
IEC 60204-1: 2003	9.2.2 Stop Functions Category 2: a controlled stop with power left available to the machine actuators.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.7.2. Interface Description

FB Name	**SF_SafeStop2**		
This FB initiates a stop of an electrical drive in accordance with stop category 2 of IEC 60204-1.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_StopIn	SAFEBOOL	FALSE	Variable. Input to request a safe operational stop, deriving from a safety FB. These preceeding FB's must ensure the re-start interlock. FALSE: Stop requested. TRUE: Stop not requested.
AxisID	INT	0	Constant. Drive address. Its range is supplier specific.
MonitoringTime	TIME	T#0s	Constant. Time until the drive shall be stopped.
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_Standstill	SAFEBOOL	FALSE	Safety output indicating the motion status of the drive. FALSE: Drive is not at a controlled standstill. TRUE: Drive is at a controlled standstill.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: This FB provides the function of both Safe Stop 2 (SS2) and Safe Operating Stop (SOS) from standard IEC 61800-5-2.			

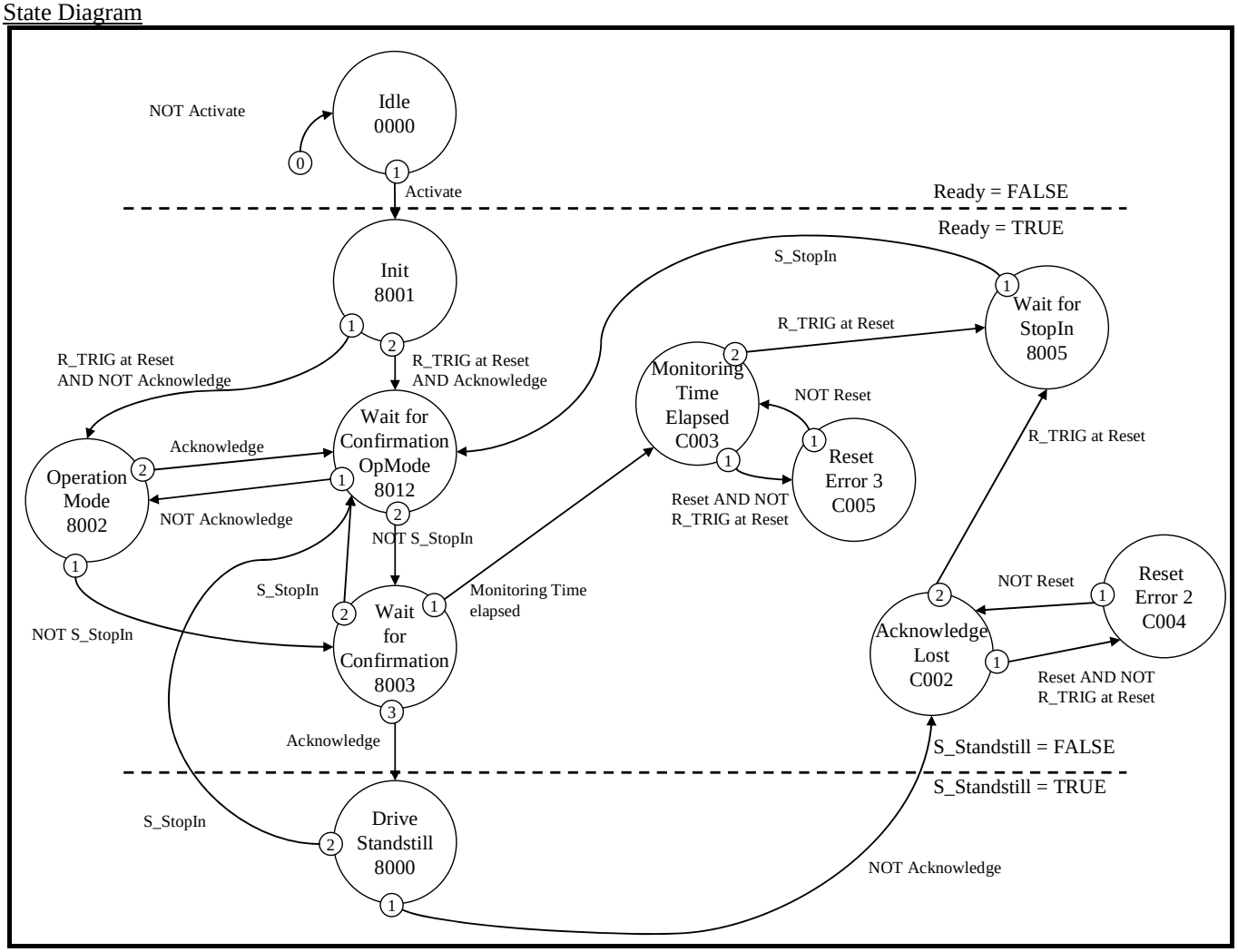


6.7.3. Functional Description

The FB initiates a controlled stop of an electrical drive in accordance with stop category 2 - with power available to the machine actuators to achieve the stop and controlled standstill when the stop is achieved.
The safety function will be provided by the drive system itself. However, the deceleration of the drive is controlled by motion control. Therefore the FB only initiates the stop, monitors it, and sets the output when the drive system acknowledges that the drive is in a controlled standstill by an internal signal on the system level. This will be indicated with the "S_Standstill" output. If the drive does not acknowledge a controlled standstill within the MonitoringTime, it should be assumed that the drive system is not able to carry out the request for some reason and may be on any speed level (S_Standstill remains FALSE). However, the drive system itself may initiate a supplier-specific failure response, which may result in switching off the power supply as a last resort. (Depending on the risk analysis further measures may be required.)

Notes:
a) Drives which do not support this functionality shall not be controlled by this FB. For these drives the FB SF_SafetyRequest is provided.
b) Category 0 and/or category 1 and/or category 2 stops shall be provided as indicated by the risk assessment and the functional requirements of the machine. Additional protective devices or interlocks might be required.
c) The drive must ensure that the stop function overrides any related start functions.
d) The application program is responsible for indicating to the supervisory control and/or machines that a stop condition exists.

State Diagram



Note 1: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).
Note 2: The signal “Acknowledge” means the internal acknowledge from the drive.
Figure 26: State diagram for SF_SafeStop2

Typical Timing Diagrams

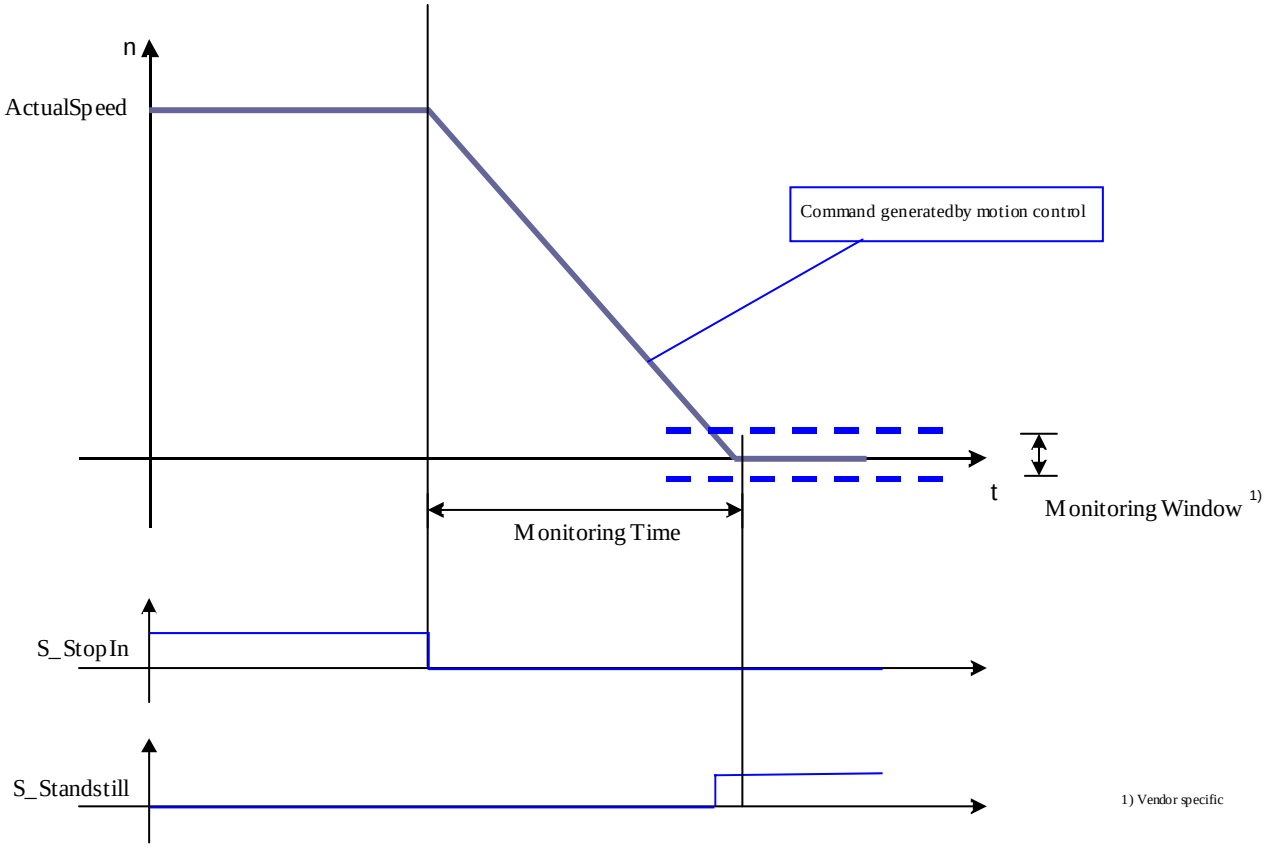


Figure 27: Process timing diagram for SF_SafeStop2

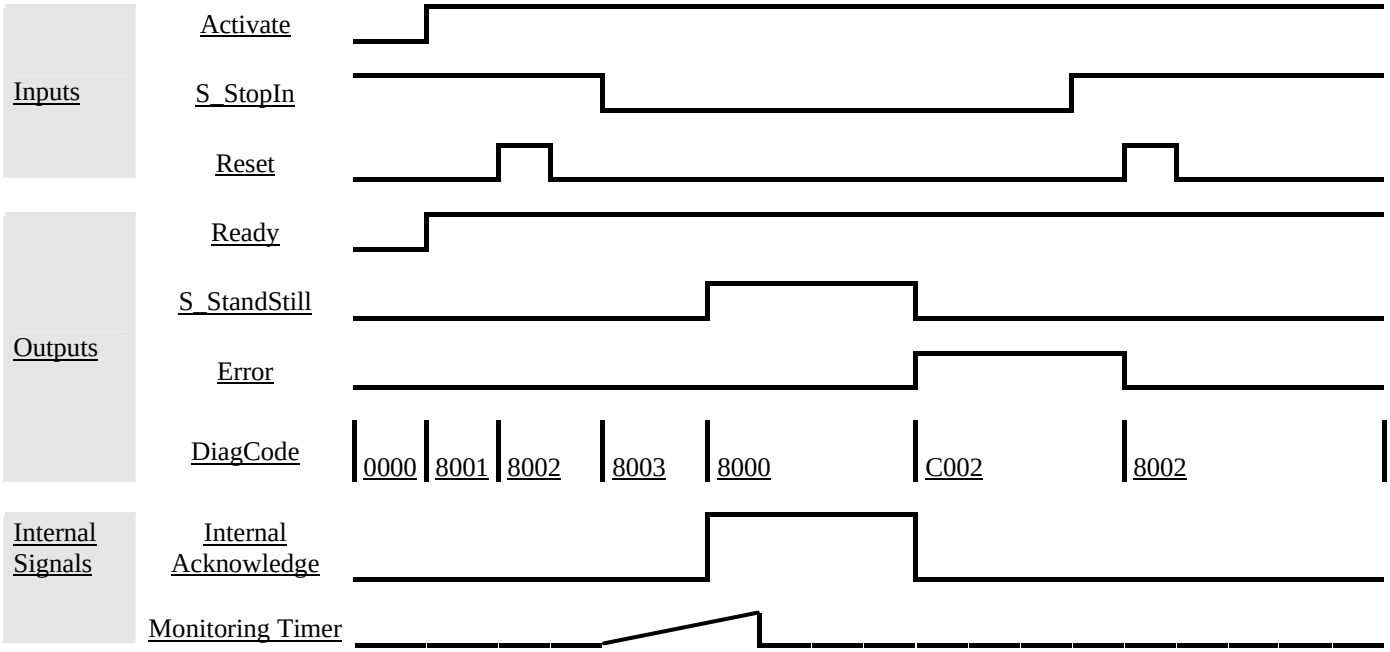


Figure 28: Timing diagram for SF_SafeStop2

6.7.4. Error Detection

The FB detects whether the drive does not come to a standstill within the monitoring time.
The FB detects whether the acknowledge signal is lost while SafeOperationalStop is still requested.
The FB detects a static Reset signal.

6.7.5. Error Behavior

In the event of an error, the S_Standstill output will be FALSE.
An error must be acknowledged by a rising trigger at the Reset input. To continue the FB after this reset, the S_StopIn request must be set to TRUE.

6.7.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C002	Acknowledge Lost	Drive was at a standstill and standstill is still requested, however, the Acknowledge signal is lost. Ready = TRUE S_Standstill = FALSE Error = TRUE
C003	Monitoring Time Elapsed	The drive was not at a standstill within the monitoring time. Ready = TRUE S_Standstill = FALSE Error = TRUE
C004	Reset Error 2	Static Reset detected in state C002 (Acknowledge Lost). Ready = TRUE S_Standstill = FALSE Error = TRUE
C005	Reset Error 3	Static Reset detected in state C003 (MonitoringTime elapsed). Ready = TRUE S_Standstill = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_Standstill = FALSE Error = FALSE
8000	Drive Standstill	Drive is at a controlled standstill. Ready = TRUE S_Standstill = TRUE Error = FALSE
8001	Init	State after Activate is set to TRUE. Ready = TRUE S_Standstill = FALSE Error = FALSE
8002	Operation Mode	Any drive mode (operation or any other safe mode). Ready = TRUE S_Standstill = FALSE Error = FALSE
8012	Wait for Confirmation OpMode	Wait for Acknowledge that the drive is in operation mode. Ready = TRUE S_Standstill = FALSE Error = FALSE
8003	Wait for Confirmation	Waiting for confirmation from the drive (system interface). Ready = TRUE S_Standstill = FALSE Error = FALSE

8005	Wait for StopIn	Error was reset. However, S_StopIn must be set to TRUE before the FB can be initialized. Ready = TRUE S_Standstill = FALSE Error = FALSE

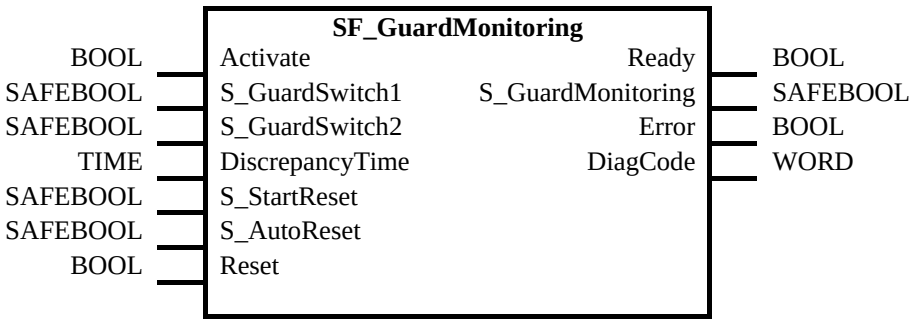
6.8. Safety Guard Monitoring

6.8.1. Applicable Safety Standards

Standards	**Requirements**
EN 953: 1997	3.3.3 Control Guard – The hazardous machine functions "covered" by the guard cannot operate until the guard is closed; – Closing the guard initiates operation of the hazardous machine function(s).
EN 1088: 1995	3.2 Interlocking Guard – The hazardous machine functions "covered" by the guard cannot operate until the guard is closed; – If the guard is opened while the hazardous machine functions are operating, a stop instruction is given; – When the guard is closed, the hazardous machine functions "covered" by the guard can operate, but the closure of the guard does not by itself initiate their operation.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4 Restart following power failure/spontaneous restart

6.8.2. Interface Description

FB Name	**SF_GuardMonitoring**		
This function block monitors the relevant safety guard. There are two independent input parameters for two switches at the safety guard coupled with a time difference (MonitoringTime) for closing the guard.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_GuardSwitch1	SAFEBOOL	FALSE	Variable. Guard switch 1 input. FALSE: Guard is open. TRUE: Guard is closed.
S_GuardSwitch2	SAFEBOOL	FALSE	Variable. Guard switch 2 input. FALSE: Guard is open. TRUE: Guard is closed.
DiscrepancyTime	TIME	T#0ms	Constant. Configures the monitored synchronous time between S_GuardSwitch1 and S_GuardSwitch2.
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters – Only Constant
S_AutoReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters – Only Constant
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_GuardMonitoring	SAFEBOOL	FALSE	Output indicating the status of the guard. FALSE: Guard is not active. TRUE: both S_GuardSwitches are TRUE, no error and acknowledgment. Guard is active.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: -			



6.8.3. Functional Description SF_GuardMonitoring

The function block requires two inputs indicating the guard position for safety guards with two switches (according to EN 1088), a DiscrepancyTime input and Reset input. If the safety guard only has one switch, the S_GuardSwitch1 and S_GuardSwitch2 inputs can be bridged. The monitoring time is the maximum time required for both switches to respond when closing the safety guard. The Reset, S_StartReset, and S_AutoReset inputs determine how the function block is reset after the safety guard has been opened.

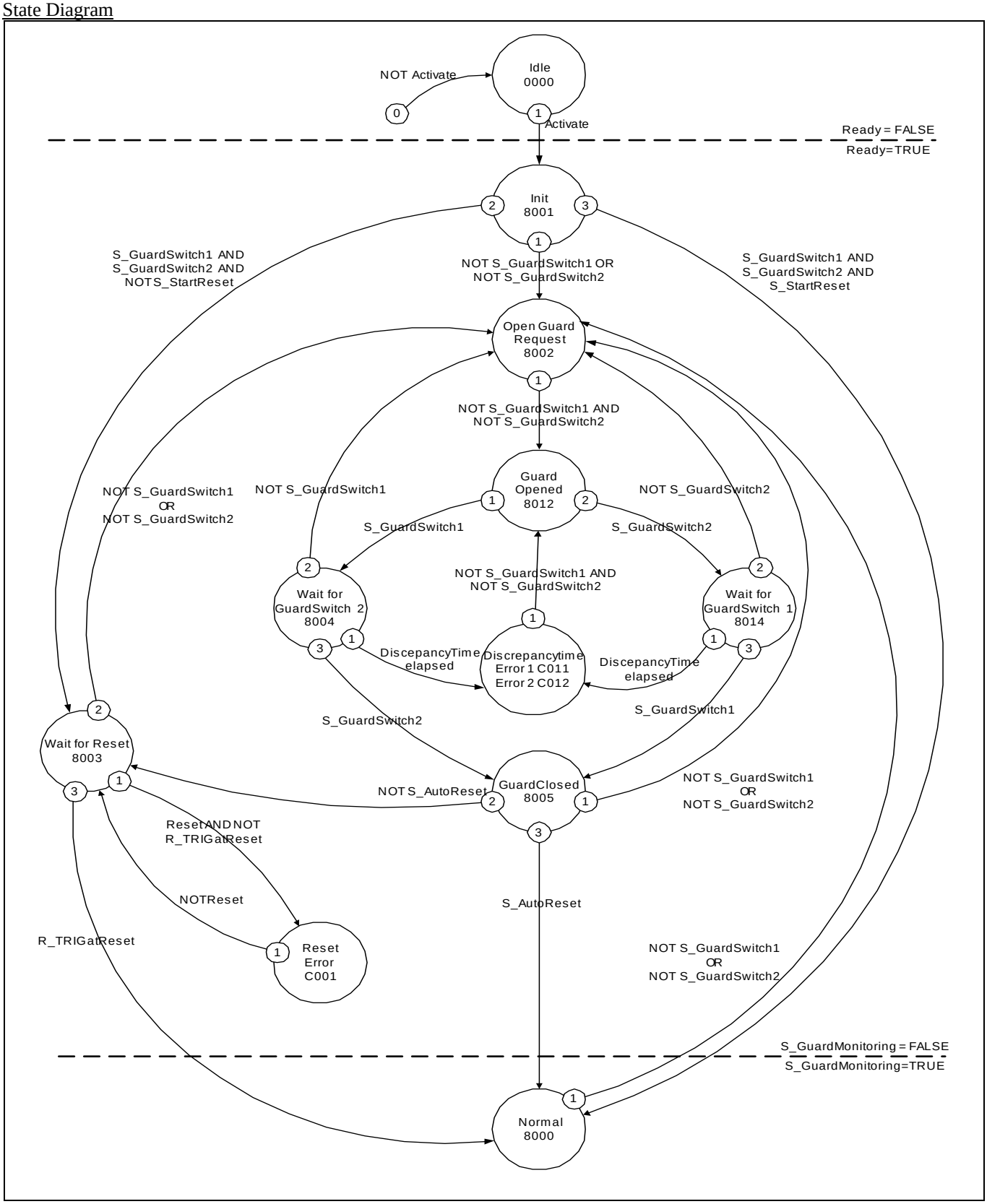
When opening the safety guard, both S_GuardSwitch1 and S_GuardSwitch2 inputs should switch to FALSE. The S_GuardMonitoring output switches to FALSE as soon as one of the switches is set to FALSE. When closing the safety guard, both S_GuardSwitch1 and S_GuardSwitch2 inputs should switch to TRUE.

This FB monitors the symmetry of the switching behavior of both switches. The S_GuardMonitoring output remains FALSE if only one of the contacts has completed an open/close process.

The behavior of the S_GuardMonitoring output depends on the time difference between the switching inputs. The discrepancy time is monitored as soon as the value of both S_GuardSwitch1/S_GuardSwitch2 inputs differs. If the DiscrepancyTime has elapsed, but the inputs still differ, the S_GuardMonitoring output remains FALSE. If the second corresponding S_GuardSwitch1/S_GuardSwitch2 input switches to TRUE within the value specified for the DiscrepancyTime input, the S_GuardMonitoring output is set to TRUE following acknowledgment.

The S_StartReset and S_AutoReset inputs shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

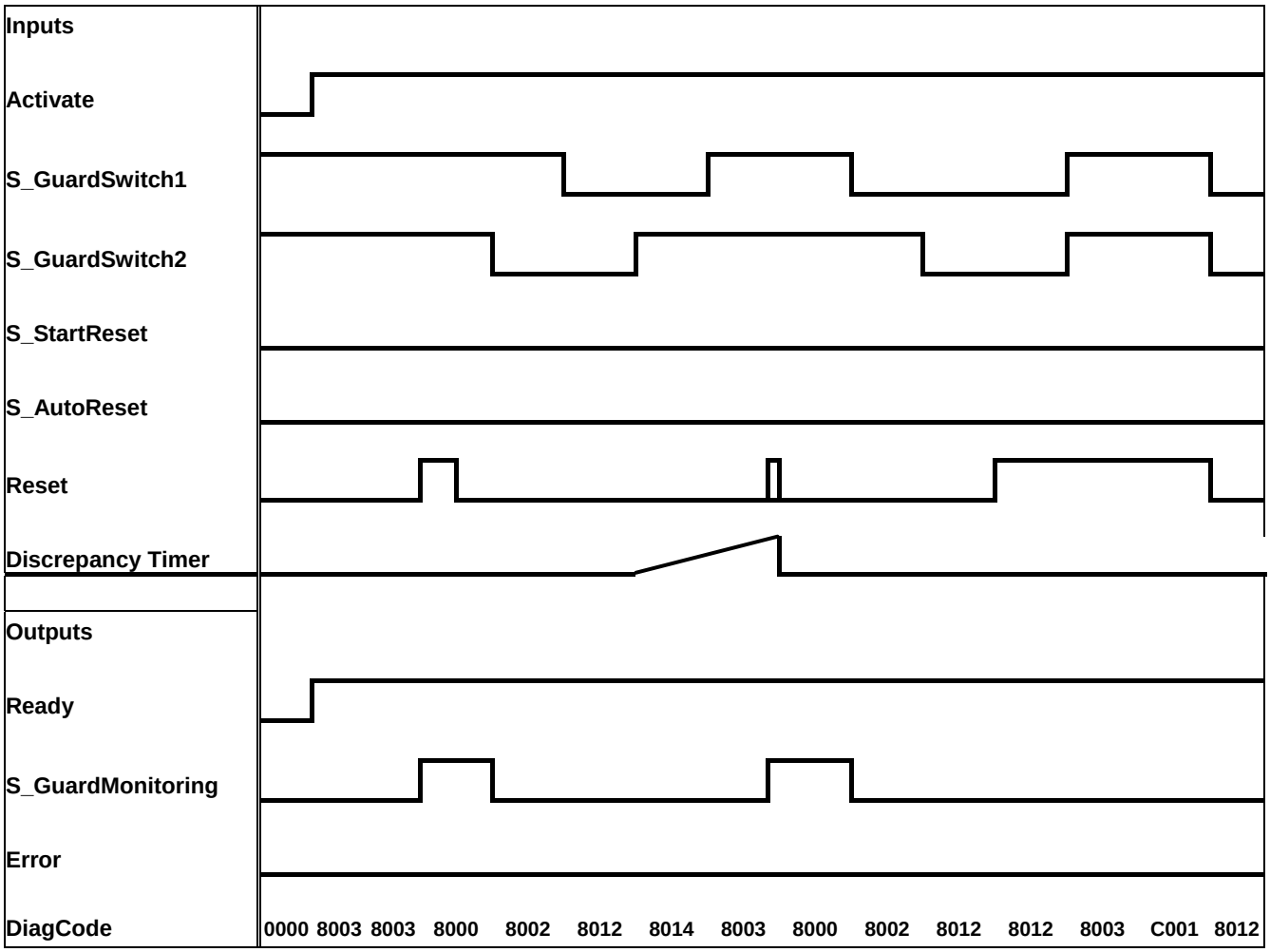
State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 29: State diagram for SF_GuardMonitoring

Typical Timing Diagrams



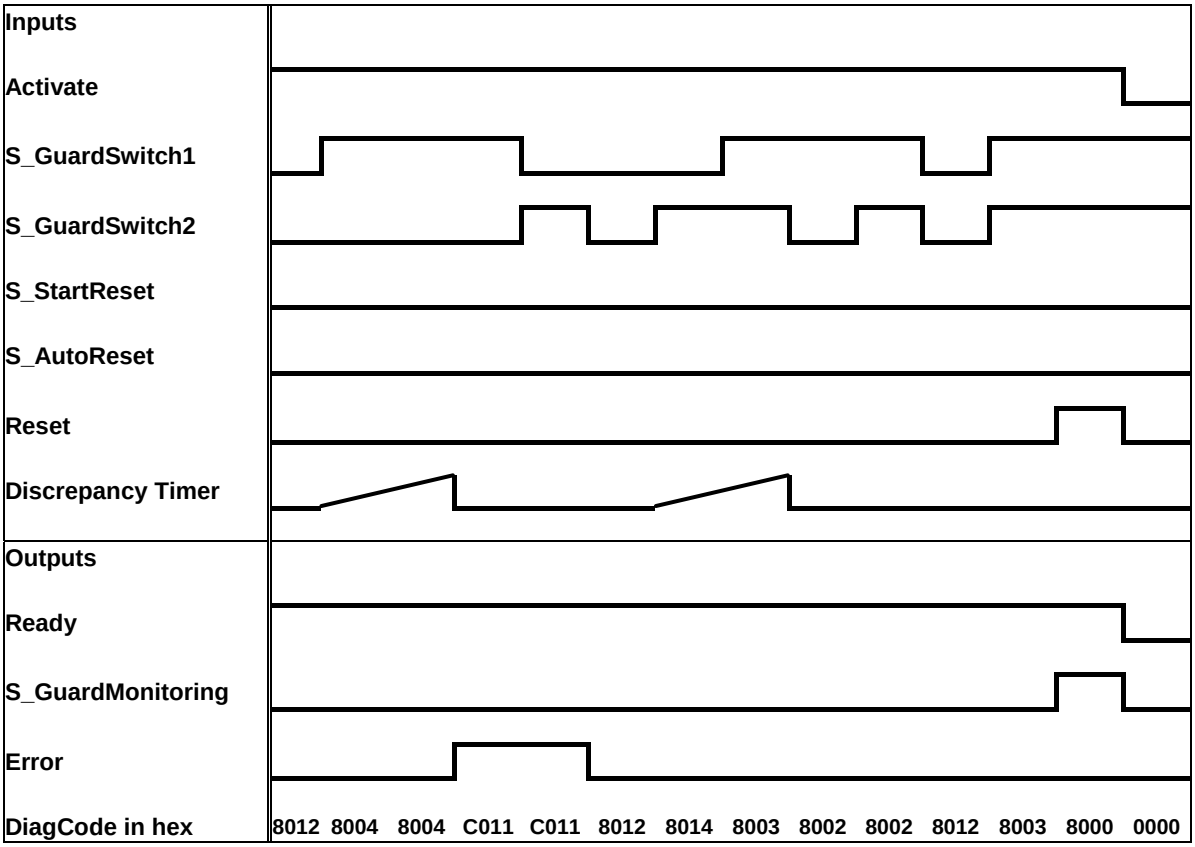


Figure 30: Timing diagrams for SF_GuardMonitoring

6.8.4. Error Detection

External signals: SAFEBOOL inputs provide inherent error detection. Mechanical setup combines that of an opening and closing switch according to EN 954 (safety guard with two switches). Discrepancy time monitoring for time lag between both mechanical switches reaction, according to EN 954 (to be considered as "application error" detection, i.e., generated by the application).
An error is detected if the time lag between the first S_GuardSwitch1/S_GuardSwitch2 input and the second is greater than the value for the DiscrepancyTime input. The Error output is set to TRUE.
The function block detects a static TRUE signal at the RESET input.

6.8.5. Error and Reset Behavior

The S_GuardMonitoring output is set to FALSE. If the two S_GuardSwitch1 and S_Guardswitch2 inputs are bridged, no error is detected. To leave the Reset error state, the Reset input must be set to FALSE. To leave the discrepancy time errors, the inputs S_GuardSwitch1 and 2 must both be set to FALSE.

6.8.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C001	Reset Error	Static reset detected in state 8003. Ready = TRUE S_GuardMonitoring = FALSE Error = TRUE
C011	Discrepancytime Error 1	DiscrepancyTime elapsed in state 8004. Ready = TRUE S_GuardMonitoring = FALSE Error = TRUE

C012	Discrepancytime Error 2	DiscrepancyTime elapsed in state 8014. Ready = TRUE S_GuardMonitoring = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_GuardMonitoring = FALSE Error = FALSE
8000	Normal	Safety guard closed and Safe state acknowledged. Ready = TRUE S_GuardMonitoring = TRUE Error = FALSE
8001	Init	Function block has been activated. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
8002	Open Guard Request	Complete switching sequence required. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
8003	Wait for Reset	Waiting for rising trigger at Reset. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
8012	Guard Opened	Guard completely opened. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
8004	Wait for GuardSwitch2	S_GuardSwitch1 has been switched to TRUE - waiting for S_GuardSwitch2; discrepancy timer started. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
8014	Wait for GuardSwitch1	S_GuardSwitch2 has been switched to TRUE - waiting for S_GuardSwitch1; discrepancy timer started. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
8005	Guard Closed	Guard closed. Waiting for Reset, if S_AutoReset = FALSE. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE

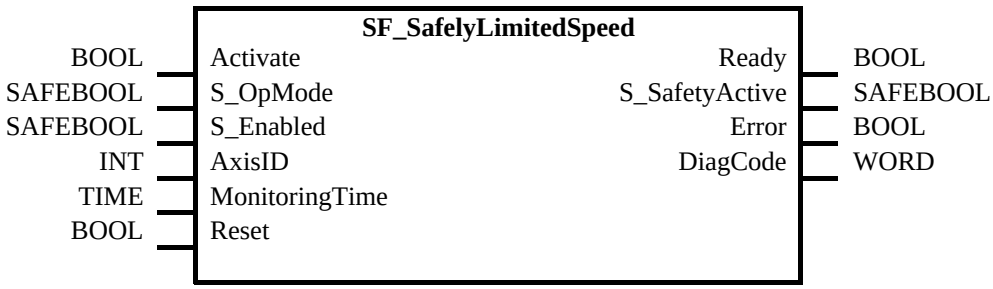
6.9. Safely Limited Speed (SLS)

6.9.1. Applicable Safety Standards

Standards	**Requirements**
CD IEC 61800-5-2: 2005	5.2.2.2 Safely-Limited Speed (SLS) The SLS function prevents the motor from exceeding the specified speed limit.
IEC 60204-1Ed.5: 2003	9.2.6.3 Enabling control Enabling control (see also 10.9) is a manually activated control function interlock that: a) when activated allows a machine operation to be initiated by a separate start control, and b) when de-activated – initiates a stop function, and – prevents initiation of machine operation.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.9.2. Interface Description

FB Name	**SF_SafelyLimitedSpeed**		
This function block provides the interface for the *safely limited speed* motion-axis-specific safety function.It does not initiate any movement of the motor, but activates the safely limited speed monitoring in the drive.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_OpMode	SAFEBOOL	FALSE	Variable. Selection of operation mode. FALSE: Safe operation mode is selected. TRUE: Deselect Safe operation mode of the axis and switch the motion control axis to the operation mode.
S_Enabled	SAFEBOOL	FALSE	Variable. Permits axis movement. Typically connected to the output S_EnableSwitchOut. FALSE: Movement cannot be enabled in Safe operation mode. TRUE: Safe axis movement is permitted.
AxisID	INT	0	Constant. Unique axis ID, axis address (e.g., SERCOS). Must be a constant value when applied in user mode. Its range is supplier specific.
MonitoringTime	TIME	T#0s	Constant. Monitoring the response time between the safety function request (S_OpMode is set to FALSE) and the internal acknowledgment (which sets S_SafetyActive to TRUE).
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_SafetyActive	SAFEBOOL	FALSE	Indicates the state of the axis. FALSE: The axis is not in a safe state. TRUE: The axis is in a safe state.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: The initiation of the movement is done by the functional application.			



6.9.3. Functional Description Including Safe State Description

The function block acts as the interface between the application program and the system environment. The supplier-specific details of the axis safety function are implemented on the system level and are hidden from the application programmer.

The safety function will be provided by the drive system itself. Therefore the FB only initiates the request, monitors it, and sets the output when the drive system acknowledges the Safe state. This will be indicated with the "S_SafetyActive" output.

This FB does not define any drive system-specific parameters. They should have been specified in the drive system itself. It switches the motion axis from a "non-safe" state to a safe state.

The request of the safe mode must be acknowledged by the drive system within the specified MonitoringTime.

Drives which do not support this function shall not be controlled by this function block. For these drives the FB SF_SafetyRequest is provided.

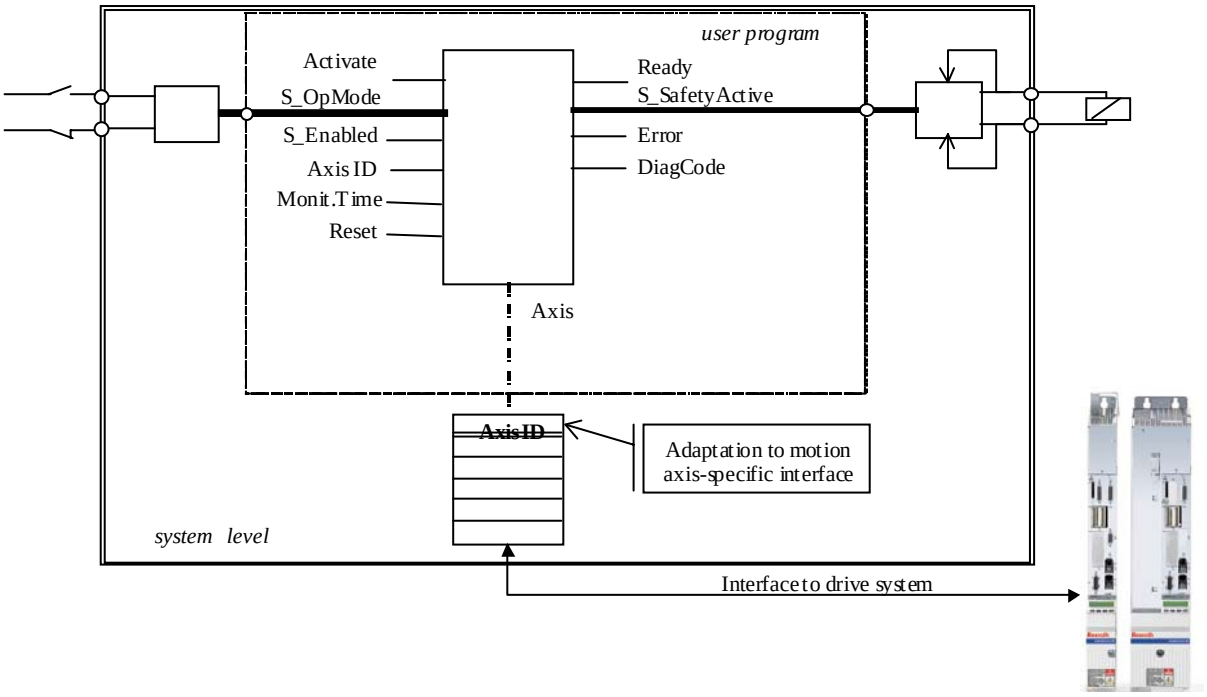
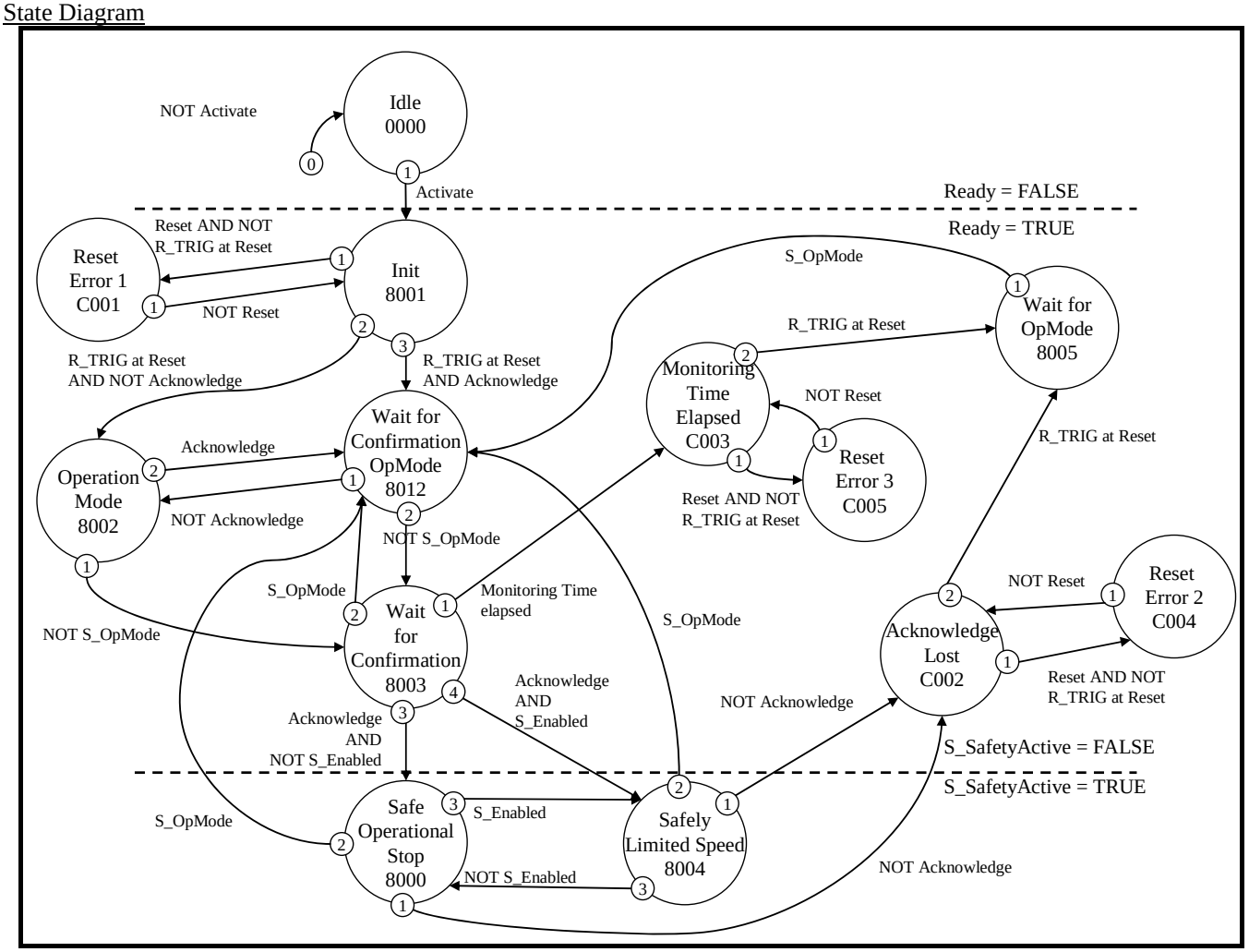


Figure 31: Example for SF_SafelyLimitedSpeed

Depending on the application, the following application-related safety standards must be considered:

Standards	Requirements
EN 201	Rubber and plastics machines - Injection molding machines - Safety requirements
prEN 691	Woodworking machines
EN 775	Manipulating industrial robots
prEN 1010	Safety of machinery - Safety requirements for the design and construction of printing and paper converting machines
EN 1921	Industrial automation systems - Safety of integrated manufacturing systems
EN 12415	Machine tools - Safety - Small numerically controlled turning machines and turning centers
prEN 12417	Machine tools - Safety - Machining centers
EN 12478	Safety of machine tools - Large numerically controlled turning machines and turning centers
EN 12840	Safety of machine-tools - Manually controlled turning machines with or without automatic control

State Diagram



Note 1: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).
Note 2: The signal “Acknowledge” means the internal acknowledge from the drive.

Figure 32: State diagram for SF_SafelyLimitedSpeed

Typical Timing Diagrams

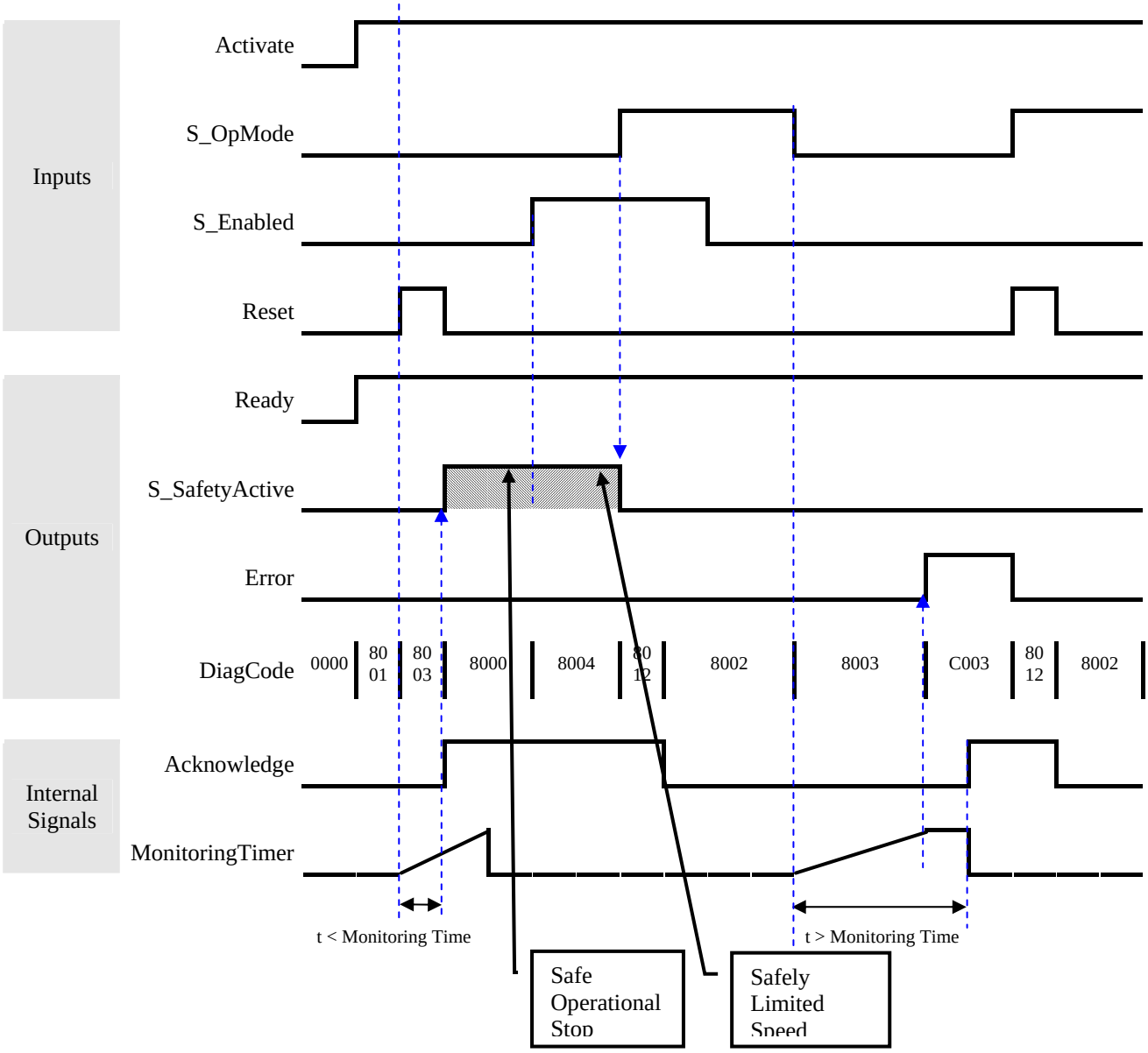


Figure 33: Timing diagram for SF_SafelyLimitedSpeed

6.9.4. Error Detection

Internal FB errors: The transition from a "non-safe" state to the Safe state is monitored by the MonitoringTime input. If the request could not be completed within the specified time this leads to an internal FB error.
The FB detects whether the acknowledge signal is lost while the request is still active. The FB detects a static Reset signal.
External FB errors: External errors are generated by the motion axis controller and are reported via the DiagCode output.

6.9.5. Error Behavior

In the event of an error the S_SafetyActive output is set to FALSE, the DiagCode output indicates the relevant error code, and the Error output is set to TRUE. An error must be acknowledged by a rising trigger at the Reset input. To continue the function block after this reset, the S_OpMode request must be set to TRUE.
To leave the Reset error state, the Reset input must be set to FALSE.

6.9.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C001	Reset Error 1	Static Reset detected in Init state. Ready = TRUE S_ SafetyActive = FALSE Error = TRUE
C002	Acknowledge Lost	Acknowledgment lost while in the Safe state. Ready = TRUE S_SafetyActive = FALSE Error = TRUE
C003	MonitoringTime Elapsed	S_OpMode request could not be completed within the monitoring time. Ready = TRUE S_SafetyActive = FALSE Error = TRUE
C004	Reset Error 2	Static Reset detected in state C002 (Acknowledge Lost). Ready = TRUE S_ SafetyActive = FALSE Error = TRUE
C005	Reset Error 3	Static Reset detected in state C003 (MonitoringTime elapsed). Ready = TRUE S_ SafetyActive = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_SafetyActive = FALSE Error = FALSE
8000	Safe Operational Stop	Drive stopped. Ready = TRUE S_SafetyActive = TRUE Error = FALSE
8001	Init	State after Activate is set to TRUE or after a rising trigger at Reset. Ready = TRUE S_SafetyActive = FALSE Error = FALSE
8002	Operation Mode	Operation mode. Ready = TRUE S_SafetyActive = FALSE Error = FALSE
8012	Wait for Confirmation OpMode	Wait for Acknowledge that the drive is in operation mode. Ready = TRUE S_SafetyActive = FALSE Error = FALSE
8003	Wait for Confirmation	Waiting for confirmation from the drive (system interface). Ready = TRUE S_SafetyActive = FALSE Error = FALSE
8004	Safely Limited Speed	Drive at safely limited (reduced) speed. Ready = TRUE S_SafetyActive = TRUE Error = FALSE
8005	Wait for OpMode	Error was cleared. However S_OpMode must be set to TRUE before the FB can enter operation mode. Ready = TRUE S_SafetyActive = FALSE Error = FALSE

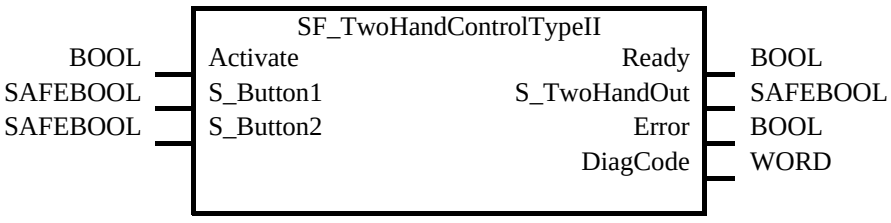
6.10. Two-Hand Control Type II

6.10.1. Applicable Safety Standards

Standards	**Requirements**
EN 574: 1996	Clause 4, Table 1, Type II. 5.1 Use of both hands / simultaneous actuation. 5.2 Relationship between output signal and input signals. 5.3 Completion of the output signal. 5.6 Reinitiation of the output signal. 6.3 Use of DIN EN 954-1 category 3 (Can only be realized by NO and NC switches together with antivalent processing)
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.10.2. Interface Description

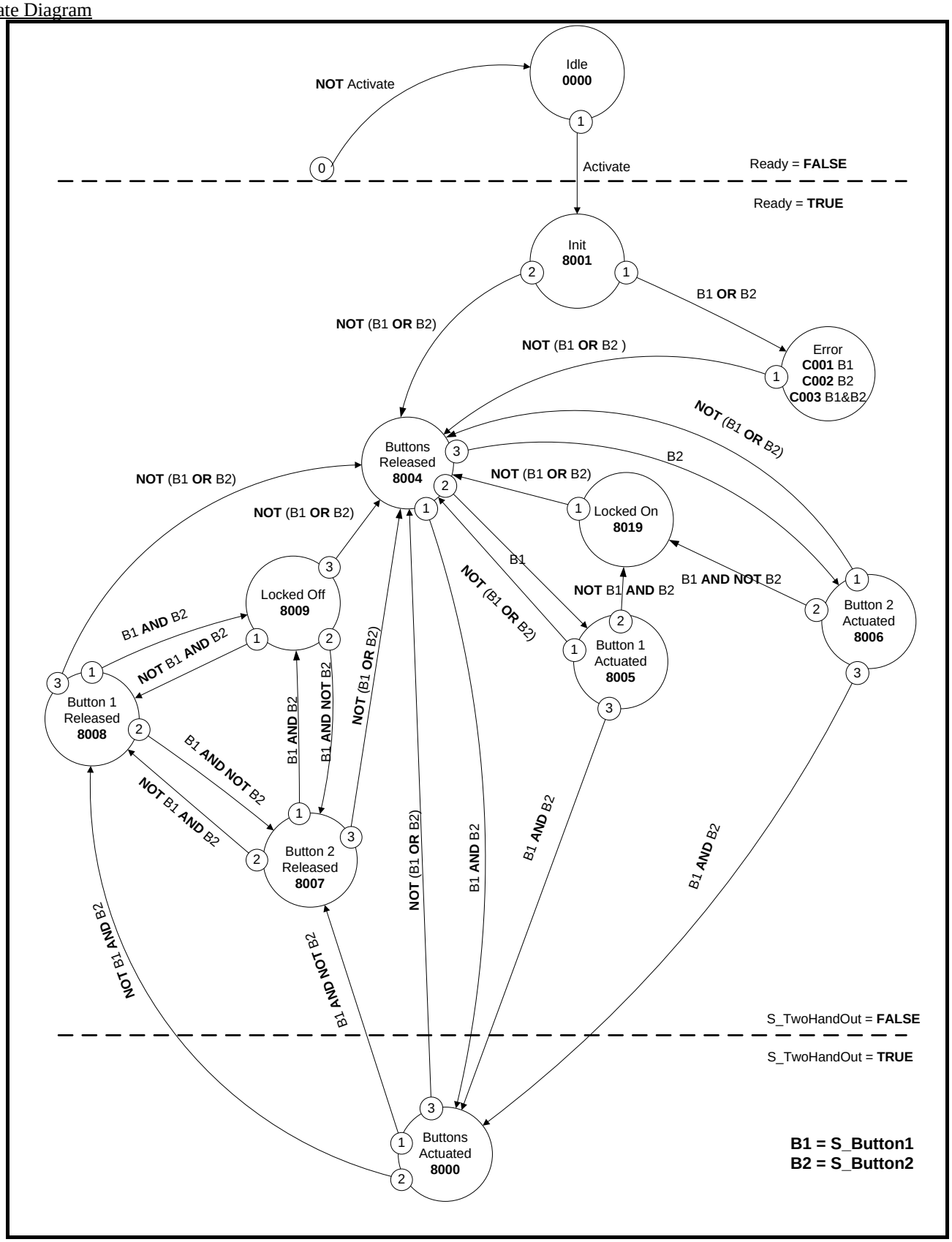
FB Name	**SF_TwoHandControlTypeII**		
This function block provides the two-hand control functionality (see EN 574, Section 4 Type II).			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_Button1	SAFEBOOL	FALSE	Variable. Input of button 1 (for category 3 or 4: two antivalent contacts) FALSE: Button 1 released. TRUE: Button 1 actuated.
S_Button2	SAFEBOOL	FALSE	Variable. Input of button 2 (for category 3 or 4: two antivalent contacts) FALSE: Button 2 released. TRUE: Button 2 actuated.
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_TwoHandOut	SAFEBOOL	FALSE	Safety related output signal. FALSE: No correct two hand operation. TRUE: S_Button1 and S_Button2 inputs are TRUE and no error occurred. Correct two hand operation.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: No Reset input or Error output is required, because no test can be performed on both switches.			



6.10.3. Functional Description

This function block provides the two-hand control functionality according to EN 574, Section 4 Type II. If S_Button1 and S_Button2 are set to TRUE in correct sequence, then the S_TwoHandOut output will also be set to TRUE. The FB also controls the release of both buttons before setting the output S_TwoHandOut again to TRUE.

State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 34: State diagram for SF_TwoHandControlTypeII

Typical Timing Diagram

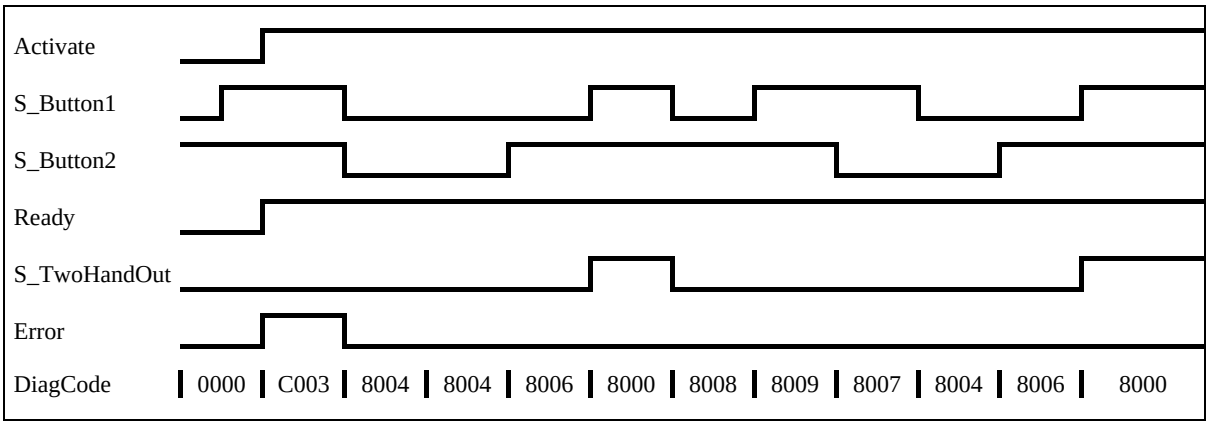


Figure 35: Timing diagram for SF_TwoHandControlTypeII

6.10.4. Error Detection

After activation of the FB, any button set to TRUE is detected as an invalid input setting leading to an error.

6.10.5. Error Behavior

In the event of an error, the S_TwoHandOut output is set to FALSE and remains in this safe state.
The Error state is exited when both buttons are released (set to FALSE).

6.10.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	Output Setting

FB-specific error codes:

C001	Error B1	S_Button1 was TRUE on FB activation. Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
C002	Error B2	S_Button2 was TRUE on FB activation. Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
C003	Error B1&B2	The signals at S_Button1 and S_Button2 were TRUE on FB activation. Ready = TRUE Error = TRUE S_TwoHandOut = FALSE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE Error = FALSE S_TwoHandOut = FALSE
8000	Buttons Actuated	Both buttons actuated correctly. The safety related output is enabled. Ready = TRUE Error = FALSE S_TwoHandOut = TRUE
8001	Init	Function block is active, but in the Init state. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE

8004	Buttons Released	No button is actuated. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8005	Button 1 Actuated	Only Button 1 is actuated. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8006	Button 2 Actuated	Only Button 2 is actuated. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8007	Button 2 Released	The safety related output was enabled and is disabled again. FALSE at both S_Button1 and S_Button2 was not achieved after disabling the safety related output. In this state, S_Button1 is TRUE and S_Button2 is FALSE after disabling the safety related output. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8008	Button 1 Released	The safety related output was enabled and is disabled again. FALSE at both S_Button1 and S_Button2 was not achieved after disabling the safety related output. In this state, S_Button1 is FALSE and S_Button2 is TRUE after disabling the safety related output. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8009	Locked Off	The safety related output was enabled and is disabled again. FALSE at both S_Button1 and S_Button2 was not achieved after disabling the safety related output. In this state, S_Button1 is TRUE and S_Button2 is TRUE after disabling the safety related output. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8019	Locked On	Incorrect actuation of the buttons. Waiting for release of both buttons. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE

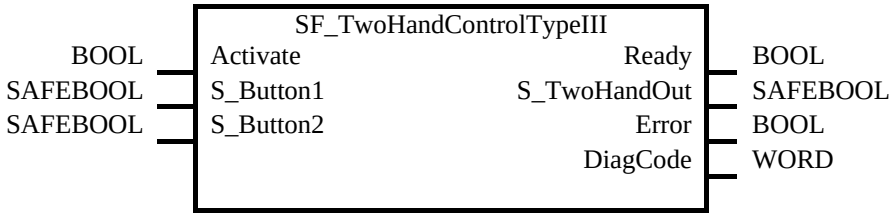
6.11. Two-Hand Control Type III

6.11.1. Applicable Safety Standards

Standards	**Requirements**
EN 574: 1996	Clause 4, Table 1, Type III A; B; C. 5.1 Use of both hands / simultaneous actuation. 5.2 Relationship between output signal and input signals. 5.3 Completion of the output signal. 5.6 Reinitiation of the output signal. 5.7 Synchronous actuation. 6.2 Use of DIN EN 954-1 category 1. 6.3 Use of DIN EN 954-1 category 3. (Can only be realized by NO and NC switches together with antivalent processing) 6.4 Use of DIN EN 954-1 category 4. (Can only be realized by NO and NC switches together with antivalent processing)
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.11.2. Interface Description

FB Name	**SF_TwoHandControlTypeIII**		
This function block provides the two-hand control functionality (see EN 574, Section 4 Type III. Fixed specified time difference is 500 ms).			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_Button1	SAFEBOOL	FALSE	Variable. Input of button 1 (for category 3 or 4: two antivalent contacts) FALSE: Button 1 released. TRUE: Button 1 actuated.
S_Button2	SAFEBOOL	FALSE	Variable. Input of button 2 (for category 3 or 4: two antivalent contacts) FALSE: Button 2 released. TRUE: Button 2 actuated.
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_TwoHandOut	SAFEBOOL	FALSE	Safety related output signal. FALSE: No correct two hand operation. TRUE: S_Button1 and S_Button2 inputs changed from FALSE to TRUE within 500 ms and no error occurred. The two hand operation has been performed correctly.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: No Reset input or Error output is required, because no test can be performed on both switches.			

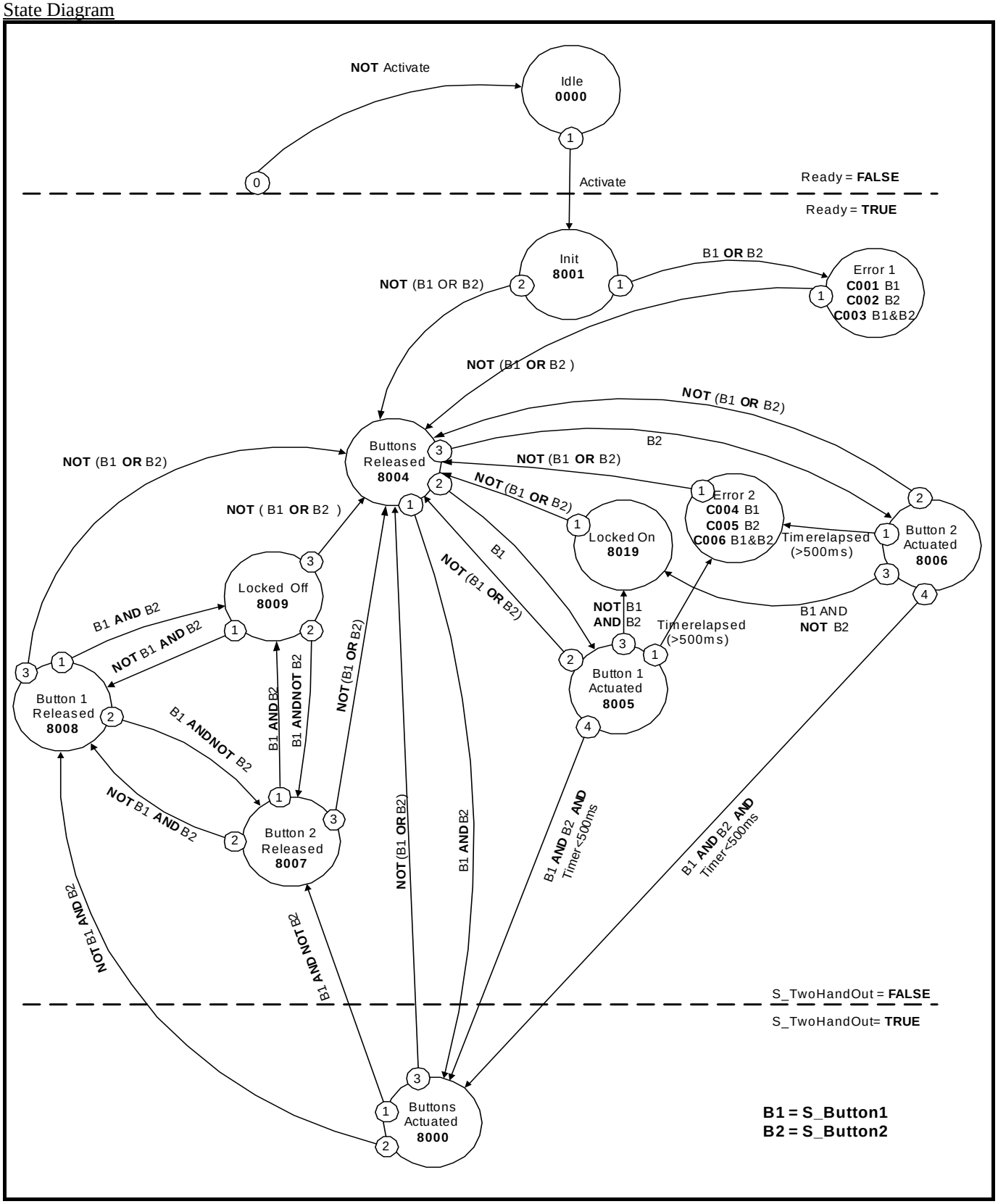


6.11.3. Functional Description

This function block provides the two-hand control functionality according to EN 574, Section 4 Type III. If S_Button1 and S_Button2 are set to TRUE within 500 ms and in correct sequence, then the S_TwoHandOut output is also set to TRUE. The

FB also controls the release of both buttons before setting the output S_TwoHandOut again to TRUE.

State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 36: State diagram for SF_TwoHandControlTypeIII

Typical Timing Diagram

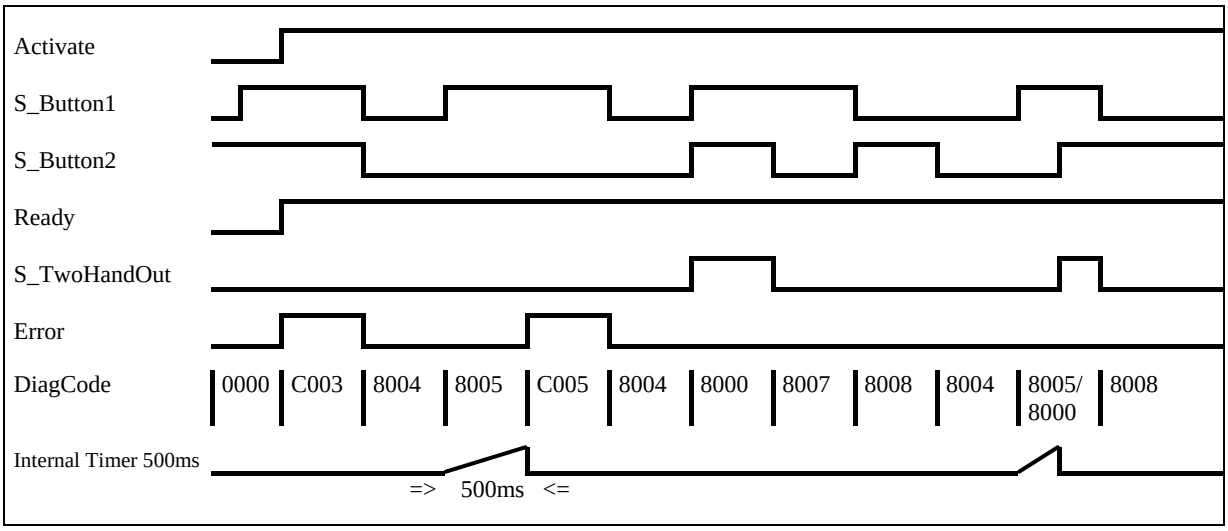


Figure 37: Timing diagram for SF_TwoHandControlTypeIII

6.11.4. Error Detection

After activation of the FB, any button set to TRUE is detected as an invalid input setting leading to an error. The FB detects when the divergence of the input signals exceeds 500 ms.

6.11.5. Error Behavior

In the event of an error, the S_TwoHandOut output is set to FALSE and remains in this safe state.
The Error state is exited when both buttons are released (set to FALSE).

6.11.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	Output Setting

FB-specific error codes:

C001	Error 1 B1	S_Button1 was TRUE on FB activation. Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
C002	Error 1 B2	S_Button2 was TRUE on FB activation. Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
C003	Error 1 B1&B2	The signals at S_Button1 and S_Button2 were TRUE on FB activation. Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
C004	Error 2 B1	S_Button1 was FALSE and S_Button 2 was TRUE after 500 ms in state 8005. Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
C005	Error 2 B2	S_Button1 was TRUE and S_Button 2 was FALSE after 500 ms in state 8005. Ready = TRUE Error = TRUE S_TwoHandOut = FALSE

C006	Error 2 B1&B2	S_Button1 was TRUE and S_Button 2 was TRUE after 500 ms in state 8005 or 8006. This state is only possible when the states of the inputs (S_Button1 and S_Button2) change from divergent to convergent (both TRUE) simultaneously when the timer elapses (500 ms) at the same cycle. Ready = TRUE Error = TRUE S_TwoHandOut = FALSE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE Error = FALSE S_TwoHandOut = FALSE
8000	Buttons Actuated	Both buttons actuated correctly. The safety related output is enabled. Ready = TRUE Error = FALSE S_TwoHandOut = TRUE
8001	Init	Function block is active, but in the Init state. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8004	Buttons Released	No Button is actuated. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8005	Button 1 Actuated	Only Button 1 is actuated. Start monitoring timer. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8006	Button 2 Actuated	Only Button 2 is actuated. Start monitoring timer. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8007	Button 2 Released	The safety related output was enabled and is disabled again. FALSE at both S_Button1 and S_Button2 was not achieved after disabling the safety related output. In this state, S_Button1 is TRUE and S_Button2 is FALSE after disabling the safety related output. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8008	Button 1 Released	The safety related output was enabled and is disabled again. FALSE at both S_Button1 and S_Button2 was not achieved after disabling the safety related output. In this state, S_Button1 is FALSE and S_Button2 is TRUE after disabling the safety related output. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
8009	Locked Off	The safety related output was enabled and is disabled again. FALSE at both S_Button1 and S_Button2 was not achieved after disabling the safety related output. In this state, S_Button1 is TRUE and S_Button2 is TRUE after disabling the safety related output. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE

8019	Locked On	Incorrect actuation of the buttons. Waiting for release of both buttons. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE

6.12. Safety Guard Interlocking with Locking

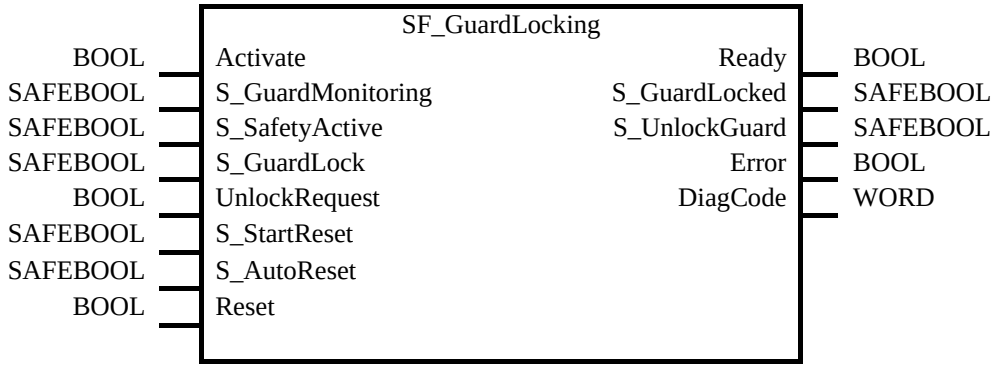
6.12.1. Applicable Safety Standards

Standards	**Requirements**
EN 953: 1997	3.3.3 Control Guard – The hazardous machine functions "covered" by the guard cannot operate until the guard is closed; – Closing the guard initiates operation of the hazardous machine function(s).
EN 1088: 1995	3.3 Definition: Interlocking Guard With Guard Locking – The hazardous machine functions "covered" by the guard cannot operate until the guard is closed and locked; – The guard remains closed and locked until the risk of injury from the hazardous machine functions has passed; – When the guard is closed and locked, the hazardous machine functions "covered" by the guard can operate, but the closure and locking of the guard do not by themselves initiate their operation. 4.2.2 – Interlocking Device With Guard Locking Conditional unlocking ("four-state interlocking"), see Fig. 3 b2)
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.12.2. Interface Description Including FB Name and Short Description

FB Name	**SF_GuardLocking**		
This FB controls an entrance to a hazardous area via an interlocking guard with guard locking (“four state interlocking”)			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_GuardMonitoring	SAFEBOOL	FALSE	Variable. Monitors the guard interlocking. FALSE: Guard open. TRUE: Guard closed.
S_SafetyActive	SAFEBOOL	FALSE	Variable. Status of the hazardous area (EDM), e.g., based on speed monitoring or safe time off delay. FALSE: Machine in "non-safe" state. TRUE: Machine in safe state.
S_GuardLock	SAFEBOOL	FALSE	Variable. Status of the mechanical guard locking. FALSE: Guard is not locked. TRUE: Guard is locked.
UnlockRequest	BOOL	FALSE	Variable. Operator intervention – request to unlock the guard. FALSE: No request. TRUE: Request made.
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
S_AutoReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters. Also used to request the guard to be locked again. The quality of the signal must conform to a manual reset device (EN954-1 Ch. 5.4)
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters

S_GuardLocked	SAFEBOOL	FALSE	Interface to hazardous area which must be stopped. FALSE: No safe state. TRUE: Safe state.
S_UnlockGuard	SAFEBOOL	FALSE	Signal to unlock the guard. FALSE: Close guard. TRUE: Unlock guard.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: --			



6.12.3. Functional Description Including Safe State Description

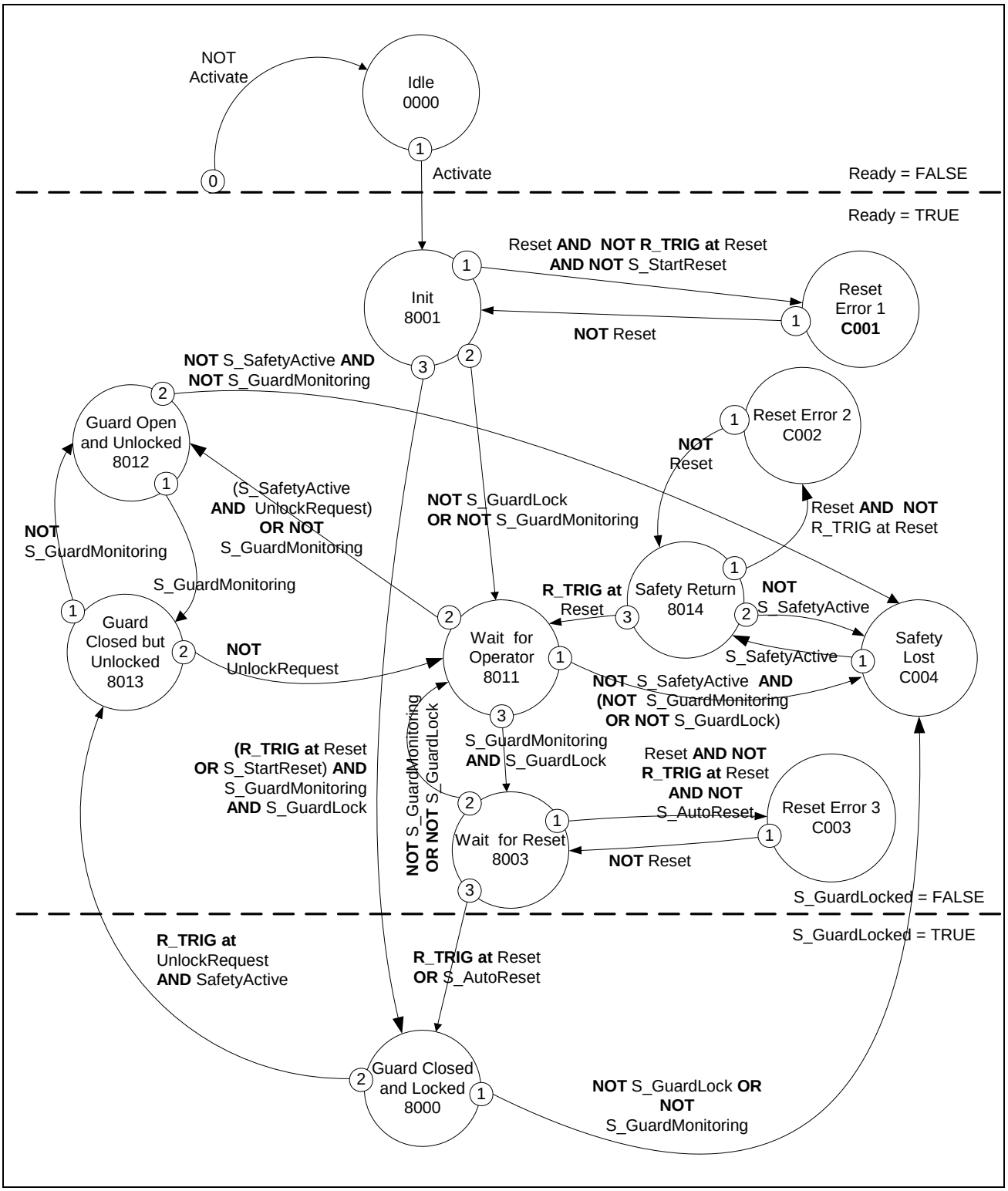
The function controls the guard lock and monitors the position of the guard and the lock. This function block can be used with a mechanical locked switch.
The operator requests to get access to the hazardous area. The guard can only be unlocked when the hazardous area is in a safe state. The guard can be locked if the guard is closed. The machine can be started when the guard is closed and the guard is locked. An open guard or unlocked guard will be detected in the event of a safety-critical situation.

The S_StartReset and S_AutoReset inputs shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

Operation Sequence

1.	External	Request to get the hazardous area to a safe state - not part of this FB
2.	In	Feedback from applicable hazardous area that it is in a safe state (via S_SafetyActive)
3.	In	Operator request to unlock the guard (via UnlockRequest)
4.	Out	Enable guard to be opened (via S_UnlockGuard)
5.	In	Guard unlocked (via S_GuardLock). Guard can be opened now. (S_GuardLocked = FALSE)
		Operator opens the guard
6.	In	Monitoring of status guard via S_GuardMonitoring – signals when guard is closed again
7.	In	Feedback from operator to restart the hazardous area (Reset)
8.	Out	Lock guard guard (S_UnlockGuard)
9.	In	Check if guard is locked (S_GuardLock)
10.	Out	Hazardous area can operate again (S_GuardLocked = TRUE)
11.	Extern	Restart the operation in the hazardous area

State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 38: State diagram for SF_GuardLocking

Typical Timing Diagram

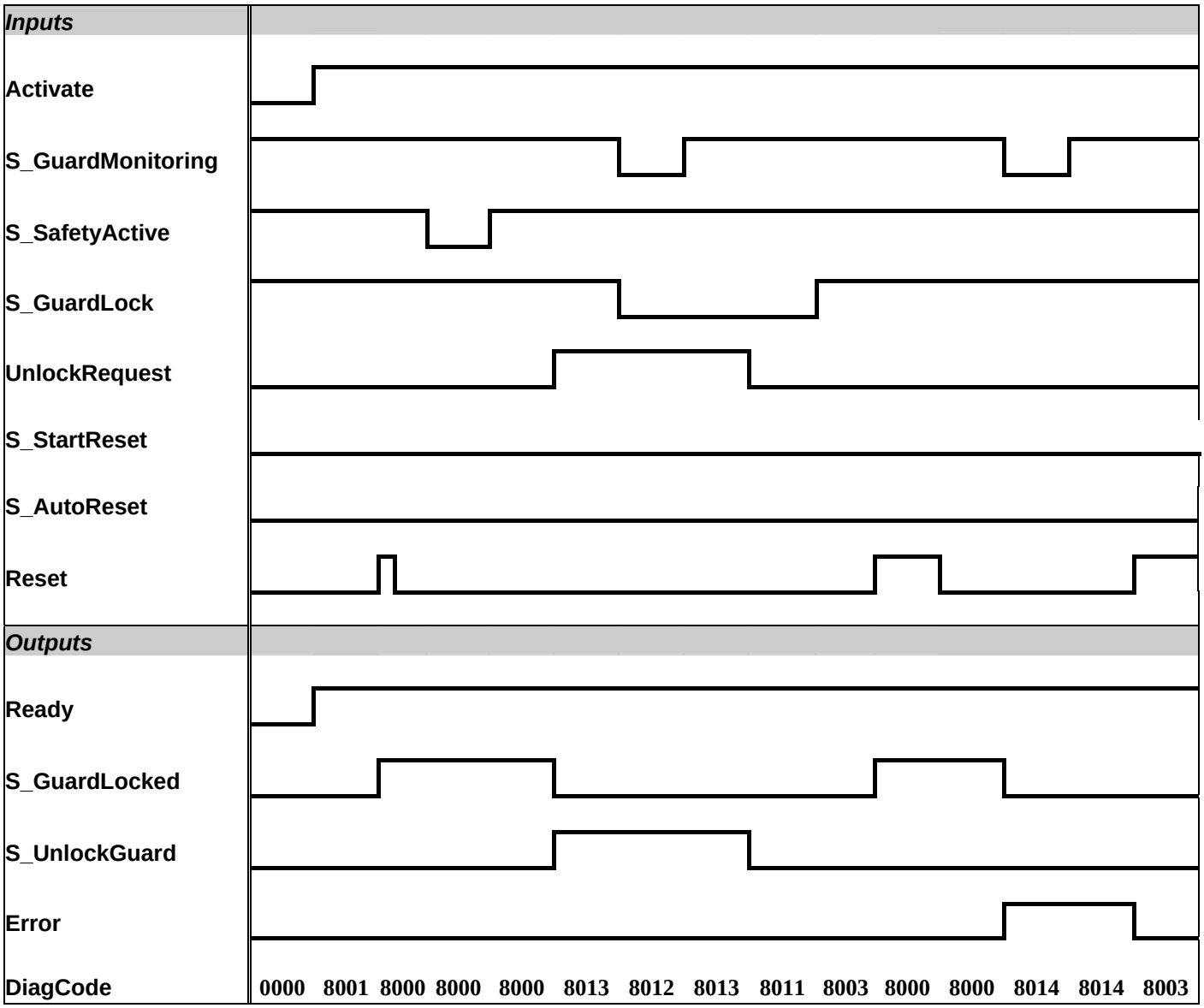


Figure 39: Timing diagram for SF_GuardLocking

6.12.4. Error Detection

Static signals are detected at Reset. Errors are detected at the Guard switches.

6.12.5. Error Behavior

In the event of an error the S_GuardLocked and S_UnlockGuard outputs are set to FALSE, the DiagCode output indicates the relevant error code, and the Error output is set to TRUE.
An error must be acknowledged by a rising trigger at the Reset input.

6.12.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C001	Reset Error1	Static Reset detected in state 8001. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = TRUE
C002	Reset Error 2	Static Reset detected in state C004. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = TRUE
C003	Reset Error 3	Static Reset detected in state 8011. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = TRUE
C004	Safety Lost	Safety lost, guard opened or guard unlocked. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE
8000	Guard Closed and Locked	Guard is locked. Ready = TRUE S_GuardLocked = TRUE S_UnlockGuard = FALSE Error = FALSE
8001	Init	Function block was activated and initiated. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE
8003	Wait for Reset	Door is closed and locked, now waiting for operator reset Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE
8011	Wait for Operator	Waiting for operator to either unlock request or reset. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE
8012	Guard Open and Unlocked	Lock is released and guard is open. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = TRUE Error = FALSE

8013	Guard Closed but Unlocked	Lock is released but guard is closed. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = TRUE Error = FALSE
8014	Safety Return	Return of S_SafetyActive signal, now waiting for operator acknowledge. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE

6.13. Testable Safety Sensors

6.13.1. Applicable Safety Standards

Standards	**Requirements**
IEC 61496-1: 2004	4.2.2.3 Particular requirements for a type 2 ESPE A type 2 ESPE shall have an means of periodic test to reveal a failure to danger (for example loss of detection capability, response time exceeding that specified). A single fault resulting in the loss of detection capability or the increase in response time beyond the specified time or preventing one or more of the OSSDs going to the OFF-state, shall result in a lock-out condition as a result of the next periodic test. Where the periodic test is intended to be initiated by an external (for example machine) safety-related control system, the ESPE shall be provided with suitable input facilities (for example terminals). The duration of the periodic test shall be such that the intended safety function is not impaired. NOTE If the type 2 ESPE is intended for use as a trip device (for example when used as a perimeter guard), and the duration of the periodic test is greater than 150 ms, it is possible for a person to pass through the detection zone without being detected. In this case a restart interlock should be included. If the periodic test is automatically initiated, the correct functioning of the periodic test shall be monitored and a single fault in the parts implementing the monitoring function shall be detected. In the event of a fault, the OSSD(s) shall be signalled to go to the OFF-state. If one or more OSSDs does not go to the OFF-state, a lock-out condition shall be initiated.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.13.2. Interface Description

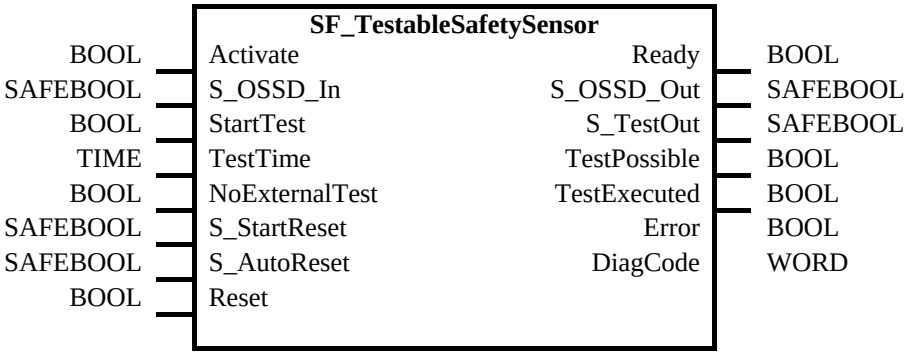
FB Name	**SF_TestableSafetySensor**

This function block detects, for example, the loss of the sensing unit detection capability, the response time exceeding that specified, and static ON signal in single-channel sensor systems. It can be used for external testable safety sensors (ESPE: Electro-sensitive protective equipment, such as a light beam).

VAR_INPUT

Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_OSSD_In	SAFEBOOL	FALSE	Variable. Status of sensor output, e.g., light curtain. FALSE: Safety sensor in test state or demand for safety-related response. TRUE: Sensor in the state for normal operating conditions.
StartTest	BOOL	FALSE	Variable. Input to start sensor test. Sets "S_TestOut" and starts the internal time monitoring function in the FB. FALSE: No test requested. TRUE: Test requested.
TestTime	TIME	T#10ms	Constant. Range: 0 … 150ms. Test time of safety sensor.
NoExternalTest	BOOL	FALSE	Constant. Indicates if external manual sensor test is supported. FALSE: The external manual sensor test is supported. Only after a complete manual sensor switching sequence, a automatic test is possible again after a faulty automatic sensor test. TRUE: The external manual sensor test is **not** supported. An automatic test is possible again without a manual sensor switching sequence after faulty automatic sensor test.
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
S_AutoReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters

VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_OSSD_Out	SAFEBOOL	FALSE	Safety related output indicating the status of the ESPE. FALSE: The sensor has a safety-related action request or test error. TRUE: The sensor has no safety-related action request AND no test error.
S_TestOut	SAFEBOOL	TRUE	Coupled with the test input of the sensor. Although specified as SAFEBOOL, in practice this signal will often be connected to a BOOL output. FALSE: Test request issued. TRUE: No test request.
TestPossible	BOOL	FALSE	Feedback signal to the process. FALSE: An automatic sensor test is **not** possible. TRUE: An automatic sensor test is possible.
TestExecuted	BOOL	FALSE	A positive signal edge indicates the successful execution of the automatic sensor test. FALSE: - An automatic sensor test was not executed yet. - An automatic sensor test is active. - An automatic sensor test was faulty. TRUE: A sensor test was executed successfully.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: OSSD: Output Signal Switching Device.			



6.13.3. Functional Description

Type 2 ESPE shall have a means of periodic testing to detect a hazardous fault (e.g., loss of sensing unit detection capability, response time exceeding that specified). The test signal shall simulate the actuation of the sensing device and the duration of the periodic test shall not exceed 150 ms. The test shall verify that each light beam operates in the manner specified by the supplier. If the periodic test is intended to be initiated by an external safety-related control system (e.g., a machine), the ESPE shall be provided with suitable input facilities (e.g., terminals).The ESPE must be selected in respect of the product standards EN IEC 61496-1, -2 and -3 and the required categories according EN 954-1. It must be monitored by separate functionality, that the test is initiated within appropriate intervals. The S_StartReset and S_AutoReset inputs shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

Test mode:

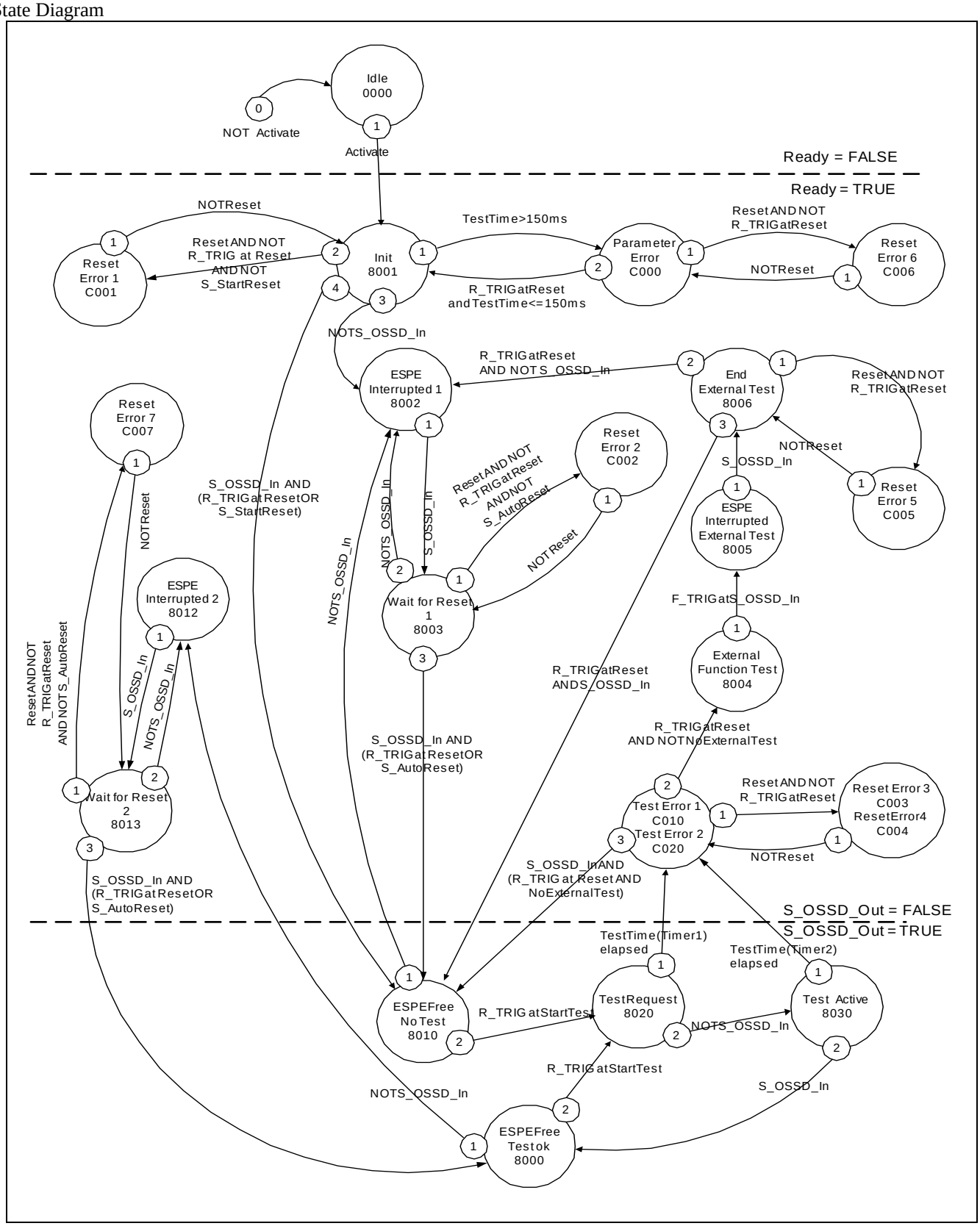
1. StartTest = TRUE: S_TestOut = FALSE. Start monitoring time
2. S_TestOut signal stops transmitter (Monitoring of TestTime started first time)
3. S_OSSD_In changes from TRUE to FALSE (Monitoring of TestTime started second time)
4. S_TestOut changes from FALSE to TRUE
5. Start transmitter
6. Sensor S_OSSD_In changes from FALSE to TRUE
7. Stop monitoring time
8. S_OSSD_Out is set to TRUE during testing

Optional startup inhibits:

- Startup inhibit after function block activation.

- Startup inhibit after interruption of the protective device.

State Diagram



Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 40: State diagram for SF_TestableSafetySensor

Typical Timing Diagram

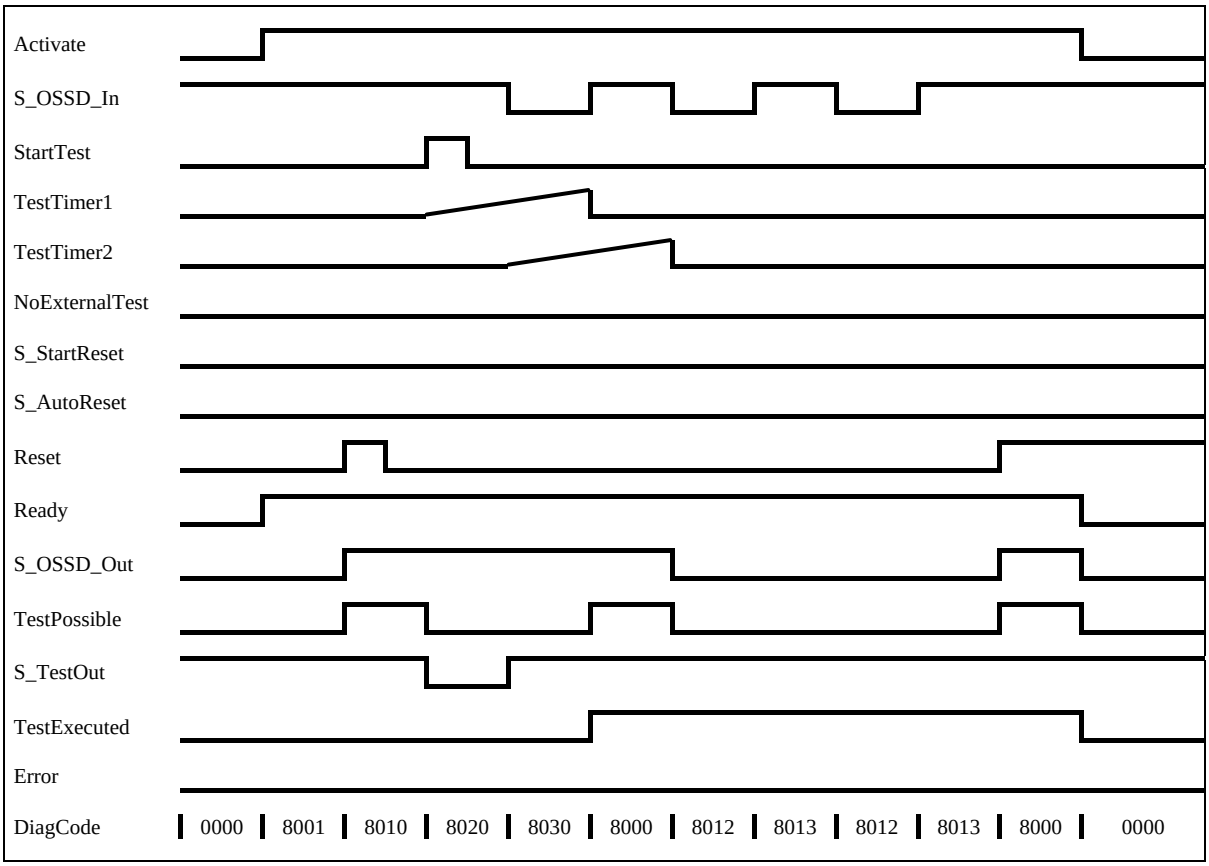


Figure 41: Timing diagram for SF_TestableSafetySensor

6.13.4. Error Detection

The following conditions force a transition to the Error state:

- Test time overrun without delayed sensor feedback.
- Test without sensor signal feedback.
- Invalid static reset signal in the process.
- Plausibility check of the monitoring time setting.

6.13.5. Error Behavior

In the event of an error, the S_OSSD_Out output is set to FALSE and remains in this safe state.
Once the error has been removed and the sensor is on (S_OSSD_In = TRUE) – a reset removes the error state and sets the S_OSSD_Out output to TRUE.
If S_AutoReset = FALSE, a rising trigger is required at Reset.
After transition of S_OSSD_In to TRUE, the optional startup inhibit can be reset by a rising edge at the Reset input.
After block activation, the optional startup inhibit can be reset by a rising edge at the Reset input.

6.13.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting
FB-specific error codes:		
C000	Parameter Error	Invalid value at the TestTime parameter. Values between 0 ms and 150 ms are possible. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
C001	Reset Error 1	Static Reset condition detected after FB activation. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
C002	Reset Error 2	Static Reset condition detected in state 8003. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
C003	Reset Error 3	Static Reset condition detected in state C010. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
C004	Reset Error 4	Static Reset condition detected in state C020. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
C005	Reset Error 5	Static Reset condition detected in state 8006. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
C006	Reset Error 6	Static Reset condition detected in state C000. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE

C007	Reset Error 7	Static Reset condition detected in state 8013. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = TRUE Error = TRUE
C010	Test Error 1	Test time elapsed in state 8020. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
C020	Test Error 2	Test time elapsed in state 8030. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
8001	Init	An activation has been detected by the FB. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
8002	ESPE Interrupted 1	The FB has detected a safety demand. The switch has not been automatically tested yet. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
8003	Wait for Reset 1	Wait for rising trigger of Reset after state 8002. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE

8004	External Function Test	The automatic sensor test was faulty. An external manual sensor test is necessary. The support for the necessary external manual sensor test has been activated at the FB (NoExternalTest = FALSE). A negative signal edge at the sensor is required. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
8005	ESPE Interrupted External Test	The automatic sensor test was faulty. An external manual sensor test is necessary. The support for the necessary external manual sensor test has been activated at the FB (NoExternalTest = FALSE). A TRUE signal at the sensor is required. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
8006	End External Test	The automatic sensor test was faulty. An external manual sensor test is necessary. The support for the necessary external manual sensor test has been activated at the FB (NoExternalTest = FALSE). The external manual test is complete. The FB detected a complete sensor switching cycle (external controlled). Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
8010	ESPE Free No Test	The FB has not detected a safety demand. The sensor has not been tested automatically. Ready = TRUE S_OSSD_Out = TRUE S_TestOut = TRUE TestPossible = TRUE TestExecuted = FALSE Error = FALSE
8020	Test Request	The automatic sensor test is active. Test Timer is started first time. The transmitter signal of the sensor is switched off by the FB. The signal of the receiver must follow the signal of the transmitter. Ready = TRUE S_OSSD_Out = TRUE S_TestOut = FALSE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
8030	Test Active	The automatic sensor test is active. Test Timer is started second time. The transmitter signal of the sensor is switched on by the FB. The signal of the receiver must follow the signal of the transmitter. Ready = TRUE S_OSSD_Out = TRUE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE

8000	ESPE Free Test ok	The FB has not detected a safety demand. The sensor was automatically tested. Ready = TRUE S_OSSD_Out = TRUE S_TestOut = TRUE TestPossible = TRUE TestExecuted = TRUE Error = FALSE
8012	ESPE Interrupted 2	The FB has detected a safety demand. The switch was automatically tested. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = TRUE Error = FALSE
8013	Wait for Reset 2	Wait for rising trigger of Reset after state 8012. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = TRUE Error = FALSE

6.14. Sequential Muting

6.14.1. Applicable Safety Standards

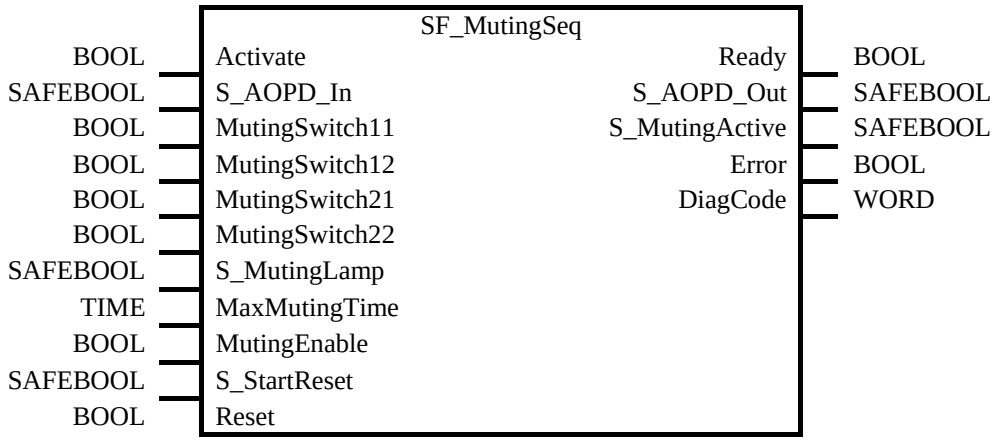
Standards	Requirements
IEC 61496-1:2004	A.7 Muting, A.7.1.2 There shall be at least two independent hard-wired muting signal sources to initiate the function. It shall not be possible to initiate muting when the OSSDs are already in the OFF-state. A.7.1.3 The mute function shall only be inititated by the correct sequence and/or timing of the mute signals. Should conflicting muting signals occur, the ESPE shall not allow a muted condition to occur. A.7.1.4 There shall be at least two independent hard-wired muting signal sources to stop the function. The muting function shall stop when the first of these muting signals changes state. The deactivation of the muting function shall not rely only on the clearance of the ESPE. A.7.1.5 The muting signals should be continuously present during muting. When the signals are not continuously present, an incorrect sequence and/or the expiration of a pre-set time limit shall cause either a lock-out condition or a restart interlock. A.7.4 Indication: A mute status signal or indicator shall be provided (in some applications, an indication signal of muting is necessary)
CD IEC 62046/Ed. 1: 2005	5.5.1: .. an indicator to show when the muting function is active can be necessary. The muting function shall be initiated and terminated automatically….Incorrect signals, sequence, or timing of the muting sensors or signals shall not allow a mute condition. It shall not be possible to initiate the muting function when: – the protective equipment OSSDs are in the OFF-state; – the protective equipment is in the lock-out condition. - initiation of the muting function by two or more independent muting sensors such that a single fault cannot cause a muted condition; - termination of the muting function by two or more independent muting sensors such that deactivation of one sensor will terminate the muting function; - use of timing and sequence control of the muting sensors to ensure correct muting operation; 5.5.3: The following measures shall be considered:… - limiting muting to a fixed time that is only sufficient for the material to pass through the detection zone. When this time is exceeded, the muting function should be cancelled and all hazardous movements stopped; Annex F.3 Four beams - sequence control: (see also Fig. F.3.1 and table F.1) The initiation of the muting function depends on monitoring the correct sequence of activation of the muting sensors. For example, in the muted condition, if S2 *[in this document MS_12]* is deactivated before S3 *[in this document MS_21]* is activated, muting is terminated. Annex F.5: Methods to avoid manipulation of the muting function: … use a muting enable command generated by the control system of the machine that will only enable the muting function when needed by the machine cycle.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.14.2. Interface Description

FB Name	**SF_MutingSeq**		
Muting is the intended suppression of the safety function (e.g., light barriers). In this FB, sequential muting with four muting sensors is specified.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_AOPD_In	SAFEBOOL	FALSE	Variable. OSSD signal from AOPD. FALSE: Protection field interrupted. TRUE: Protection field not interrupted.
MutingSwitch11	BOOL	FALSE	Variable. Status of Muting sensor 11. FALSE: Muting sensor 11 not actuated. TRUE: Workpiece actuates muting sensor 11. It shall be noted in the FB manual that a SAFEBOOL must be connected instead of a BOOL depending on the safety requirements.
MutingSwitch12	BOOL	FALSE	Variable. Status of Muting sensor 12. FALSE: Muting sensor 12 not actuated. TRUE: Workpiece actuates muting sensor 12. It shall be noted in the FB manual that a SAFEBOOL must be connected instead of a BOOL depending on the safety requirements.
MutingSwitch21	BOOL	FALSE	Variable. Status of Muting sensor 21. FALSE: Muting sensor 21 not actuated. TRUE: Workpiece actuates muting sensor 21. It shall be noted in the FB manual that a SAFEBOOL must be connected instead of a BOOL depending on the safety requirements.
MutingSwitch22	BOOL	FALSE	Variable. Status of Muting sensor 22. FALSE: Muting sensor 22 not actuated. TRUE: Workpiece actuates muting sensor 22. It shall be noted in the FB manual that a SAFEBOOL must be connected instead of a BOOL depending on the safety requirements.
S_MutingLamp	SAFEBOOL	FALSE	Variable or constant. Indicates operation of the muting lamp. FALSE: Muting lamp failure. TRUE: Muting lamp no failure
MaxMutingTime	TIME	T#0s	Constant 0 .. 10 min; Maximum time for complete muting sequence, timer started when first muting sensor is actuated.
MutingEnable	BOOL	FALSE	Variable or constant. Command by the control system that enables the start of the muting function when needed by the machine cycle. After the start of the muting function, this signal can be switched off. FALSE: Muting not enabled TRUE: Start of Muting function enabled
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			

Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_AOPD_Out	SAFEBOOL	FALSE	Safety related output, indicates status of the muted guard. FALSE: AOPD protection field interrupted and muting not active. TRUE: AOPD protection field not interrupted or muting active.
S_MutingActive	SAFEBOOL	FALSE	Indicates status of Muting process. FALSE: Muting not active. TRUE: Muting active.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters

Notes: A short circuit in the muting sensor signals, or a functional application error to supply these signals, are not detected by this FB but interpreted as incorrect muting sequence (The types are BOOL, provided by the functional application hardware and / or software). However, this condition should not lead to unwanted muting. The user should take care to include this in his risk analysis.



6.14.3. Functional Description

Muting is the intended suppression of the safety function. This is required, e.g., when transporting the material into the danger zone without causing the machine to stop. Muting is triggered by muting sensors. The use of two or four muting sensors and correct integration into the production sequence must ensure that no persons enter the danger zone while the light curtain is muted. Muting sensors can be proximity switches, photoelectric barriers, limit switches, etc. which do not have to be failsafe. Active muting mode must be indicated by indicator lights.

There are sequential and parallel muting procedures. In this FB, sequential muting with four muting sensors was used; an explanation for the forward direction of transportation is provided below. The FB can be used in both directions, forward and backward. The muting should be enabled with the MutingEnable signal by the process control to avoid manipulation. When the MutingEnable signal is not available, this input must be set to TRUE.

The FB input parameters include the signals of the four muting sensors (MutingSwitch11 ... MutingSwitch22) as well as the OSSD signal from the "active opto-electronic protective device", S_AOPD_In.

The S_StartReset input shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

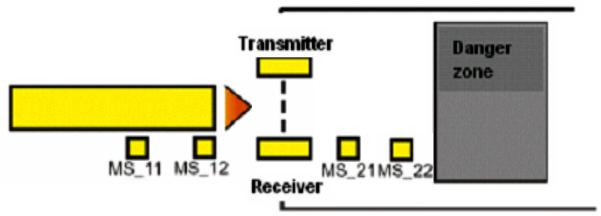
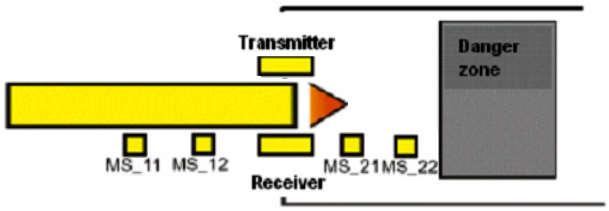
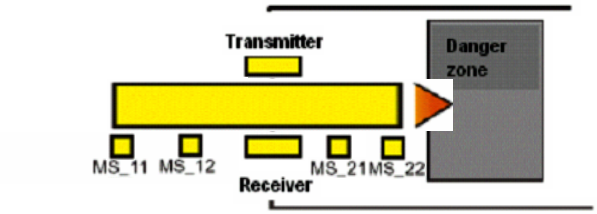
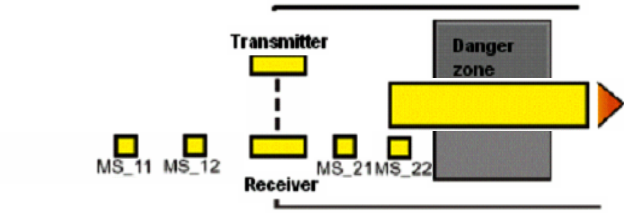
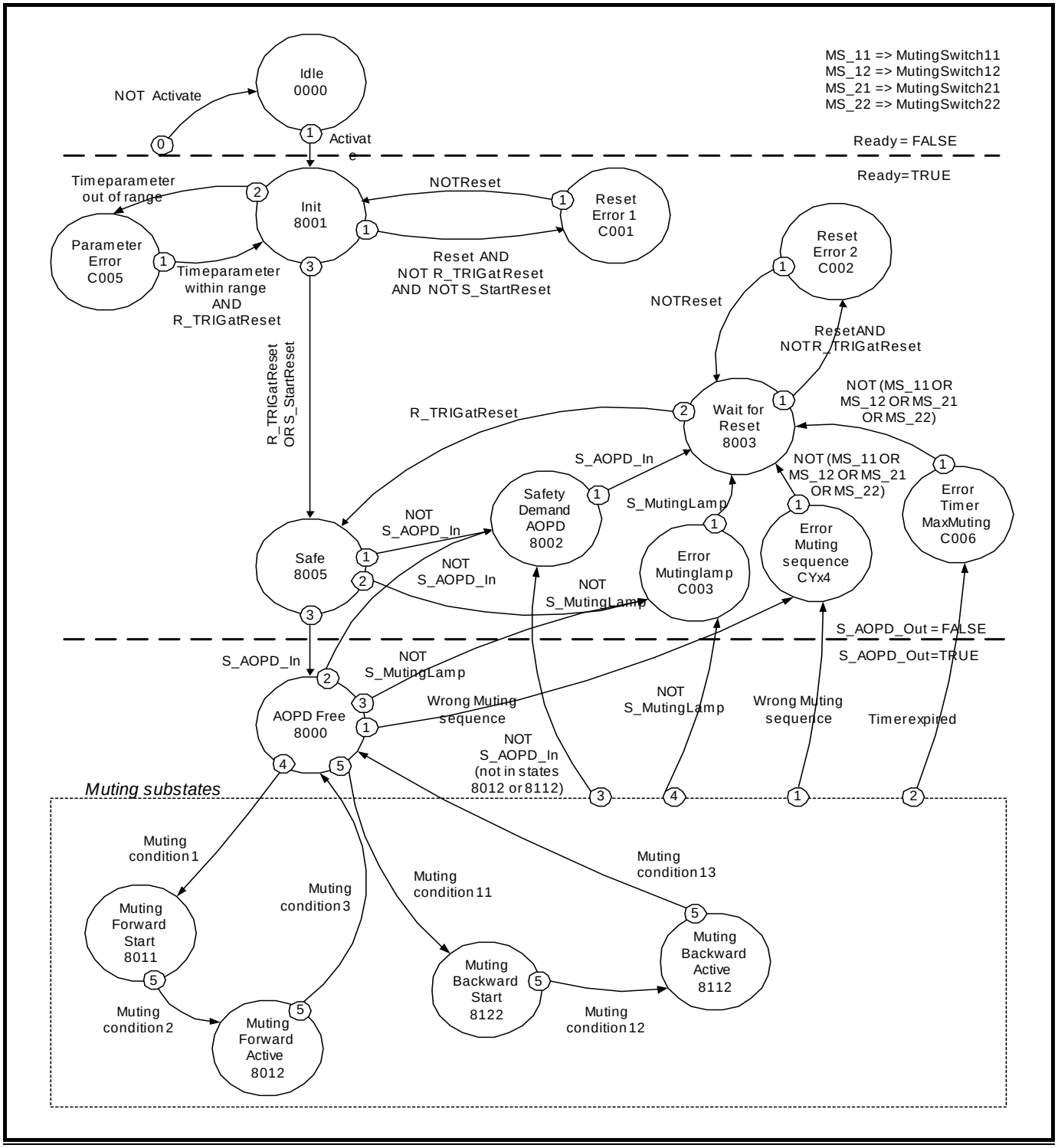
No.	Figure	Explanation
1		If muting sensor MutingSwitch12 (MS_12) is activated by the product after MutingSwitch11 (MS_11), the muting mode is activated.
2		Muting mode remains active as long as MutingSwitch11 (MS_11) and MutingSwitch12 (MS_12) are activated by the product. The product may pass through the light curtain without causing a machine stop.
3		Before muting sensors MutingSwitch11 (MS_11) and MutingSwitch12 (MS_12) are disabled, muting sensors MutingSwitch21 (MS_21) and MutingSwitch22 (MS_22) must be activated. This ensures that muting mode remains active.
4		Muting mode is terminated if only muting sensor MutingSwitch22 (MS_22) is activated by the product.

Figure 42: Example for SF_MutingSeq in forward direction with four sensors

State Diagram



Note 1: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Note 2: Within muting substates, transitions due to Error Muting sequence (priority 1), Error Timer (priority 2), Safety demand AOPD (priority 3) or Error Muting lamp (priority 4) have higher priority than transitions to Muting substates (priority 5).

Figure 43: State diagram for SF_MutingSeq

Muting Conditions

Forward Direction
Muting condition 1 (to 8011) (MS_11 is the first entry switch actuated). Start timer MaxMutingTime**:**
MutingEnable AND (**R_TRIG at** MS_11 **AND NOT** MS_12 **AND NOT** MS_21 **AND NOT** MS_22)

Muting condition 2 (from 8011 to 8012) (MS_12 is the second entry switch actuated)**:**
MutingEnable AND (MS_11 **AND R_TRIG at** MS_12 **AND NOT** MS_21 **AND NOT** MS_22)

Muting condition 3 (from 8012 to 8000) (MS_21 is the first exit switch released). Stop timer MaxMutingTime**:**
NOT MS_11 **AND NOT** MS_12 **AND F_TRIG at** MS_21 **AND** MS_22

Backward Direction
Muting condition 11 (to 8122) (MS_22 is the first entry switch actuated). Start timer MaxMutingTime**:**
MutingEnable AND (**NOT** MS_11 **AND NOT** MS_12 **AND NOT** MS_21 **AND R_TRIG at** MS_22)

Muting condition 12 (from 8122 to 8112) (MS_21 is the second entry switch actuated):
MutingEnable AND (**NOT** MS_11 **AND NOT** MS_12 **AND R_TRIG at** MS_21 **AND** MS_22)

Muting condition 13 (MS_12 is the first exit switch released). Stop timer MaxMutingTime**:**
MS_11 **AND F_TRIG at** MS_12 **AND NOT** MS_21 **AND NOT** MS_22

Specification of wrong Muting Sequences:

In state 8000: (**NOT** MutingEnable **AND R_TRIG** at MS_11) **OR** (**NOT** MutingEnable **AND R_TRIG** at MS_22) **OR** (MS_12 **OR** MS_21) **OR** (MS_11 **AND** MS_22)
In state 8011: **NOT** MutingEnable **OR NOT** MS_11 **OR** MS_21 **OR** MS_22
In state 8012: **R_TRIG at** MS_11 **OR R_TRIG at** MS_12 **OR F_TRIG at** MS_22
In state 8122: **NOT** MutingEnable **OR** MS_11 **OR** MS_12 **OR NOT** MS_22
In state 8112: **F_TRIG at** MS_11 **OR R_TRIG at** MS_21 **OR R_TRIG at** MS_22

Typical Timing Diagram

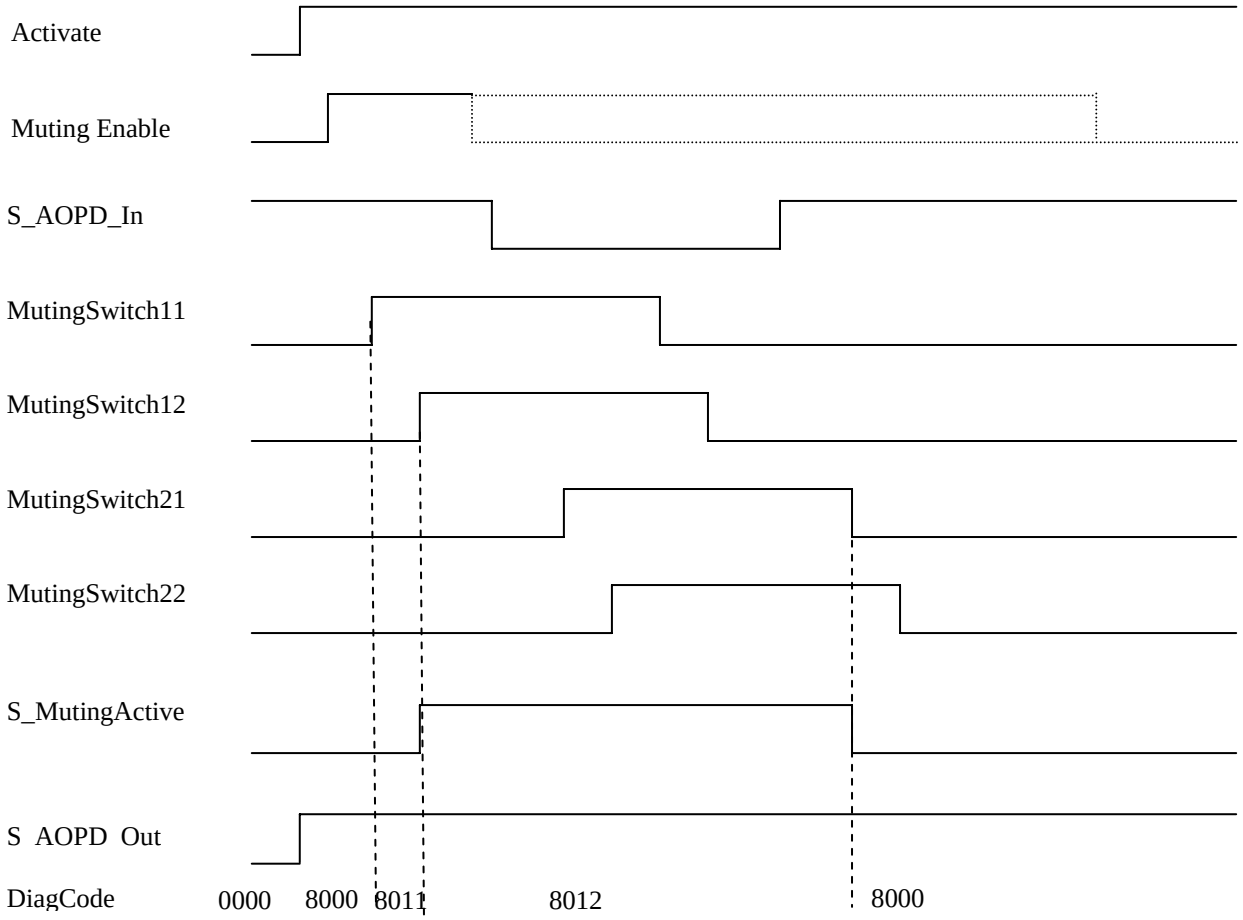


Figure 44: Timing diagram for SF_MutingSeq with S_StartReset = TRUE

6.14.4. Error Detection

The FB detects the following error conditions:

- Muting sensors MutingSwitch11, MutingSwitch12, MutingSwitch21, and MutingSwitch22 are activated in the wrong order.
- Muting sequence starts without being enabled by MutingEnable
- A faulty muting lamp is indicated by S_MutingLamp = FALSE.
- A static Reset condition.
- MaxMutingTime has been set to a value less than T#0s or greater than T#10min.
- The muting function (S_MutingActive = TRUE) exceeds the maximum muting time MaxMutingTime.

6.14.5. Error Behavior

In the event of an error, the S_AOPD_Out and S_MutingActive outputs are set to FALSE. The DiagCode output indicates the relevant error code and the Error output is set to TRUE.

A restart is inhibited until the error conditions are cleared and the Safe state is acknowledged with Reset by the operator.

6.14.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C001	Reset Error 1	Static Reset condition detected after FB activation. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C002	Reset Error 2	Static Reset condition detected in state 8003. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C003	Error Muting lamp	Error detected in muting lamp. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
CYx4	Error Muting sequence	Error detected in muting sequence in states 8000, 8011, 8012, 8112 or 8122. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE Y = Status in the sequence (2 states for forward and 2 states for backward direction). C0x4 = Error occurred in state 8000 C1x4 = Error occurred in state Forward 8011 C2x4 = Error occurred in state Forward 8012 C3x4 = Error occurred in state Backward 8122 C4x4 = Error occurred in state Backward 8112 CFx4 = Muting Enable missing x = Status of the sensors when error occurred (4 bits: LSB = MS_11; MS_12; MS_21; MSB = MS_22).
C005	Parameter Error	MaxMutingTime value out of range. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C006	Error Timer MaxMuting	Timing error: Active muting time (when S_MutingActive = TRUE) exceeds MaxMutingTime. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE

8000	AOPD Free	Muting not active and no safety demand from AOPD. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8001	Init	Function block has been activated. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8002	Safety Demand AOPD	Safety demand detected by AOPD, muting not active. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8003	Wait for Reset	Safety demand or errors have been detected and are now cleared. Operator acknowledgment by Reset required. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8005	Safe	Safety function activated. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8011	Muting Forward Start	Muting forward, sequence is in starting phase and no safety demand. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8012	Muting Forward Active	Muting forward, sequence is active. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
8112	Muting Backward Active	Muting backward, sequence is active. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
8122	Muting Backward Start	Muting backward, sequence is in starting phase and no safety demand. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE

6.15. Parallel Muting

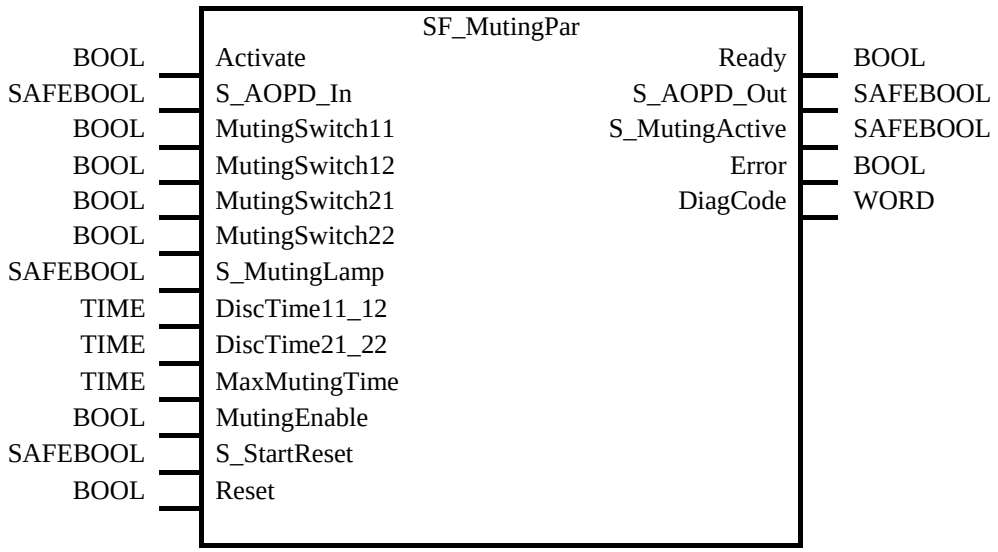
6.15.1. Applicable Safety Standards

Standards	Requirements
IEC 61496-1:2004	A.7 Muting, A.7.1.2 There shall be at least two independent hard-wired muting signal sources to initiate the function. It shall not be possible to initiate muting when the OSSDs are already in the OFF-state. A.7.1.3 The mute function shall only be inititated by the correct sequence and/or timing of the mute signals. Should conflicting muting signals occur, the ESPE shall not allow a muted condition to occur. A.7.1.4 There shall be at least two independent hard-wired muting signal sources to stop the function. The muting function shall stop when the first of these muting signals changes state. The deactivation of the muting function shall not rely only on the clearance of the ESPE. A.7.1.5 The muting signals should be continuously present during muting. When the signals are not continuously present, an incorrect sequence and/or the expiration of a pre-set time limit shall cause either a lock-out condition or a restart interlock. A.7.4 Indication: A mute status signal or indicator shall be provided (in some applications, an indication signalof muting is necessary
CD IEC 62046/Ed. 1: 2005	5.5.1: .. an indicator to show when the muting function is active can be necessary. The muting function shall be initiated and terminated automatically….Incorrect signals, sequence, or timing of the muting sensors or signals shall not allow a mute condition. It shall not be possible to initiate the muting function when: – the protective equipment OSSDs are in the OFF-state; – the protective equipment is in the lock-out condition. - initiation of the muting function by two or more independent muting sensors such that a single fault cannot cause a muted condition; - termination of the muting function by two or more independent muting sensors such that deactivation of one sensor will terminate the muting function; - use of timing and sequence control of the muting sensors to ensure correct muting operation; 5.5.3: The following measures shall be considered:… - limiting muting to a fixed time that is only sufficient for the material to pass through the detection zone. When this time is exceeded, the muting function should be cancelled and all hazardous movements stopped; Annex F.2 Four beams – timing control: (see also Fig. F.2.4): The monitoring of the muting function is based on time limitation between the actuation of the sensors S1 *[in this document MS_11]* and S2 *[in this document MS_12]* and between the actuation of sensors S3 *[in this document MS_21]* and S4 *[in this document MS_22]*. A maximum time limit of 4 sec. is recommended. The muting function is initiated by the two sensors S1, S2 and maintained by the two sensors S3, S4; this means that for a certain time all the four sensors are activated. The muting function is terminated when S3 or S4 is deactivated. Annex F.5: Methods to avoid manipulation of the muting function: … use a muting enable command generated by the control system of the machine that will only enable the muting function when needed by the machine cycle.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.15.2. Interface Description

FB Name	**SF_MutingPar**		
Muting is the intended suppression of the safety function. In this FB, parallel muting with four muting sensors is specified.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_AOPD_In	SAFEBOOL	FALSE	Variable. OSSD signal from AOPD. FALSE: Protection field interrupted. TRUE: Protection field not interrupted.
MutingSwitch11	BOOL	FALSE	Variable. Status of Muting sensor 11. FALSE: Muting sensor 11 not actuated. TRUE: Workpiece actuates muting sensor 11. It shall be noted in the FB manual that a SAFEBOOL must be connected instead of a BOOL depending on the safety requirements.
MutingSwitch12	BOOL	FALSE	Variable. Status of Muting sensor 12. FALSE: Muting sensor 12 not actuated. TRUE: Workpiece actuates muting sensor 12. It shall be noted in the FB manual that a SAFEBOOL must be connected instead of a BOOL depending on the safety requirements.
MutingSwitch21	BOOL	FALSE	Variable. Status of Muting sensor 21. FALSE: Muting sensor 21 not actuated. TRUE: Workpiece actuates muting sensor 21. It shall be noted in the FB manual that a SAFEBOOL must be connected instead of a BOOL depending on the safety requirements.
MutingSwitch22	BOOL	FALSE	Variable. Status of Muting sensor 22. FALSE: Muting sensor 22 not actuated. TRUE: Workpiece actuates muting sensor 22. It shall be noted in the FB manual that a SAFEBOOL must be connected instead of a BOOL depending on the safety requirements.
S_MutingLamp	SAFEBOOL	FALSE	Variable or constant. Indicates operation of the muting lamp. FALSE: Muting lamp failure. TRUE: Muting lamp no failure.
DiscTime11_12	TIME	T#0s	Constant 0..4 s; Maximum discrepancy time for MutingSwitch11 and MutingSwitch12.
DiscTime21_22	TIME	T#0s	Constant 0..4 s; Maximum discrepancy time for MutingSwitch21 and MutingSwitch22.
MaxMutingTime	TIME	T#0s	Constant 0..10 min; Maximum time for complete muting sequence, timer started when first muting sensor is actuated.

MutingEnable	BOOL	FALSE	Variable or constant. Command by the control system that enables the start of the muting function when needed by the machine cycle. After the start of the muting function, this signal can be switched off. FALSE: Muting not enabled TRUE: Start of Muting function enabled
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_AOPD_Out	SAFEBOOL	FALSE	Safety related output, indicates status of the muted guard. FALSE: AOPD protection field interrupted and muting not active. TRUE: AOPD protection field not interrupted or muting active.
S_MutingActive	SAFEBOOL	FALSE	Indicates status of Muting process. FALSE: Muting not active. TRUE: Muting active.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: A short circuit in the muting sensor signals, or a functional application error to supply these signals, are not detected by this FB (The types are BOOL, provided by the functional application hardware and / or software). However, this condition should not lead to unwanted muting. The user should take care to include this in his risk analysis.			



6.15.3. Functional Description

Muting is the intended suppression of the safety function. This is required, e.g., when transporting the material into the danger zone without causing the machine to stop. Muting is triggered by muting sensors. The use of two or four muting sensors and correct integration into the production sequence must ensure that no persons enter the danger zone while the light curtain is muted. Muting sensors can be proximity switches, photoelectric barriers, limit switches, etc. which do not have to be failsafe. Active muting mode must be indicated by indicator lights.

There are sequential and parallel muting procedures. In this FB, parallel muting with four muting sensors was used; an explanation is provided below. The FB can be used in both directions, forward and backward. The muting should be enabled with the MutingEnable signal by the process control to avoid manipulation.

The FB input parameters include the signals of the four muting sensors (MutingSwitch11 ... MutingSwitch22), the OSSD signal from the "active opto-electronic protective device", S_AOPD_In, as well as three parameterizable times (DiscTime11_12, DiscTime21_22, and MaxMutingTime).

The S_StartReset input shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

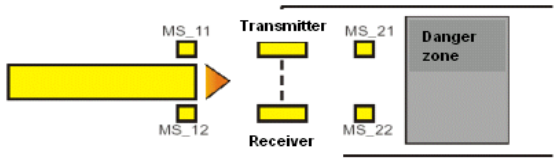
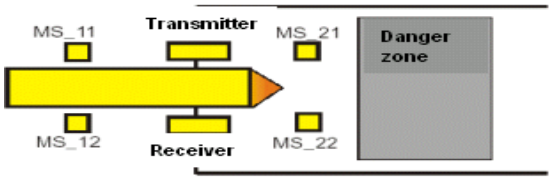
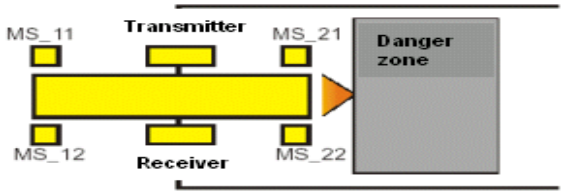
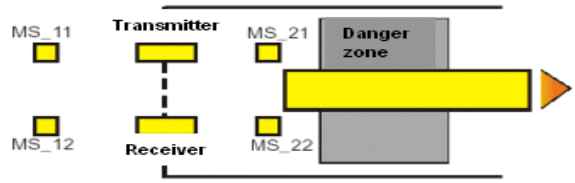
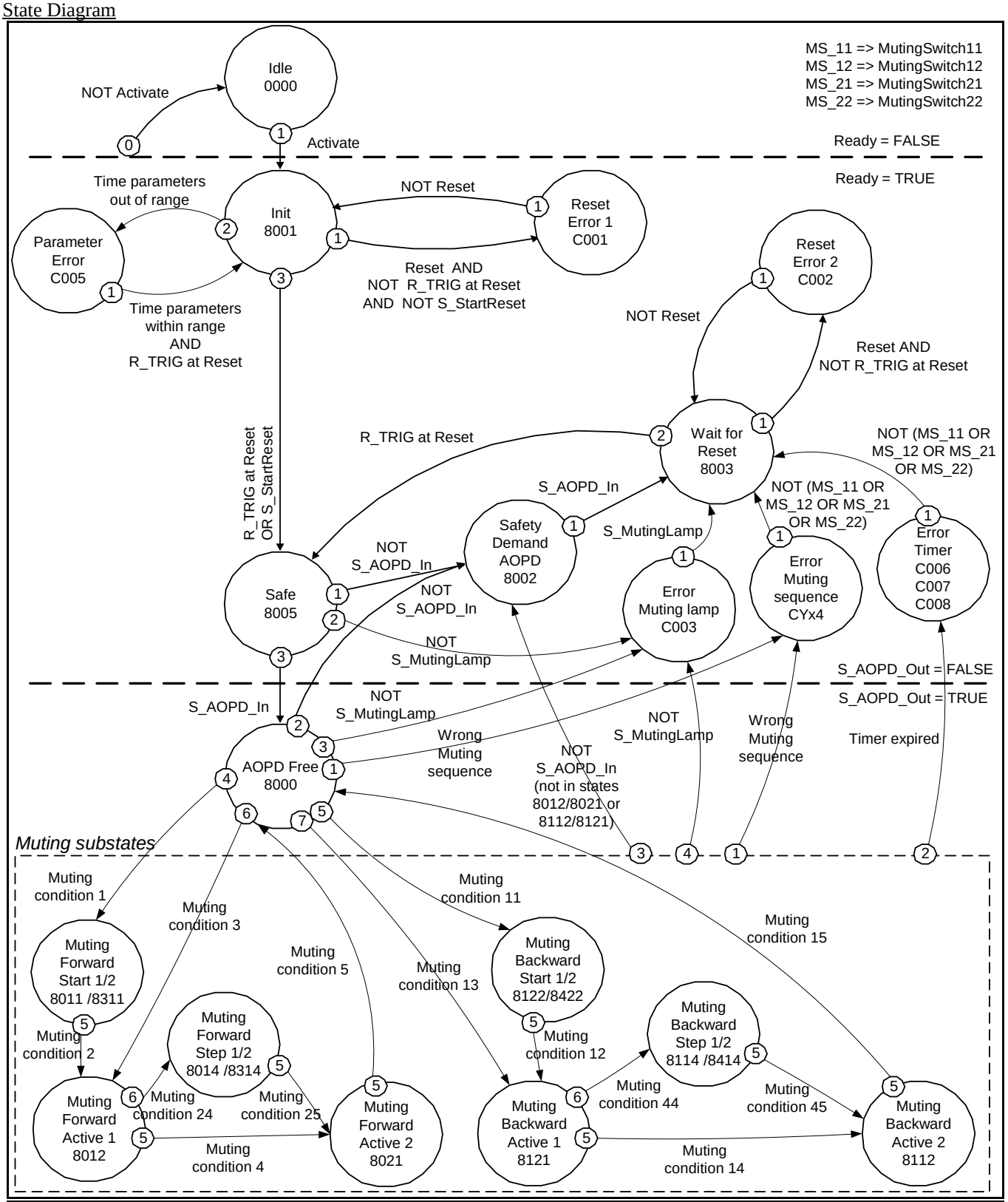
No.	Figure	Explanation
1		If the muting sensors MutingSwitch11 (MS_11) and MutingSwitch12 (MS_12) are activated by the product within the time DiscTime11_12, muting mode is activated (S_MutingActive = TRUE).
2		Muting mode remains active as long as MutingSwitch11 (MS_11) and MutingSwitch12 (MS_12) are activated by the product. The product may pass through the light curtain without causing a machine stop.
3		Before muting sensors MutingSwitch11 (MS_11) and MutingSwitch12 (MS_12) are disabled, muting sensors MutingSwitch21 (MS_21) and MutingSwitch22 (MS_22) must be activated. This ensures that muting mode remains active. The time discrepancy between switching of MutingSwitch21 and MutingSwitch22 is monitored by the time DiscTime21_22.
4		Muting mode is terminated if either muting sensor MutingSwitch21 (MS_21) or MutingSwitch22 (MS_22) is disabled by the product. The maximum time for muting mode to be active is the MaxMutingTime.

Figure 45: Example for SF_MutingPar in forward direction with four sensors

State Diagram



Note1: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Note 2: Within muting substates, transitions due to Error Muting sequence (priority 1), Error Timer (priority 2), Safety demand AOPD (priority 3) or Error Muting lamp (priority 4) have higher priority than transitions to Muting substates (priority 5 or 6).

Note 3: Muting conditions are defined below.

Figure 46: State diagram for SF_MutingPar

Forward Direction
Muting condition 1 (to 8011) (MS_11 is the first entry switch actuated). Start timers MaxMutingTime and DiscTime11_12**:** MutingEnable **AND** (**R_TRIG at** MS_11 **AND NOT** MS_12 **AND NOT** MS_21 **AND NOT** MS_22)
Muting condition 1 (to 8311) (MS_12 is the first entry switch actuated). Start timers MaxMutingTime and DiscTime11_12**:** MutingEnable **AND** (**NOT** MS_11 **AND R_TRIG at** MS_12 **AND NOT** MS_21 **AND NOT** MS_22)

Muting condition 2 (from 8011) (MS_12 is the second entry switch actuated). Stop timer DiscTime11_12**:** MutingEnable **AND** (MS_11 **AND R_TRIG at** MS_12 **AND NOT** MS_21 **AND NOT** MS_22)
Muting condition 2 (from 8311) (MS_11 is the second entry switch actuated). Stop timer DiscTime11_12**:** MutingEnable **AND** (**R_TRIG at** MS_11 **AND** MS_12 **AND NOT** MS_21 **AND NOT** MS_22)

Muting condition 3 (both entry switches actuated in same cycle). Start timer MaxMutingTime**:** MutingEnable **AND** (**R_TRIG at** MS_11 **AND R_TRIG at** MS_12 **AND NOT** MS_21 **AND NOT** MS_22)

Muting condition 4 (all switches actuated)**:** MS_11 **AND** MS_12 **AND** MS_21 **AND** MS_22
Muting condition 24 (to 8014) (MS_21 is the first exit switch actuated). Start timer DiscTime21_22**:** MS_11 **AND** MS_12 **AND R_TRIG at** MS_21 **AND NOT** MS_22
Muting condition 24 (to 8314) (MS_22 is the first exit switch actuated). Start timer DiscTime21_22**:** MS_11 **AND** MS_12 **AND NOT** MS_21 **AND R_TRIG at** MS_22
Muting condition 25 (from 8014) (MS_22 is the second exit switch actuated). Stop timer DiscTime21_22**:** MS_11 **AND** MS_12 **AND** MS_21 **AND R_TRIG at** MS_22
Muting condition 25 (from 8314) (MS_21 is the second exit switch actuated). Stop timer DiscTime21_22**:** MS_11 **AND** MS_12 **AND R_TRIG at** MS_21 **AND** MS_22

Muting condition 5 (one of the exit switches released). Stop timer MaxMutingTime**: NOT** MS_11 **AND NOT** MS_12 **AND (F_TRIG at** MS_21 **OR F_TRIG at** MS_22)

Backward Direction
Muting condition 11 (to 8122) (MS_21 is the first entry switch actuated). Start timers MaxMutingTime and DiscTime21_22**:** MutingEnable AND (**NOT** MS_22 **AND R_TRIG at** MS_21 **AND NOT** MS_11 **AND NOT** MS_12)
Muting condition 11 (to 8422) (MS_22 is the first entry switch actuated). Start timers MaxMutingTime and DiscTime21_22**:** MutingEnable AND (**R_TRIG at** MS_22 **AND NOT** MS_21 **AND NOT** MS_11 **AND NOT** MS_12)

Muting condition 12 (from 8122) (MS_22 is the second entry switch actuated). Stop timer DiscTime21_22: MutingEnable AND (MS_21 **AND R_TRIG at** MS_22 **AND NOT** MS_11 **AND NOT** MS_12)
Muting condition 12 (from 8422) (MS_21 is the second entry switch actuated) . Stop timer DiscTime21_22: MutingEnable AND (**R_TRIG at** MS_21 **AND** MS_22 **AND NOT** MS_11 **AND NOT** MS_12)

Muting condition 13 (both entry switches actuated in same cycle). Start timer MaxMutingTime**:** MutingEnable AND (**R_TRIG at** MS_21 **AND R_TRIG at** MS_22 **AND NOT** MS_11 **AND NOT** MS_12)

Muting condition 14 (all switches actuated)**:** MS_11 **AND** MS_12 **AND** MS_21 **AND** MS_22
Muting condition 44 (to 8114) (MS_11 is the first exit switch actuated). Start timer DiscTime11_12**:** MS_21 **AND** MS_22 **AND R_TRIG at** MS_11 **AND NOT** MS_12
Muting condition 44 (to 8414) (MS_12 is the first exit switch actuated). Start timer DiscTime11_12**:** MS_21 **AND** MS_22 **AND NOT** MS_11 **AND R_TRIG at** MS_12
Muting condition 45 (from 8114) (MS_12 is the second exit switch actuated). Stop timer DiscTime11_12**:** MS_21 **AND** MS_22 **AND** MS_11 **AND R_TRIG at** MS_12
Muting condition 45 (from 8414) (MS_11 is the second exit switch actuated). Stop timer DiscTime11_12**:** MS_21 **AND** MS_22 **AND R_TRIG at** MS_11 **AND** MS_12

Muting condition 15 (one of the exit switches released). Stop timer MaxMutingTime**: NOT** MS_21 **AND NOT** MS_22 **AND (F_TRIG at** MS_11 **OR F_TRIG at** MS_12)

Wrong Muting Sequences:

State 8000:
(MutingEnable = FALSE when muting sequence starts) OR
((MS_11 **OR** MS_12) **AND** (MS_21 **OR** MS_22)) **OR**
(R_TRIG at MS_11 AND MS_12 AND NOT R_TRIG at MS_12) OR
(R_TRIG at MS_12 AND MS_11 AND NOT R_TRIG at MS_11) OR
(R_TRIG at MS_21 AND MS_22 AND NOT R_TRIG at MS_22) OR
(R_TRIG at MS_22 AND MS_21 AND NOT R_TRIG at MS_21) OR
((MS_11 AND NOT R_TRIG at MS_11) AND (MS_12 AND NOT R_TRIG at MS_12)) OR
((MS_21 AND NOT R_TRIG at MS_21) AND (MS_22 AND NOT R_TRIG at MS_22))
State 8011: NOT MutingEnable OR **NOT** MS_11 **OR** MS_21 **OR** MS_22
State 8311: NOT MutingEnable OR **NOT** MS_12 **OR** MS_21 **OR** MS_22
State 8012: **NOT** MS_11 **OR** NOT MS_12
State 8021: **R_TRIG at** MS_11 **OR R_TRIG at** MS_12 **OR R_TRIG at** MS_21 **OR R_TRIG at** MS_22
State 8014: **NOT** MS_11 **OR NOT** MS_12 **OR NOT** MS_21
State 8314: **NOT** MS_11 **OR NOT** MS_12 **OR NOT** MS_22

State 8122: NOT MutingEnable OR MS_11 **OR** MS_12 **OR NOT** MS_21
State 8422: NOT MutingEnable OR MS_11 **OR** MS_12 **OR NOT** MS_22
State 8121: **NOT** MS_21 **OR NOT** MS_22
State 8112: **R_TRIG at** MS_11 **OR R_TRIG at** MS_12 **OR R_TRIG at** MS_21 **OR R_TRIG at** MS_22
State 8114: **NOT** MS_21 **OR NOT** MS_22 **OR NOT** MS_11
State 8414: **NOT** MS_21 **OR NOT** MS_22 **OR NOT** MS_12

Typical Timing Diagram

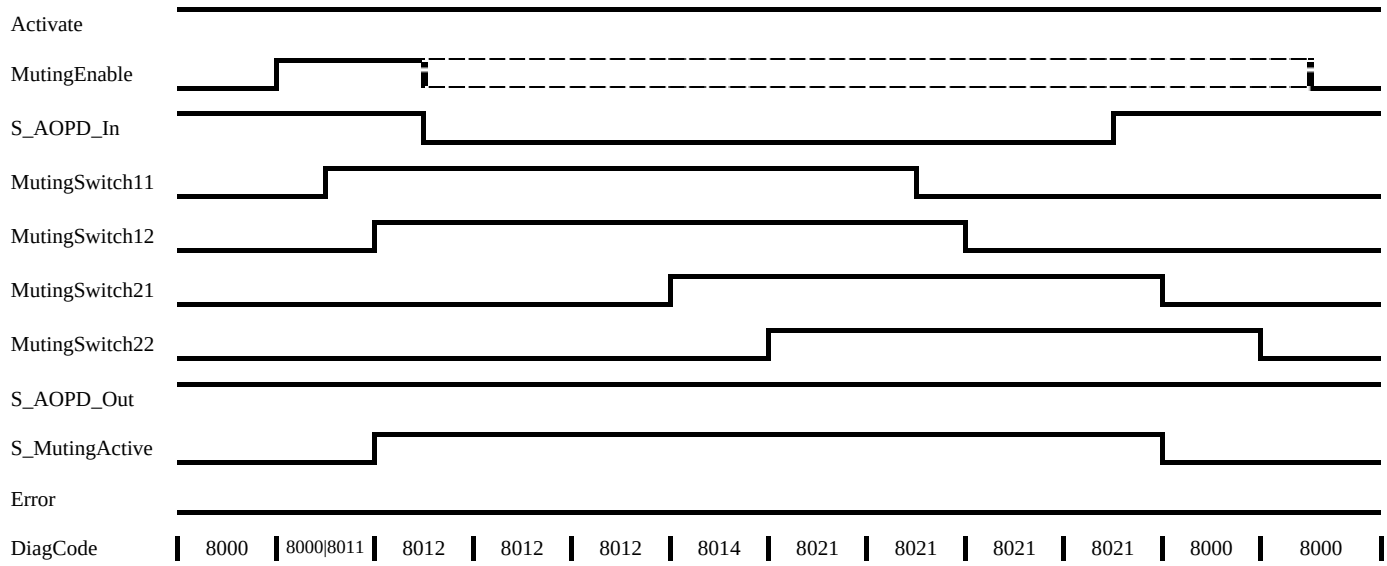


Figure 47: Timing diagram for SF_MutingPar

6.15.4. Error Detection

The FB detects the following error conditions:

- DiscTime11_12 and DiscTime21_22 have been set to values less than T#0s or greater than T#4s.
- MaxMutingTime has been set to a value less than T#0s or greater than T#10min.
- The discrepancy time for the MutingSwitch11/MutingSwitch12 or MutingSwitch21/MutingSwitch22 sensor pairs has been exceeded.
- The muting function (S_MutingActive = TRUE) exceeds the maximum muting time MaxMutingTime.
- Muting sensors MutingSwitch11, MutingSwitch12, MutingSwitch21, and MutingSwitch22 are activated in the wrong order.
- Muting sequence starts without being enabled by MutingEnable
- A faulty muting lamp is indicated by S_MutingLamp = FALSE.
- A static Reset condition is detected in state 8001 and 8003.

6.15.5. Error Behavior

In the event of an error, the S_AOPD_Out and S_MutingActive outputs are set to FALSE. The DiagCode output indicates the relevant error code and the Error output is set to TRUE.
A restart is inhibited until the error conditions are cleared and the Safe state is acknowledged with Reset by the operator.

6.15.6. Function Block-Specific Error and Status Codes

DiagCode	**State Name**	**State Description and Output Setting**

FB-specific error codes:

C001	Reset Error 1	Static Reset condition detected after FB activation in state 8001. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C002	Reset Error 2	Static Reset condition detected in state 8003. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C003	Error Muting Lamp	Error detected in muting lamp. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

CYx4	Error Muting sequence	Error detected in muting sequence state 8000, 8011, 8311, 8012, 8021, 8014, 8314, 8122, 8422, 8121, 8112, 8114 or 8414. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE Y = Status in the sequence (6 states for forward and 6 states for backward direction). C0x4 = Error occurred in state 8000 C1x4 = Error occurred in state Forward 8011 C2x4 = Error occurred in state Forward 8311 C3x4 = Error occurred in state Forward 8012 C4x4 = Error occurred in state Forward 8014 C5x4 = Error occurred in state Forward 8314 C6x4 = Error occurred in state Forward 8021 C7x4 = Error occurred in state Backward 8122 C8x4 = Error occurred in state Backward 8422 C9x4 = Error occurred in state Backward 8121 CAx4 = Error occurred in state Backward 8114 CBx4 = Error occurred in state Backward 8414 CCx4 = Error occurred in state Backward 8112 CFx4 = Muting Enable missing x = Status of the sensors when error occurred (4 bits: LSB = MS_11; MS_12; MS_21; MSB = MS_22).
C005	Parameter Error	DiscTime11_12, DiscTime21_22 or MaxMutingTime value out of range. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C006	Error Timer MaxMuting	Timing error: Active muting time (when S_MutingActive = TRUE) exceeds MaxMutingTime. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C007	Error Timer MS11_12	Timing error: Discrepancy time for switching MutingSwitch11 and MutingSwitch12 > DiscTime11_12. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C008	Error Timer MS21_22	Timing error: Discrepancy time for switching MutingSwitch21 and MutingSwitch22 > DiscTime21_22. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE

8000	AOPD Free	Muting not active and no safety demand from AOPD. If timers from subsequent muting are still running, they are stopped. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8001	Init	Function block has been activated. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8002	Safety Demand AOPD	Safety demand detected by AOPD, muting not active. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8003	Wait for Reset	Safety demand or errors have been detected and are now cleared. Operator acknowledgment by Reset required. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8005	Safe	Safety function activated. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8011	Muting Forward Start 1	Muting forward sequence is in starting phase after rising trigger of MutingSwitch 11. Monitoring of DiscTime11_12 is activated. Monitoring of MaxMutingTime is activated. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8311	Muting Forward Start 2	Muting forward sequence is in starting phase after rising trigger of MutingSwitch 12. Monitoring of DiscTime11_12 is activated. Monitoring of MaxMutingTime is activated. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8012	Muting Forward Active 1	Muting forward sequence is active either: - After rising trigger of the second entry MutingSwitch 12 or 11 has been detected. - When both MutingSwitch 11 and 12 have been actuated in the same cycle. Monitoring of DiscTime11_12 is stopped. Monitoring of MaxMuting-Time is activated, when transition came directly from state 8000. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
8014	Muting Forward Step 1	Muting forward sequence is active. MutingSwitch21 is the first exit switch actuated. Monitoring of DiscTime21_22 is started. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE

8314	Muting Forward Step 2	Muting forward sequence is active. MutingSwitch22 is the first exit switch actuated. Monitoring of DiscTime21_22 is started. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
8021	Muting Forward Active 2	Muting forward sequence is still active. Both MutingSwitch21 and 22 are actuated, the monitoring of DiscTime21_22 is stopped. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
8122	Muting Backward Start 1	Muting backward sequence is in starting phase after rising trigger of MutingSwitch21. Monitoring of DiscTime21_22 is activated. Monitoring of MaxMutingTime is activated. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8422	Muting Backward Start 2	Muting backward sequence is in starting phase after rising trigger of MutingSwitch22. Monitoring of DiscTime21_22 is activated. Monitoring of MaxMutingTime is activated. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8121	Muting Backward Active 1	Muting backward sequence is active either: - After rising trigger of the second MutingSwitch 21 or 22 has been detected. - When both MutingSwitch 21 and 22 have been actuated in the same cycle. Monitoring of DiscTime21_22 is stopped. Monitoring of MaxMuting-Time is activated, when transition came directly from state 8000. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
8114	Muting Backward Step 1	Muting backward sequence is active. MutingSwitch11 is the first exit switch actuated. Monitoring of DiscTime11_12 is started. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
8414	Muting Backward Step 2	Muting backward sequence is active. MutingSwitch12 is the first exit switch actuated. Monitoring of DiscTime11_12 is started. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
8112	Muting Backward Active 2	Muting backward sequence is still active. Both exit switches MutingSwitch11 and 12 are actuated, the monitoring of DiscTime11_12 is stopped. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE

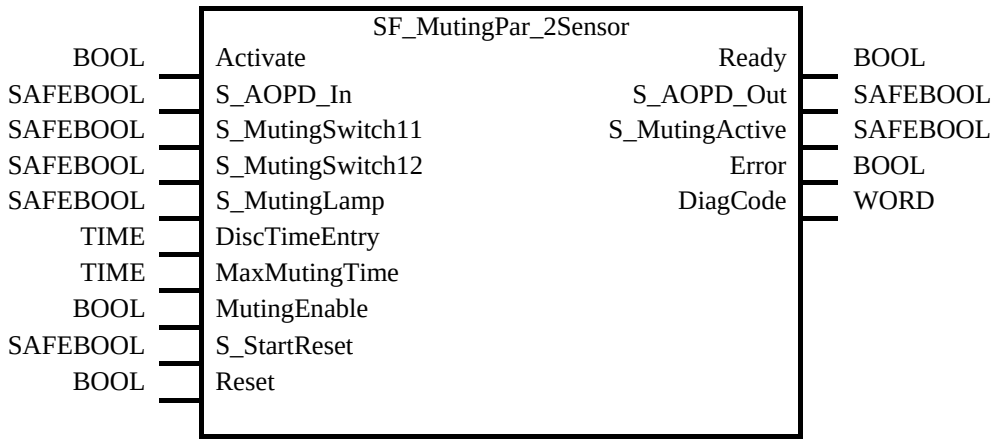
6.16. Parallel Muting with 2 Sensors

6.16.1. Applicable Safety Standards

Standards	Requirements
IEC 61496-1:2004	A.7 Muting, A.7.1.2 There shall be at least two independent hard-wired muting signal sources to initiate the function. It shall not be possible to initiate muting when the OSSDs are already in the OFF-state. A.7.1.3 The mute function shall only be inititated by the correct sequence and/or timing of the mute signals. Should conflicting muting signals occur, the ESPE shall not allow a muted condition to occur. A.7.1.4 There shall be at least two independent hard-wired muting signal sources to stop the function. The muting function shall stop when the first of these muting signals changes state. The deactivation of the muting function shall not rely only on the clearance of the ESPE. A.7.1.5 The muting signals should be continuously present during muting. When the signals are not continuously present, an incorrect sequence and/or the expiration of a pre-set time limit shall cause either a lock-out condition or a restart interlock. A.7.4 Indication: A mute status signal or indicator shall be provided (in some applications, an indication signal of muting is necessary
CD IEC 62046/Ed. 1: 2005	5.5.1: .. an indicator to show when the muting function is active can be necessary. The muting function shall be initiated and terminated automatically….Incorrect signals, sequence, or timing of the muting sensors or signals shall not allow a mute condition. It shall not be possible to initiate the muting function when: – the protective equipment OSSDs are in the OFF-state; – the protective equipment is in the lock-out condition. - initiation of the muting function by two or more independent muting sensors such that a single fault cannot cause a muted condition; - termination of the muting function by two or more independent muting sensors such that deactivation of one sensor will terminate the muting function; - use of timing and sequence control of the muting sensors to ensure correct muting operation; 5.5.3: The following measures shall be considered:… - limiting muting to a fixed time that is only sufficient for the material to pass through the detection zone. When this time is exceeded, the muting function should be cancelled and all hazardous movements stopped; Annex F.7 Two sensors – Crossed beams (see also Fig. F.7.2 and F.7.3) The muting function should only be initiated when the two beams are activated within a time limit of 4 sec. The muting function should be terminated as soon as one of the two beams of the muting sensors is no longer activated. A monitored timer that limits the muting function to the minimum time practicable is required. Annex F.5: Methods to avoid manipulation of the muting function: … use a muting enable command generated by the control system of the machine that will only enable the muting function when needed by the machine cycle.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.16.2. Interface Description

FB Name	**SF_MutingPar_2Sensor**		
Muting is the intended suppression of the safety function. In this FB, parallel muting with two muting sensors is specified.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_AOPD_In	SAFEBOOL	FALSE	Variable. OSSD signal from AOPD. FALSE: Protection field interrupted. TRUE: Protection field not interrupted.
S_MutingSwitch11	SAFEBOOL	FALSE	Variable. Status of Muting sensor 11. FALSE: Muting sensor 11 not actuated. TRUE: Workpiece actuates muting sensor 11.
S_MutingSwitch12	SAFEBOOL	FALSE	Variable. Status of Muting sensor 12. FALSE: Muting sensor 12 not actuated. TRUE: Workpiece actuates muting sensor 12.
S_MutingLamp	SAFEBOOL	FALSE	Variable or constant. Indicates operation of the muting lamp. FALSE: Muting lamp failure. TRUE: Muting lamp no failure.
DiscTimeEntry	TIME	T#0s	Constant 0..4 s; Max. discrepancy time for S_MutingSwitch11 and S_MutingSwitch12 entering muting gate
MaxMutingTime	TIME	T#0s	Constant 0..10 min; Maximum time for complete muting sequence, timer started when first muting sensor is actuated.
MutingEnable	BOOL	FALSE	Variable or constant. Command by the control system that enables the start of the muting function when needed by the machine cycle. After the start of the muting function, this signal can be switched off. FALSE: Muting not enabled TRUE: Start of Muting function enabled
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_AOPD_Out	SAFEBOOL	FALSE	Safety related output, indicates status of the muted guard. FALSE: AOPD protection field interrupted and muting not active. TRUE: AOPD protection field not interrupted or muting active.
S_MutingActive	SAFEBOOL	FALSE	Indicates status of Muting process. FALSE: Muting not active. TRUE: Muting active.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: Line control of muting sensor signals must be active in the safety loop			



6.16.3. Functional Description

Muting is the intended suppression of the safety function. This is required, e.g., when transporting the material into the danger zone without causing the machine to stop. Muting is triggered by muting sensors. The use of two muting sensors and correct integration into the production sequence must ensure that no persons enter the danger zone while the light curtain is muted. Muting sensors can be push buttons, proximity switches, photoelectric barriers, limit switches, etc. which do not have to be failsafe. Active muting mode must be indicated by indicator lights.

There are sequential and parallel muting procedures. In this FB, parallel muting with two muting sensors was used; an explanation is provided below. The positioning of the sensors should be as described in Annex F.7 of IEC 62046, CD 2005, as shown in Figure 48. The FB can be used in both directions, forward and backward. However, the actual direction cannot be identified. The muting should be enabled with the MutingEnable signal by the process control to avoid manipulation.

The FB input parameters include the signals of the two muting sensors (S_MutingSwitch11 and S_MutingSwitch12), the OSSD signal from the "active opto-electronic protective device", S_AOPD_In, as well as two parameterizable times (DiscTimeEntry and MaxMutingTime).

The S_StartReset input shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

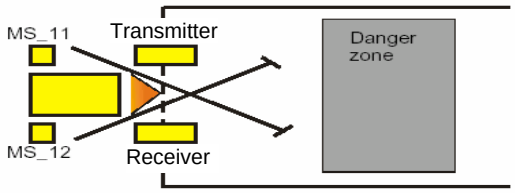
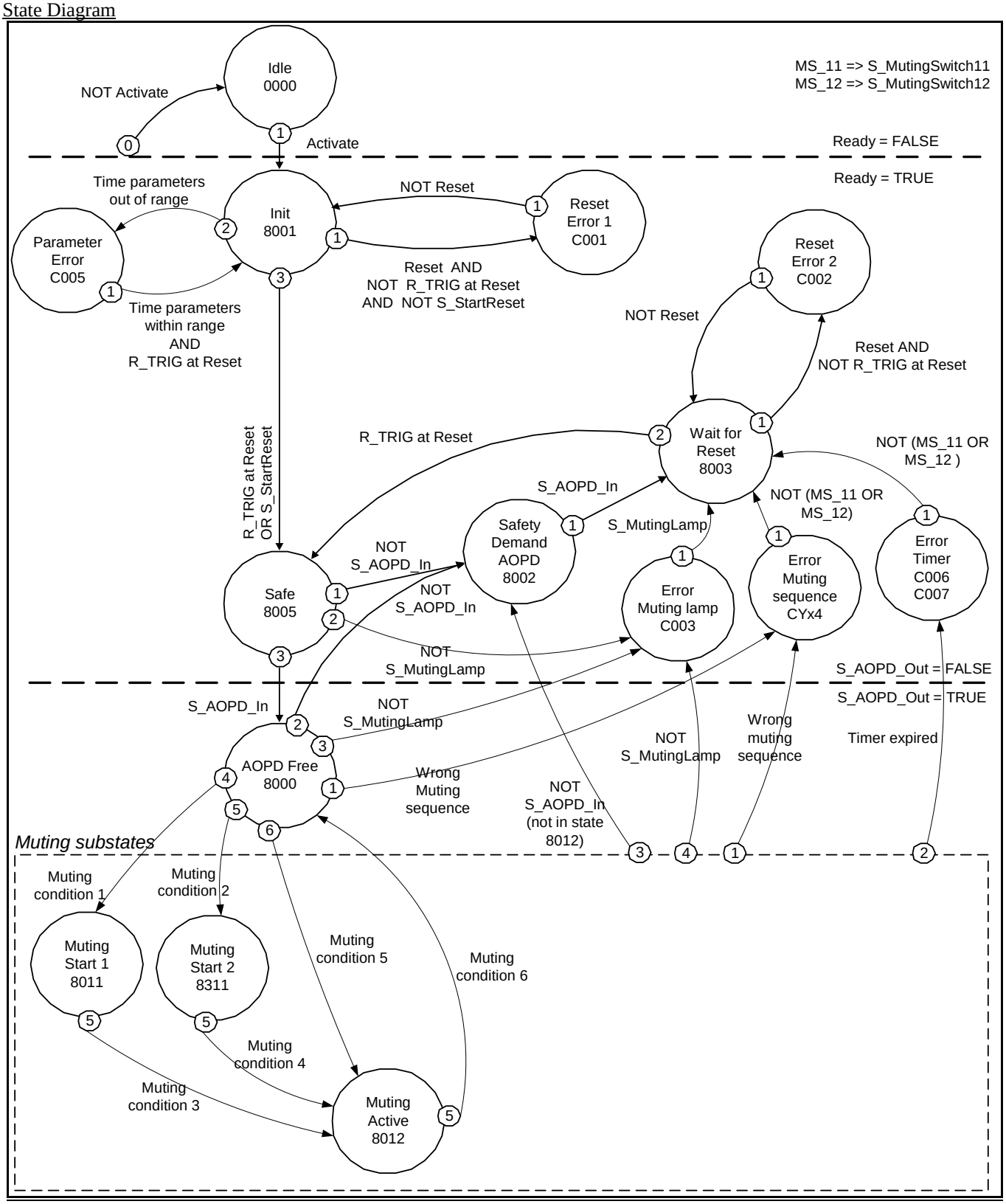
No.	Figure	Explanation
1		If reflection light barriers are used as muting sensors, they are generally arranged diagonally. In general, this arrangement of reflection light barriers as muting sensors requires only two light barriers, and only S_MutingSwitch11 (MS_11) and S_MutingSwitch12 (MS_12) are allocated.

Figure 48: Example for SF_MutingPar_2Sensor with two reflecting light barriers

State Diagram



Note1: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Note 2: Within muting substates, transitions due to Error Muting sequence (priority 1), Error Timer (priority 2), Safety demand AOPD (priority 3) or Error Muting lamp (priority 4) have higher priority than transitions to Muting substates (priority 5).

Note 3: Muting conditions are defined below.

Figure 49: State diagram for SF_MutingPar_2Sensor

Muting conditions:

Muting condition 1 (to 8011) (MS_11 is the first entry switch actuated). Start timer DiscTimeEntry and MaxMutingTime**:** MutingEnable **AND R_TRIG at** MS_11 **AND NOT** MS_12
Muting condition 2 (to 8311) (MS_12 is the first entry switch actuated). Start timer DiscTimeEntry and MaxMutingTime**:** MutingEnable **AND NOT** MS_11 **AND R_TRIG at** MS_12
Muting condition 3 (from 8011 to 8012) (MS_12 is the second entry switch actuated)**:** Stop timer DiscTimeEntry: MutingEnable **AND** MS_11 **AND R_TRIG at** MS_12
Muting condition 4 (from 8311 to 8012) (MS_11 is the second entry switch actuated)**:** Stop timer DiscTimeEntry: MutingEnable **AND R_TRIG at** MS_11 **AND** MS_12
Muting condition 5 (from 8000 to 8012) (both switches actuated in same cycle)**:** Start Timer MaxMutingTime: MutingEnable **AND R_TRIG at** MS_11 **AND R_TRIG at** MS_12
Muting condition 6 (from 8012 to 8000) (both switches released in same cycle or MS_11 and MS_12 released consecutively). Stop timer MaxMutingTime**: NOT** MS_11 **OR NOT** MS_12

Wrong Muting Sequences:

State 8000: (**R_TRIG** at MS_11 **AND** MS_12 **AND NOT R_TRIG** at MS_12) **OR**
(**R_TRIG** at MS_12 **AND** MS_11 **AND NOT R_TRIG** at MS_11) **OR**
((MS_11 **AND NOT R_TRIG** at MS_11) **AND** (MS_12 **AND NOT R_TRIG** at MS_12)) **OR**
(**NOT** MutingEnable **AND R_TRIG** at MS_11) **OR**
(**NOT** MutingEnable **AND R_TRIG** at MS_12)

State 8011: **NOT** MutingEnable **OR NOT** MS_11
State 8311: **NOT** MutingEnable **OR NOT** MS_12
State 8012: **all possible transitions allowed**

Typical Timing Diagram

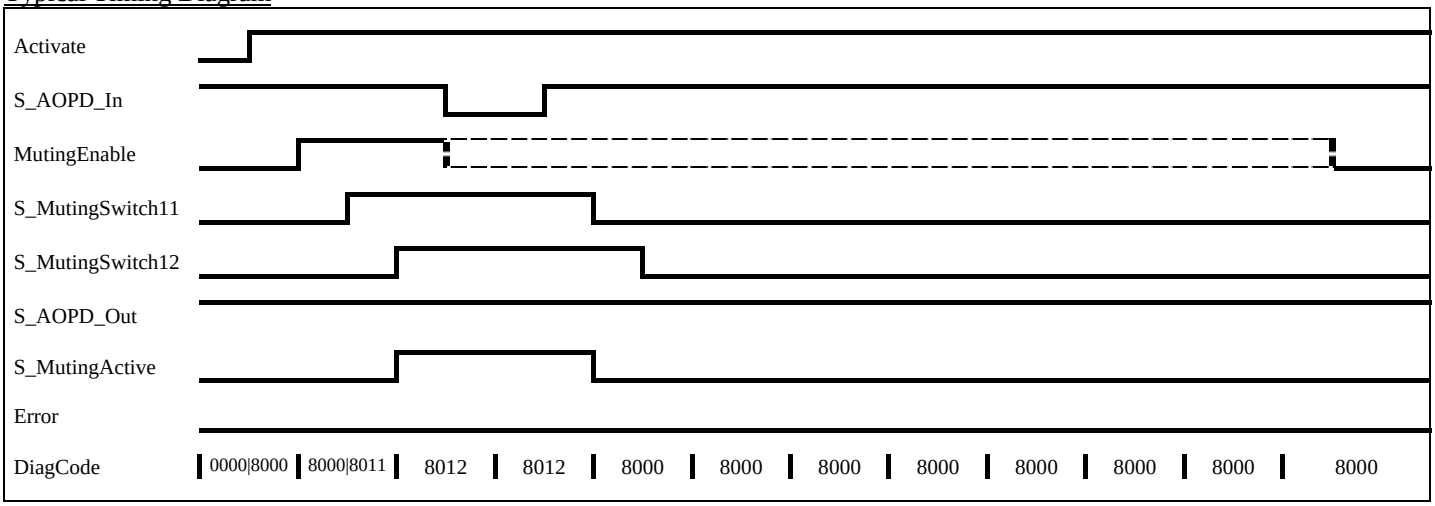


Figure 50: Timing diagram for SF_MutingPar_2Sensor (S_StartReset = TRUE, Reset = FALSE, S_MutingLamp = TRUE)

6.16.4. Error Detection

The FB detects the following error conditions:

- DiscTimeEntry has been set to value less than T#0s or greater than T#4s.
- MaxMutingTime has been set to a value less than T#0s or greater than T#10min.
- The discrepancy time for the S_MutingSwitch11/S_MutingSwitch12 sensor pair has been exceeded.
- The muting function (S_MutingActive = TRUE) exceeds the maximum muting time MaxMutingTime.
- Muting sensors S_MutingSwitch11,S_MutingSwitch12 are activated in the wrong order.
- Muting sequence starts without being enabled by MutingEnable
- Static muting sensor signals.
- A faulty muting lamp is indicated by S_MutingLamp = FALSE.
- A static Reset condition is detected in state 8001 and 8003.

6.16.5. Error Behavior

In the event of an error, the S_AOPD_Out and S_MutingActive outputs are set to FALSE. The DiagCode output indicates the relevant error code and the Error output is set to TRUE.
A restart is inhibited until the error conditions are cleared and the Safe state is acknowledged with Reset by the operator.

6.16.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting

FB-specific error codes:

C001	Reset Error 1	Static Reset condition detected after FB activation in state 8001. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C002	Reset Error 2	Static Reset condition detected in state 8003. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C003	Error Muting Lamp	Error detected in muting lamp. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
CYx4	Error Muting sequence	Error detected in muting sequence state 8000, 8011, 8311. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE Y = Status in the sequence C0x4 = Error occurred in state 8000 C1x4 = Error occurred in state 8011 C2x4 = Error occurred in state 8311 CFx4 = Muting Enable missing x = Status of the sensors when error occurred (4 bits: LSB = MS_11; next to LSB = MS_12).
C005	Parameter Error	DiscTimeEntry or MaxMutingTime value out of range. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C006	Error timer MaxMuting	Timing error: Active muting time (when S_MutingActive = TRUE) exceeds MaxMutingTime. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
C007	Error timer Entry	Timing error: Discrepancy time for switching S_MutingSwitch11 and S_MutingSwitch12 from FALSE to TRUE > DiscTimeEntry. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8000	AOPD Free	Muting not active and no safety demand from AOPD. If timers from subsequent muting are still running, they are stopped. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8001	Init	Function block was activated. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8002	Safety Demand AOPD	Safety demand detected by AOPD, muting not active. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8003	Wait for Reset	Safety demand or errors have been detected and are now cleared. Operator acknowledgment by Reset required. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8005	Safe	Safety function activated. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
8011	Muting Start 1	Muting sequence is in starting phase after rising trigger of S_MutingSwitch11. Monitoring of DiscTimeEntry is activated. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8311	Muting Start 2	Muting sequence is in starting phase after rising trigger of S_MutingSwitch12. Monitoring of DiscTimeEntry is activated. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
8012	Muting Active	Muting sequence is active either: - After rising trigger of the second S_MutingSwitch 12 or 11 has been detected. - When both S_MutingSwitch 11 and 12 have been actuated in the same cycle. Monitoring of DiscTimeEntry is stopped. Monitoring of MaxMutingTime is activated. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE

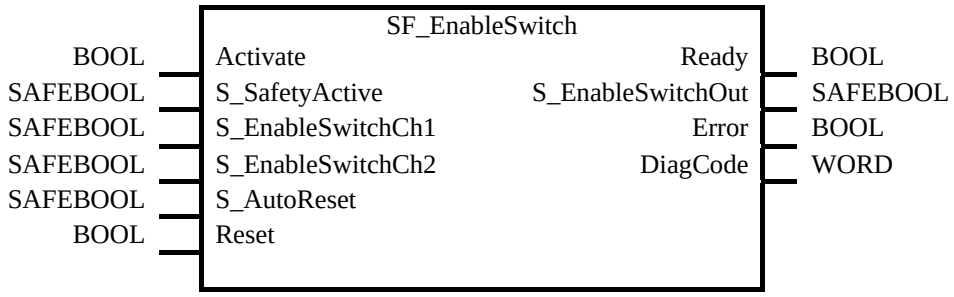
6.17. Enable Switch

6.17.1. Applicable Safety Standards

Standards	**Requirements**
IEC 60204-1, Ed. 5.0: 2003	9.2.6.3: Enabling control (see also 10.9) is a manually activated control function interlock that: a) when activated allows a machine operation to be initiated by a separate start control, and b) when de-activated – initiates a stop function, and – prevents initiation of machine operation. Enabling control shall be so arranged as to minimize the possibility of defeating, for example by requiring the de-activation of the enabling control device before machine operation may be reinitiated. It should not be possible to defeat the enabling function by simple means. 10.9: When an enabling control device is provided as a part of a system, it shall signal the enabling control to allow operation when actuated in one position only. In any other position, operation shall be stopped or prevented. Enabling control devices shall be selected that have the following features: … – for a three-position type: - position 1: off-function of the switch (actuator is not operated); - position 2: enabling function (actuator is operated in its mid position); - position 3: off-function (actuator is operated past its mid position); - when returning from position 3 to position 2, the enabling function is not activated.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.17.2. Interface Description

FB Name	**SF_EnableSwitch**		
The SF_EnableSwitch FB evaluates the signals of an enable switch with three positions.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, parameter values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_SafetyActive	SAFEBOOL	FALSE	Variable or constant. Confirmation of the safe mode (limitation of the speed or the power of motion, limitation of the range of motion). FALSE: Safe mode is not active. TRUE: Safe mode is active.
S_EnableSwitchCh1	SAFEBOOL	FALSE	Variable. Signal of contacts E1 and E2 of the connected enable switch. FALSE: Connected switches are open. TRUE: Connected switches are closed.
S_EnableSwitchCh2	SAFEBOOL	FALSE	Variable. Signal of contacts E3 and E4 of the connected enable switch. FALSE: Connected switches are open. TRUE: Connected switches are closed.
S_AutoReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_EnableSwitchOut	SAFEBOOL	FALSE	Safety related output: Indicates suspension of guard. FALSE: Disable suspension of safeguarding. TRUE: Enable suspension of safeguarding.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: -			



6.17.3. Functional Description

The SF_EnableSwitch FB supports the suspension of safeguarding (DIN EN 60204 Section 9.2.4) using enable switches (DIN EN 60204 Section 9.2.5.8), if the relevant operating mode is selected and active. The relevant operating mode (limitation of the speed or the power of motion, limitation of the range of motion) must be selected outside the SF_EnableSwitch FB.
The SF_EnableSwitch FB evaluates the signals of an enable switch with three positions (DIN EN 60204 Section 9.2.5.8).
The S_EnableSwitchCh1 and S_EnableSwitchCh2 input parameters process the following signal levels of contacts E1 to E4:

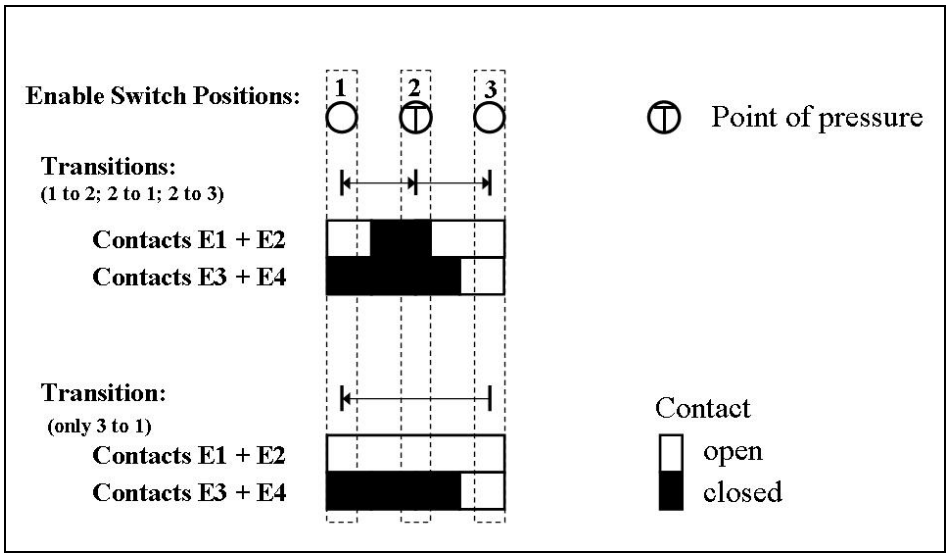


Figure 51: Switch positions

The signal from E1+E2 must be connected to the S_EnableSwitchCh1 parameter. The signal from E3+E4 must be connected to the S_EnableSwitchCh2 parameter. The position of the enable switch is detected in the FB using this signal sequence. The transition from position 2 to 3 can be different from shown here.

The switching direction (position 1 => position 2/position 3 => position 2) can be detected in the FB using the defined signal sequence of the enable switch contacts. The suspension of safeguarding can only be enabled by the FB after a move from position 1 to position 2. Other switching directions or positions may not be used to enable the suspension of safeguarding. This measure meets the requirements of EN 60204 Section 9.2.5.8.

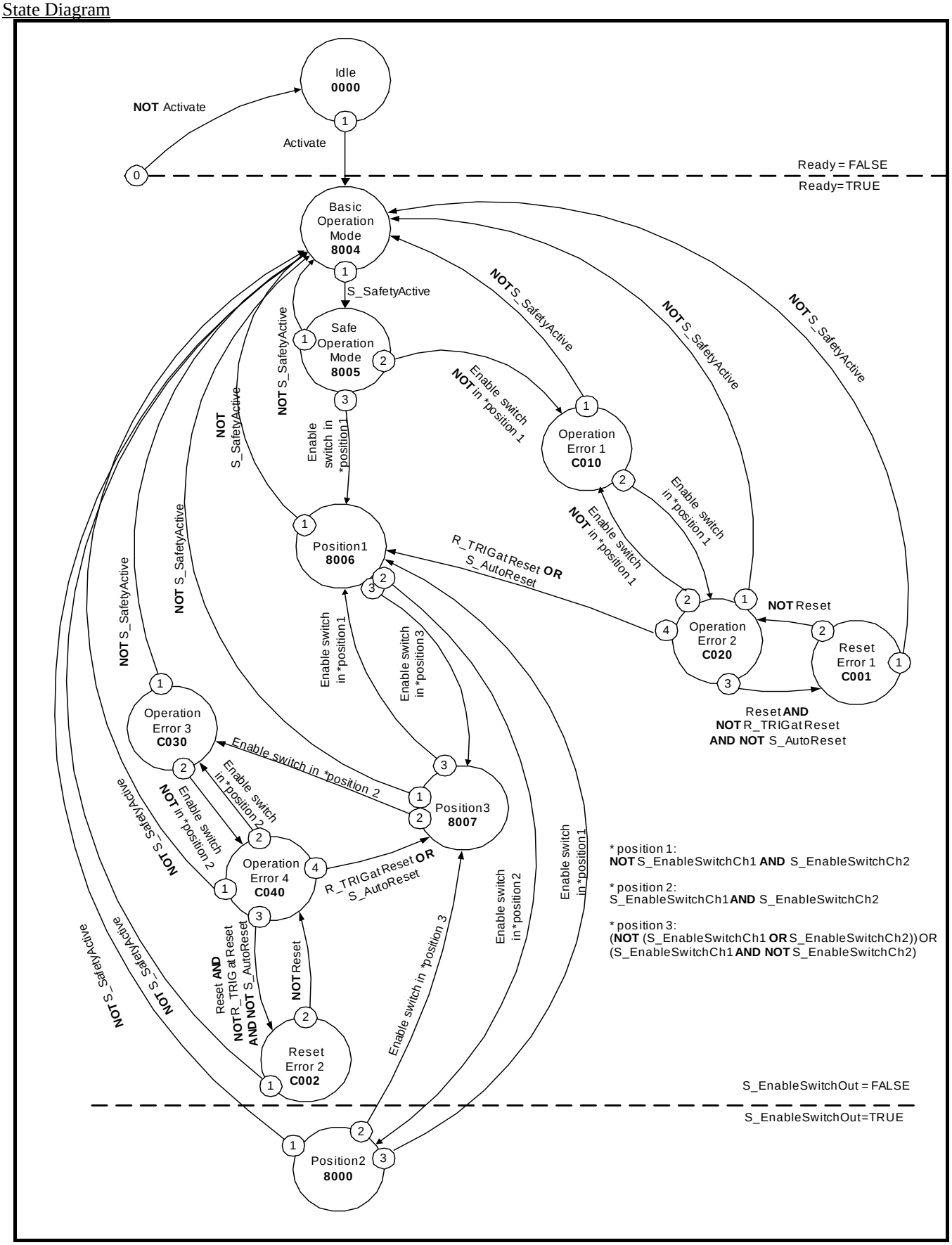
In order to meet the requirements of DIN EN 60204 Section 9.2.4, the user shall use a suitable switching device. In addition, the user must ensure that the relevant operating mode (DIN EN 60204 Section 9.2.3) is selected in the application (automatic operation must be disabled in this operating mode using appropriate measures).

The operating mode is usually specified using an operating mode selection switch in conjunction with the SF_ModeSelector FB and the SF_SafeRequest or SF_SafelyLimitedSpeed FB.

The SF_EnableSwitch FB processes the confirmation of the "safe mode" state via the "S_SafetyActive" parameter. On implementation in an application of the safe mode without confirmation, a static TRUE signal is connected to the "S_SafetyActive" parameter.

The S_AutoReset input shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

State Diagram

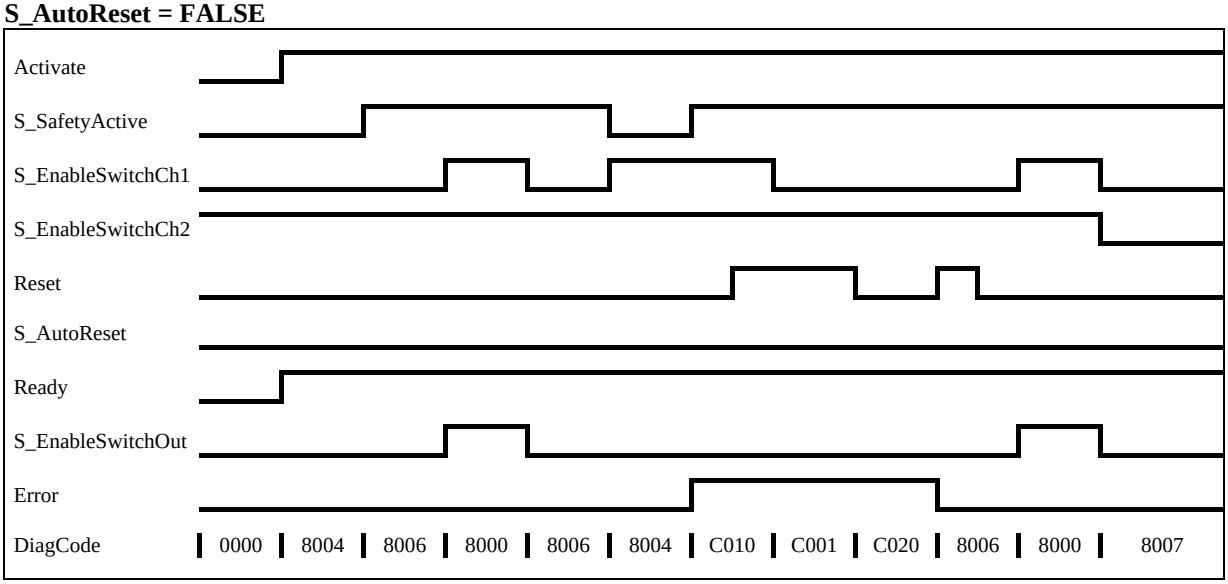


Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 52: State diagram for SF_EnableSwitch

Typical Timing Diagrams

S_AutoReset = FALSE



S_AutoReset = TRUE

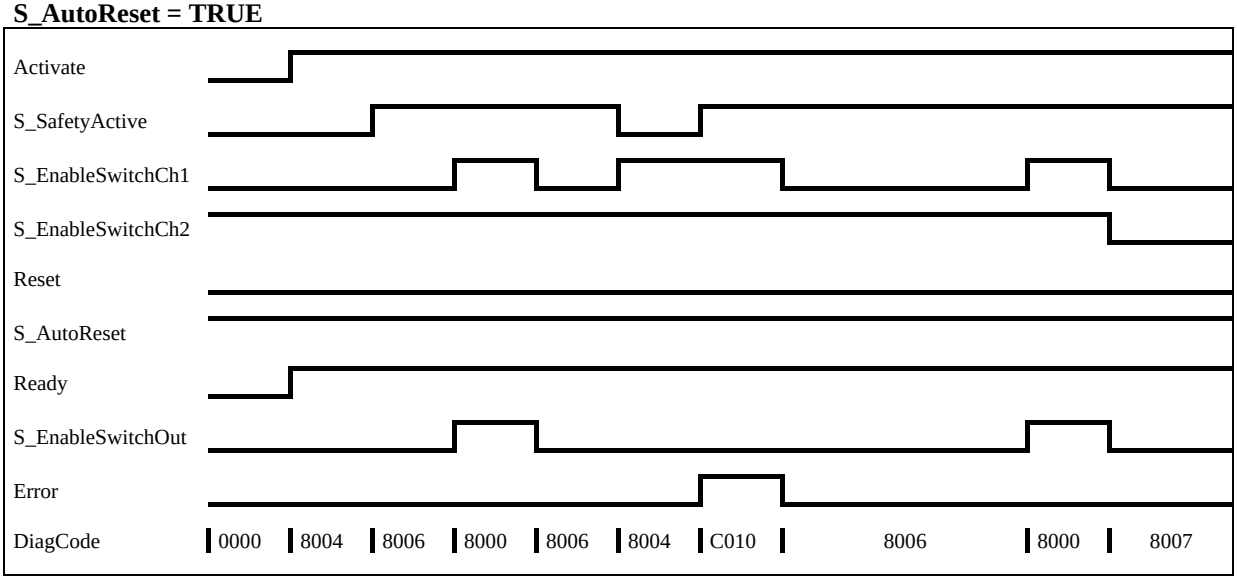


Figure 53: Timing diagram for SF_EnableSwitch.

6.17.4. Error Detection

The following conditions force a transition to the Error state:

- Invalid static Reset signal in the process.
- Invalid switch positions.

6.17.5. Error Behavior

In the event of an error, the S_EnableSwitchOut safe output is set to FALSE and remains in this Safe state.
Different from other FBs, a Reset Error state can be left by the condition Reset = FALSE or, additionally, when the signal S_SafetyActive is FALSE.
Once the error has been removed, the enable switch must be in the initial position specified in the process before the S_EnableSwitchOut output can be set to TRUE using the enable switch. If S_AutoReset = FALSE, a rising trigger is required at Reset.

6.17.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	State Description and Output Setting
FB-specific error codes:		
C001	Reset Error 1	Static Reset signal detected in state C020. Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
C002	Reset Error 2	Static Reset signal detected in state C040. Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
C010	Operation Error 1	Enable switch not in position 1 during activation of S_SafetyActive. Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
C020	Operation Error 2	Enable switch in position 1 after C010. Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
C030	Operation Error 3	Enable switch in position 2 after position 3. Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
C040	Operation Error 4	Enable switch not in position 2 after C030. Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
FB-specific status codes (no error):		
0000	Idle	The function block is not active (initial state). Ready = FALSE S_EnableSwitchOut = FALSE Error = FALSE
8004	Basic Operation Mode	Safe operation mode is not active. Ready = TRUE S_EnableSwitchOut = FALSE Error = FALSE
8005	Safe Operation Mode	Safe operation mode is active. Ready = TRUE S_EnableSwitchOut = FALSE Error = FALSE
8006	Position 1	Safe operation mode is active and the enable switch is in position 1. Ready = TRUE S_EnableSwitchOut = FALSE Error = FALSE
8007	Position 3	Safe operation mode is active and the enable switch is in position 3. Ready = TRUE S_EnableSwitchOut = FALSE Error = FALSE
8000	Position 2	Safe operation mode is active and the enable switch is in position 2. Ready = TRUE S_EnableSwitchOut = TRUE Error = FALSE

6.18. Safety Request

6.18.1. Applicable Safety Standards

Standards	Requirements
IEC 60204-1, Ed. 5.0: 2003	9.2.4 Suspension of safety functions and/or protective measures Where it is necessary to suspend safety functions and/or protective measures (for example for setting or maintenance purposes), protection shall be ensured by: – disabling all other operating (control) modes; and – other relevant means (see 4.11.9 of ISO 12100-2:2003), that can include, for example, one or more of the following: - limitation of the speed or the power of motion; - limitation of the range of motion.
EN 954-1: 1996	5.4 Manual reset
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart

6.18.2. Interface Description

The function block represents the interface between the user program and system environment.

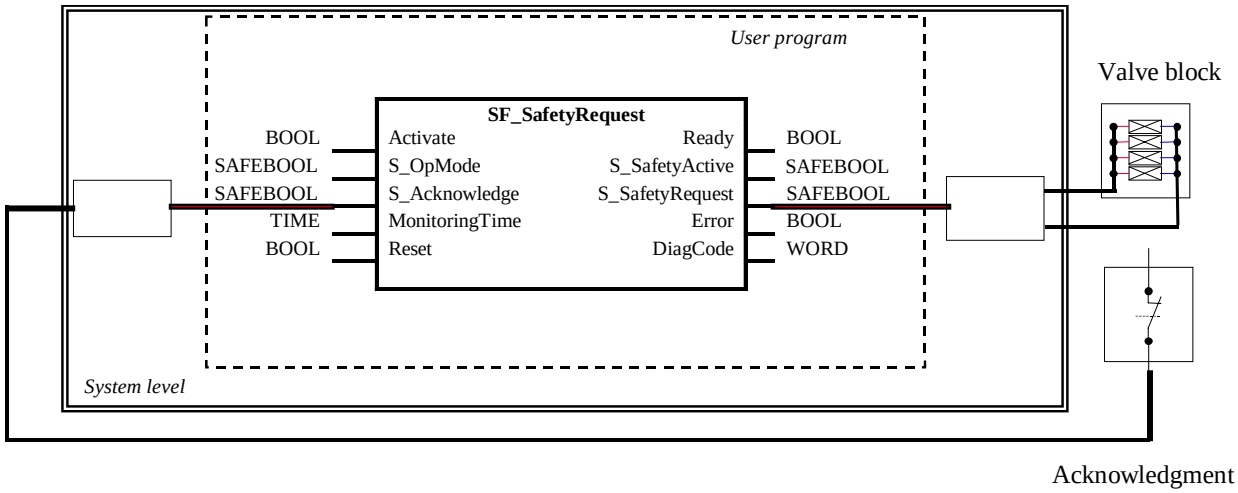
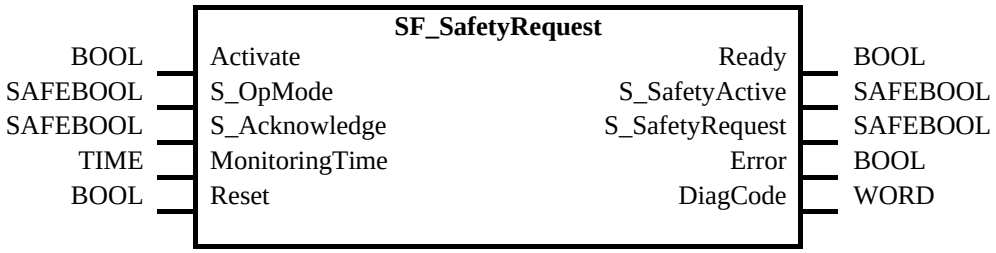


Figure 54: Example SF_SafetyRequest.

FB Name	**SF_SafetyRequest**		
This function block provides the interface to a generic actuator, e.g. a safety drive or safety valve, to place the actuator in a safe state.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_OpMode	SAFEBOOL	FALSE	Variable. Requested mode of a generic safe actuator. FALSE: Safe mode is requested. TRUE: Operation mode is requested.
S_Acknowledge	SAFEBOOL	FALSE	Variable. Confirmation of the generic actuator, if actuator is in the Safe state. FALSE: Operation mode (non-safe). TRUE: Safe mode.
MonitoringTime	TIME	T#0s	Constant. Monitoring of the response time between the safety function request (S_OpMode set to FALSE) and the actuator acknowledgment (S_Acknowledge switches to TRUE).
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_SafetyActive	SAFEBOOL	FALSE	Confirmation of the Safe state. FALSE: Non-safe state. TRUE: Safe state.
S_SafetyRequest	SAFEBOOL	FALSE	Request to place the actuator in a safe state. FALSE: Safe state is requested. TRUE: Non-safe state.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: --			



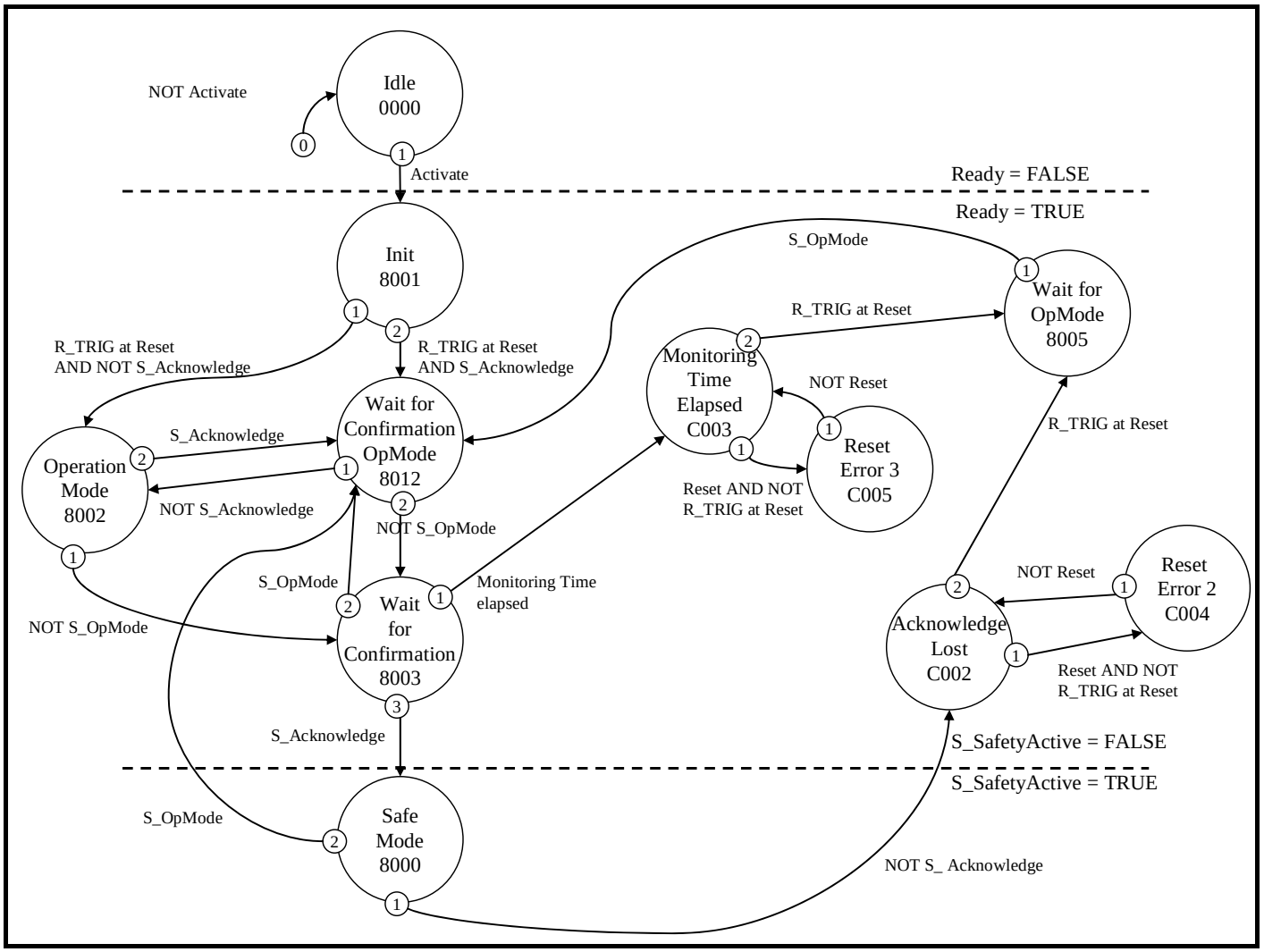
6.18.3. Functional Description

This FB provides the interface between the safety-related system and a generic actuator. This means that the safety-related functions of the actuator are available within the application program. However, there are only two binary signals to control the Safe state of the generic actuator, i.e., one for requesting and one for receiving the confirmation.

The safety function will be provided by the actuator itself. Therefore the FB only initiates the request, monitors it, and sets the output when the actuator acknowledges the Safe state. This will be indicated with the "S_SafetyActive" output.

This FB does not define any generic actuator-specific parameters. They should have been specified in the generic actuator itself. It switches the generic actuator from the operation mode to a safe state.

State Diagram



Note 1: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 55: State diagram for SF_SafetyRequest

Typical Timing Diagram

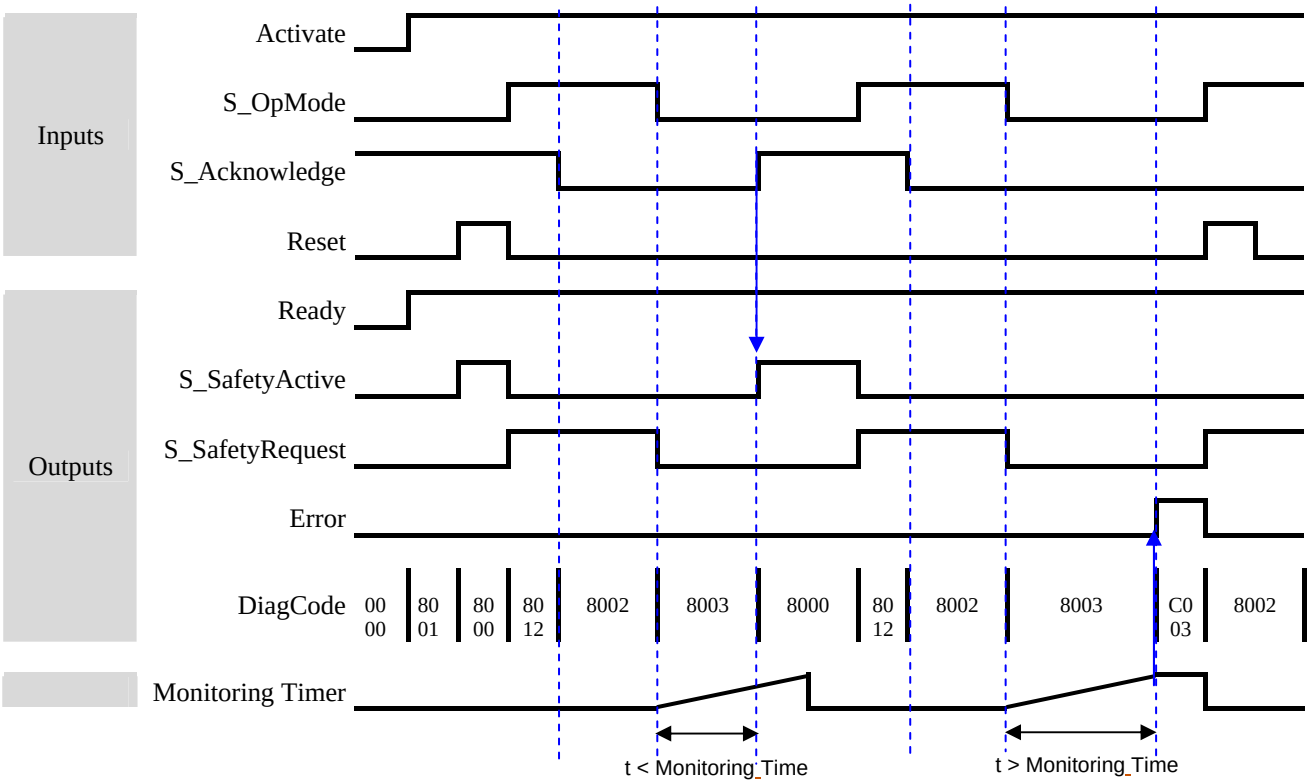


Figure 56: Timing diagram for SF_SafetyRequest

6.18.4. Error Detection

The FB detects whether the actuator does not enter the Safe state within the monitoring time.
The FB detects whether the acknowledge signal is lost while the request is still active.
The FB detects a static Reset signal.

External FB errors:
There are no external errors, since there is no error bits/information provided by the generic actuator.

6.18.5. Error Behavior

In the event of an error, the S_SafetyActive output is set to FALSE.
An error must be acknowledged by a rising trigger at the Reset input. To continue the function block after this reset, the S_OpMode request must be set to TRUE.

6.18.6. Function Block-Specific Error and Status Codes

DiagCode	**State Name**	**State Description and Output Setting**

FB-specific error codes:

C002	Acknowledge Lost	Acknowledgment lost while in the Safe state. Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = TRUE
C003	MonitoringTime Elapsed	S_OpMode request could not be completed within the monitoring time. Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = TRUE

C004	Reset Error 2	Static Reset detected in state C002 (Acknowledge Lost). Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = TRUE
C005	Reset Error 3	Static Reset detected in state C003 (MonitoringTime Elapsed). Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = FALSE
8000	Safe Mode	Actuator is in a safe mode. Ready = TRUE S_SafetyActive = TRUE S_SafetyRequest = FALSE Error = FALSE
8001	Init	State after Activate is set to TRUE or after a rising trigger at Reset. Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = FALSE
8002	Operation Mode	Operation mode without Acknowledge of safe mode Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = TRUE Error = FALSE
8012	Wait for Confirmation OpMode	Operation mode with Acknowledge of safe mode Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = TRUE Error = FALSE
8003	Wait for Confirmation	Waiting for confirmation from the drive (system interface). Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = FALSE
8005	Wait for OpMode	Error was cleared. However S_OpMode must be set to TRUE before the FB can be initialized. Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = FALSE

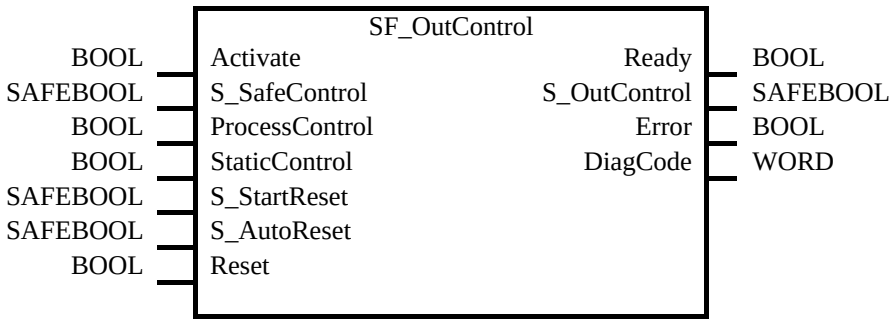
6.19. OutControl

6.19.1. Applicable Safety Standards

Standards	**Requirements**
IEC 60204-1, Ed. 5.0: 2003	9.2.2: Stop functions: Stop function categories; Category 0 - stopping by immediate removal of power to the machine actuators (i.e. an uncontrolled stop …) 9.2.5.2: Start: The start of an operation shall be possible only when all of the relevant safety functions and/or protective measures are in place and are operational except for conditions as described in 9.2.4.. Suitable interlocks shall be provided to secure correct sequential starting.
EN 954-1: 1996	5.2: Stop function; stop initiated by protective devices shall put the machine in a safe state … and shall have priority over a stop for operational reasons 5.5: Start and restart; automatic restart only if a hazardous situation cannot exist. 5.11: Fluctuations in energy levels; in case of loss of energy supply, provide or initiate outputs to maintain a safe state.
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart
EN 954-1: 1996	5.4 Manual reset

6.19.2. Interface Description

FB Name	**SF_OutControl**		
Control of a safety output with a signal from the functional application and a safety signal with optional startup inhibits.			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_SafeControl	SAFEBOOL	FALSE	Variable. Control signal of the preceding safety FB. Typical function block signals from the library (e.g., SF_EStop, SF_GuardMonitoring, SF_TwoHandControlTypeII, and/or others). FALSE: The preceding safety FB's are in safe state. TRUE: The preceding safety FB's enable safety control.
ProcessControl	BOOL	FALSE	Variable or constant. Control signal from the functional application. FALSE: Request to set S_OutControl to FALSE. TRUE: Request to set S_OutControl to TRUE.
StaticControl	BOOL	FALSE	Constant. Optional conditions for process control. FALSE: Dynamic change at ProcessControl (FALSE => TRUE) required after block activation or triggered safety function. Additional function start required. TRUE: No dynamic change at ProcessControl (FALSE => TRUE) required after block activation or triggered safety function.
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
S_AutoReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_OutControl	SAFEBOOL	FALSE	Controls connected actuators. FALSE: Disable connected actuators. TRUE: Enable connected actuators.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: -			



6.19.3. Functional Description

General:

The SF_OutControl FB is an output driver for a safety output.

The safety output is controlled via S_OutControl using a signal from the functional application (ProcessControl/BOOL to control the process) and a signal from the safety application (S_SafeControl/SAFEBOOL to control the safety function).

Optional conditions for process control (ProcessControl):

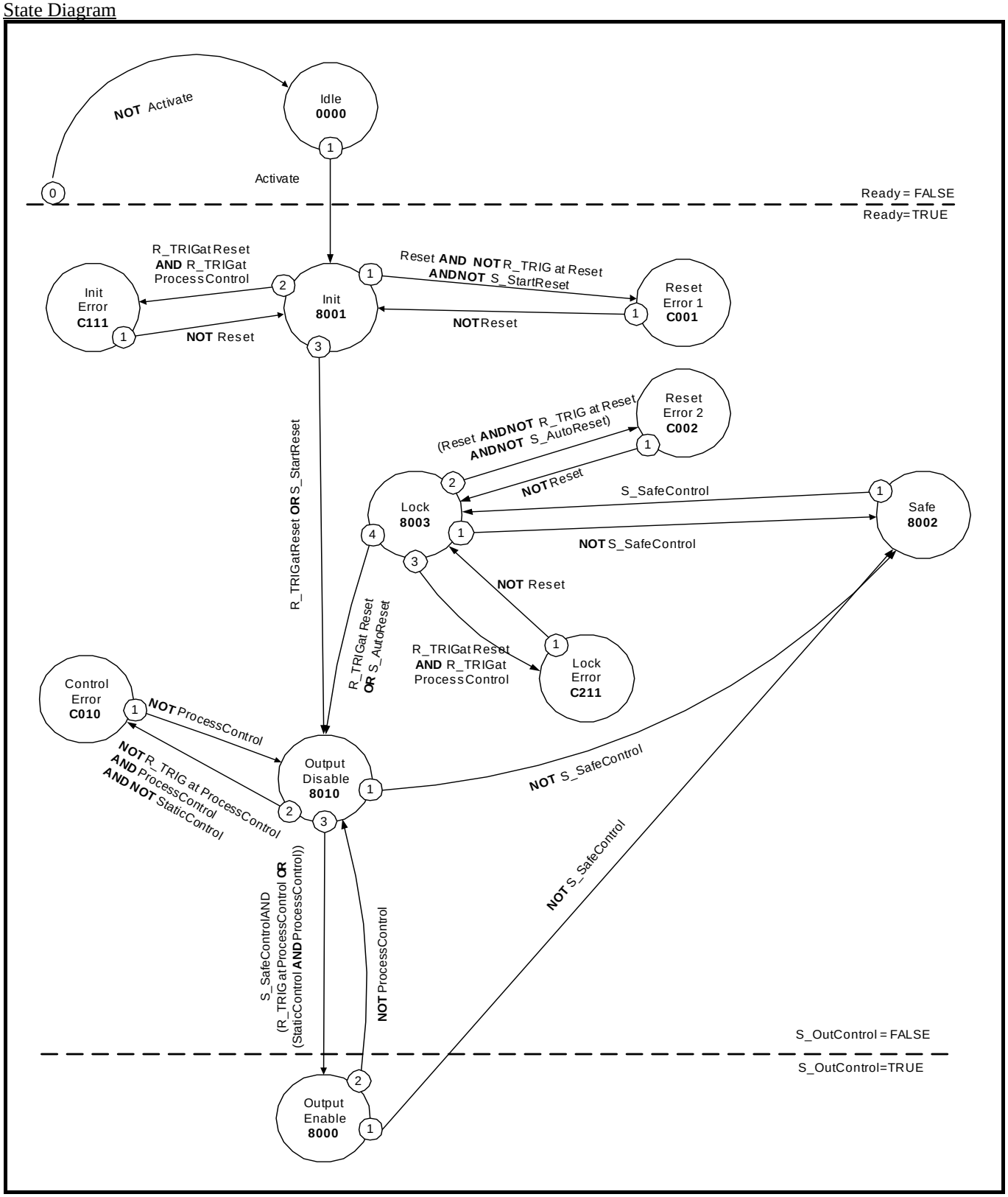
- An additional function start (ProcessControl FALSE => TRUE) is required following block activation or feedback of the safe signal (S_SafeControl). A static TRUE signal at ProcessControl does **not** set S_OutControl to TRUE.
- An additional function start (ProcessControl FALSE => TRUE) is **not** required following block activation or feedback of the safe signal (S_SafeControl). A static TRUE signal at ProcessControl sets S_OutControl to TRUE if the other conditions have been met.

Optional startup inhibits:

- Startup inhibit after function block activation.
- Startup inhibit after interruption of the protective device.

The StaticControl, S_StartReset and S_AutoReset inputs shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

State Diagram

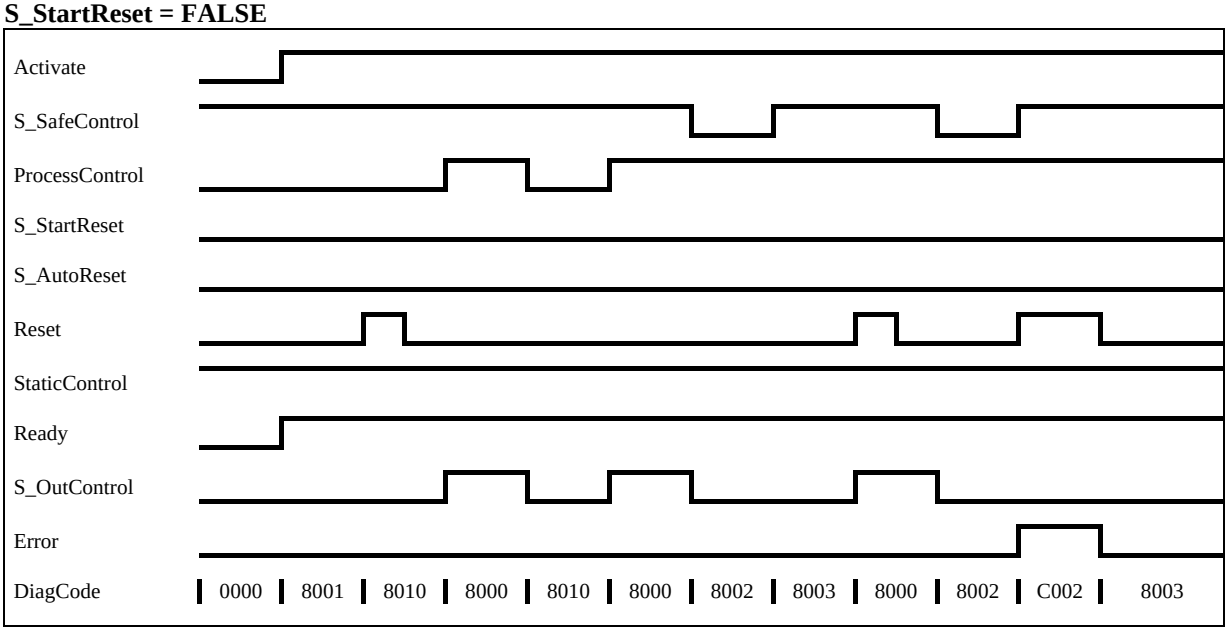


Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 57: State diagram for SF_OutControl

Typical Timing Diagrams

S_StartReset = FALSE



S_StartReset = TRUE

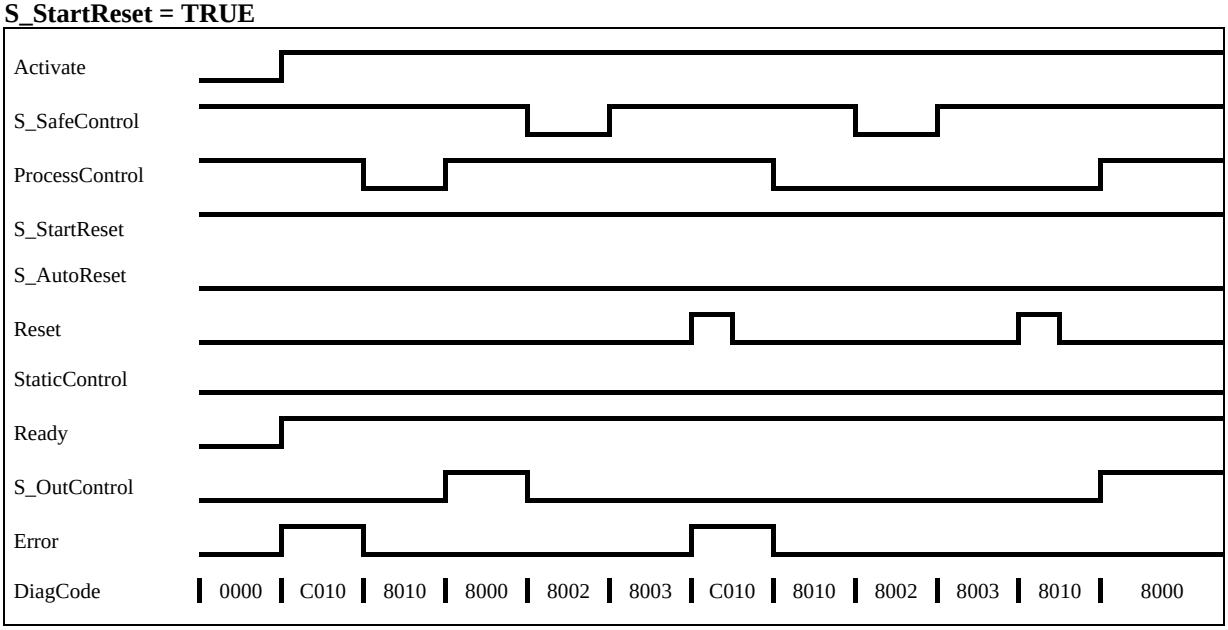


Figure 58: Timing diagram for SF_OutControl

6.19.4. Error Detection

The following conditions force a transition to the Error state:

- Invalid static Reset signal in the process.
- Invalid static ProcessControl signal.
- ProcessControl and Reset are incorrectly interconnected due to programming error.

6.19.5. Error Behavior

In the event of an error, the S_OutControl output is set to FALSE and remains in this safe state.
To leave the Reset, Init or Lock error states, the Reset input must be set to FALSE. To leave the Control error state, the ProcessControl input must be set to FALSE.
After transition of S_SafeControl to TRUE, the optional startup inhibit can be reset by a rising edge at the Reset input.
After block activation, the optional startup inhibit can be reset by a rising edge at the Reset input.

6.19.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	Output Setting
FB-specific error codes:		
C001	Reset Error 1	Static Reset signal in state 8001. Ready = TRUE S_OutControl = FALSE Error = TRUE
C002	Reset Error 2	Static Reset signal in state 8003. Ready = TRUE S_OutControl = FALSE Error = TRUE
C010	Control Error	Static signal at ProcessControl in state 8010. Ready = TRUE S_OutControl = FALSE Error = TRUE
C111	Init Error	Simultaneous rising trigger at Reset and ProcessControl in state 8001. Ready = TRUE S_OutControl = FALSE Error = TRUE
C211	Lock Error	Simultaneous rising trigger at Reset and ProcessControl in state 8003. Ready = TRUE S_OutControl = FALSE Error = TRUE
FB-specific status codes (no error):		
0000	Idle	The function block is not active (initial state). Ready = FALSE S_OutControl = FALSE Error = FALSE
8001	Init	Block activation startup inhibit is active. Reset required. Ready = TRUE S_OutControl = FALSE Error = FALSE
8002	Safe	Triggered safety function. Ready = TRUE S_OutControl = FALSE Error = FALSE
8003	Lock	Safety function startup inhibit is active. Reset required. Ready = TRUE S_OutControl = FALSE Error = FALSE
8010	Output Disable	Process control is not active. Ready = TRUE S_OutControl = FALSE Error = FALSE
8000	Output Enable	Process control is active and safety is enabled. Ready = TRUE S_OutControl = TRUE Error = FALSE

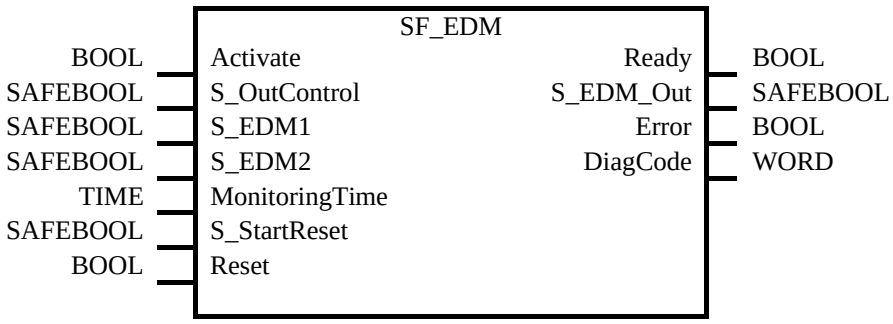
6.20. External Device Monitoring

6.20.1. Applicable Safety Standards

Standards	**Requirements**
IEC 60204-1, Ed. 5.0: 2003	Section 9.2.2: Stop function categories; Category 0
EN 954-1: 1996	5.2: Stop function; stop initiated by protective devices shall put the machine in a safe state 6.2: Specification of categories: Fault detection (of the actuator, e.g. open circuits)
ISO 12100-2: 2003	4.11.4: Restart following power failure/spontaneous restart
EN 954-1: 1996	5.4 Manual reset

6.20.2. Interface Description

FB Name	**SF_EDM**		
External device monitoring – The FB controls a safety output and monitors controlled actuators, e.g. subsequent contactors			
VAR_INPUT			
Name	*Data Type*	*Initial Value*	*Description, Parameter Values*
Activate	BOOL	FALSE	See Section 5.1.1 General Input Parameters
S_OutControl	SAFEBOOL	FALSE	Variable. Control signal of the preceeding safety FB's. Typical function block signals from the library (e.g., SF_OutControl, SF_TwoHandControlTypeII, and/or others). FALSE: Disable safety output (S_EDM_Out). TRUE: Enable safety output (S_EDM_Out).
S_EDM1	SAFEBOOL	FALSE	Variable. Feedback signal of the first connected actuator. FALSE: Switching state of the first connected actuator. TRUE: Initial state of the first connected actuator.
S_EDM2	SAFEBOOL	FALSE	Variable. Feedback signal of the second connected actuator. If using only one signal in the application, the user must use a graphic connection to jumper the S_EDM1 and S_EDM2 parameters. S_EDM1 and S_EDM2 are then controlled by the same signal. FALSE: Switching state of the second connected actuator. TRUE: Initial state of the second connected actuator.
MonitoringTime	TIME	#0ms	Constant. Max. response time of the connected and monitored actuators.
S_StartReset	SAFEBOOL	FALSE	See Section 5.1.1 General Input Parameters
Reset	BOOL	FALSE	See Section 5.1.1 General Input Parameters
VAR_OUTPUT			
Ready	BOOL	FALSE	See Section 5.1.2 General Output Parameters
S_EDM_Out	SAFEBOOL	FALSE	Controls the actuator. The result is monitored by the feedback signal S_EDMx. FALSE: Disable connected actuators. TRUE: Enable connected actuators.
Error	BOOL	FALSE	See Section 5.1.2 General Output Parameters
DiagCode	WORD	16#0000	See Section 5.1.2 General Output Parameters
Notes: -			



6.20.3. Functional Description

General:

The SF_EDM FB controls a safety output and monitors controlled actuators.

This function block monitors the initial state of the actuators via the feedback signals (S_EDM1 and S_EDM2) before the actuators are enabled by the FB.

The function block monitors the switching state of the actuators (MonitoringTime) after the actuators have been enabled by the FB.

Two single feedback signals must be used for an exact diagnosis of the connected actuators. A common feedback signal from the two connected actuators must be used for a restricted yet simple diagnostic function of the connected actuators. When doing so, the user must connect this common signal to both parameter S_EDM1 and parameter S_EDM2. S_EDM1 and S_EDM2 are then controlled by the same signal.

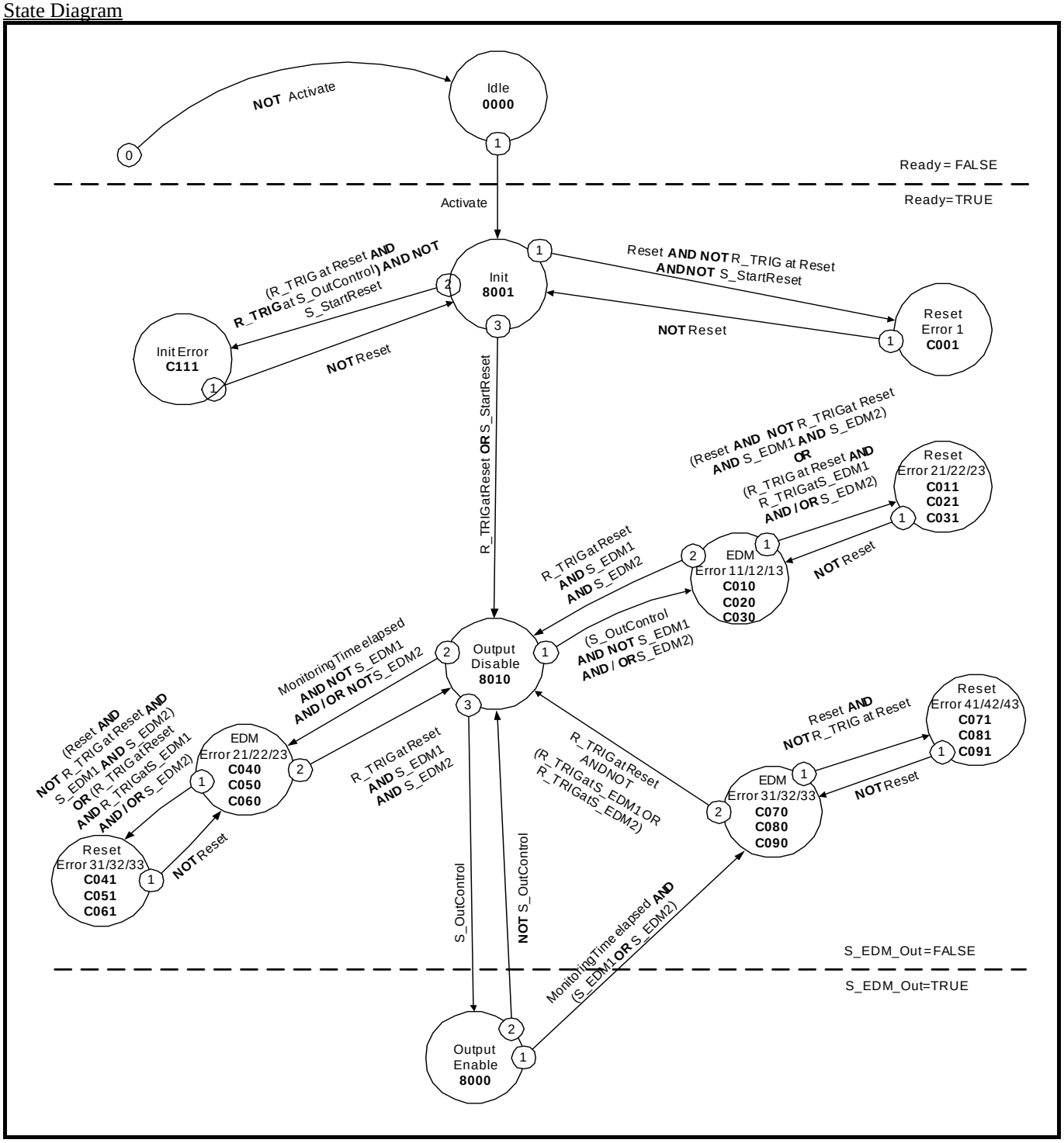
The switching devices used in the safety function should be selected from the category specified in the risk analysis (EN 954-1).

Optional startup inhibits:

- Startup inhibit in the event of block activation.

The S_StartReset input shall only be activated if it is ensured that no hazardous situation can occur when the PES is started.

State Diagram

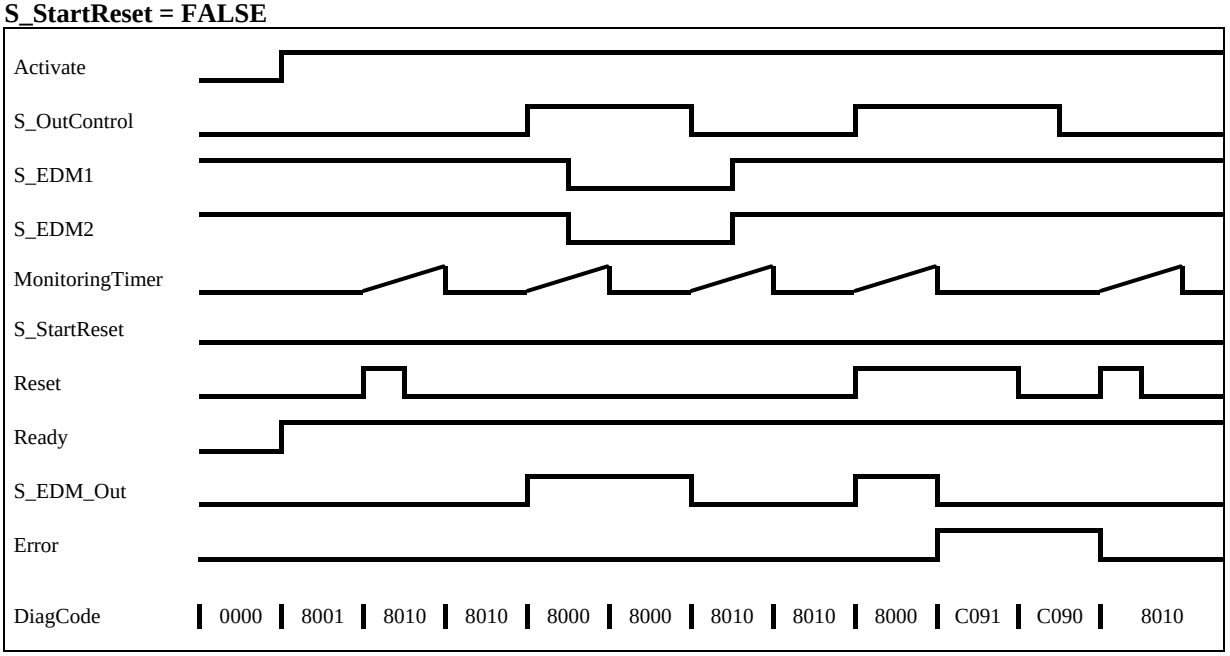


Note: The transition from any state to the Idle state due to Activate = FALSE is not shown. However these transitions have the highest priority (0).

Figure 59: State diagram for SF_EDM

Typical Timing Diagrams

S_StartReset = FALSE



S_StartReset = TRUE

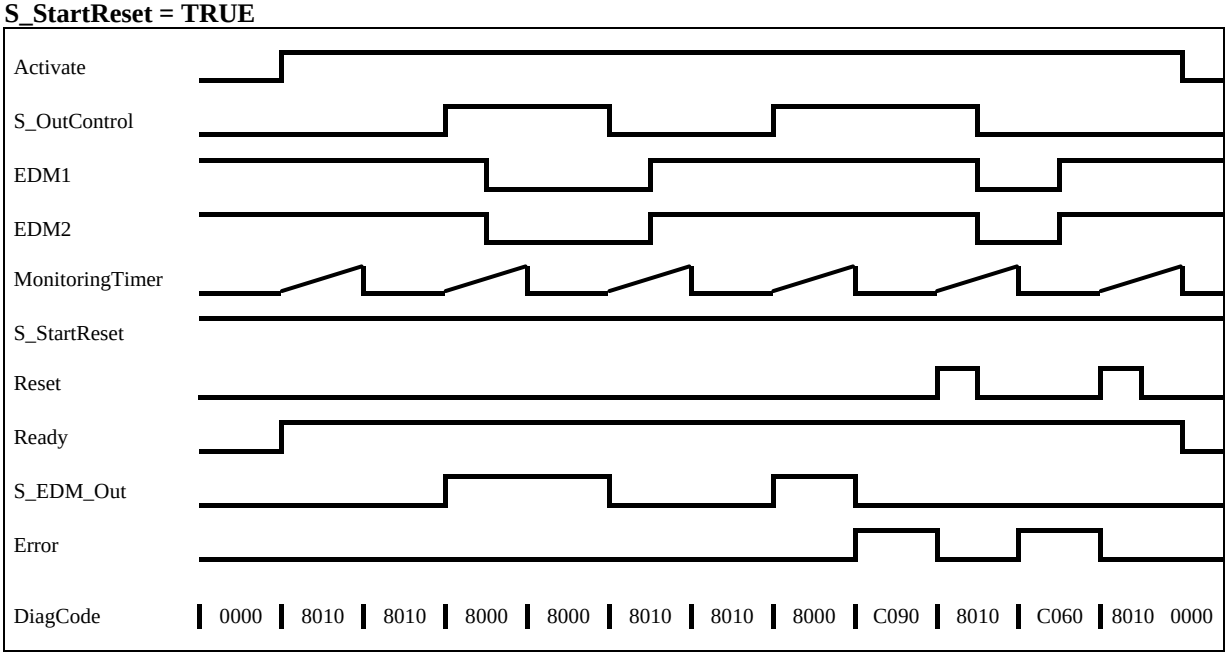


Figure 60: Timing diagrams for SF_EDM

6.20.4. Error Detection

The following conditions force a transition to the Error state:

- Invalid static Reset signal in the process.
- Invalid EDM signal in the process.
- S_OutControl and Reset are incorrectly interconnected due to programming error.

6.20.5. Error Behavior

In error states, the outputs are as follows:

- In the event of an error, the S_EDM_Out is set to FALSE and remains in this safe state.
- An EDM error message must always be reset by a rising trigger at Reset.
- A Reset error message can be reset by setting Reset to FALSE.

After block activation, the optional startup inhibit can be reset by a rising edge at the Reset input.

6.20.6. Function Block-Specific Error and Status Codes

DiagCode	State Name	Output Setting
FB-specific error codes:		
C001	Reset Error 1	Static Reset signal in state 8001. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C011	Reset Error 21	Static Reset signal or same signals at EDM1 and Reset (rising trigger at Reset and EDM1 at the same time) in state C010. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C021	Reset Error 22	Static Reset signal or same signals at EDM2 and Reset (rising trigger at Reset and EDM2 at the same time) in state C020. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C031	Reset Error 23	Static Reset signal or same signals at EDM1, EDM2, and Reset (rising trigger at Reset, EDM1, and EDM2 at the same time) in state C030. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C041	Reset Error 31	Static Reset signal or same signals at EDM1 and Reset (rising trigger at Reset and EDM1 at the same time) in state C040. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C051	Reset Error 32	Static Reset signal or same signals at EDM2 and Reset (rising trigger at Reset and EDM2 at the same time) in state C050. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C061	Reset Error 33	Static Reset signal or same signals at EDM1, EDM2, and Reset (rising trigger at Reset, EDM1, and EDM2 at the same time) in state C060. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C071	Reset Error 41	Static Reset signal in state C070. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C081	Reset Error 42	Static Reset signal in state C080. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C091	Reset Error 43	Static Reset signal in state C090. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C010	EDM Error 11	The signal at EDM1 is not valid in the initial actuator state. In state 8010 the EDM1 signal is FALSE when enabling O_OutControl. Ready = TRUE S_EDM_Out = FALSE Error = TRUE

C020	EDM Error 12	The signal at EDM2 is not valid in the initial actuator state. In state 8010 the EDM2 signal is FALSE when enabling O_OutControl. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C030	EDM Error 13	The signals at EDM1 and EDM2 are not valid in the initial actuator states. In state 8010 the EDM1 and EDM2 signals are FALSE when enabling O_OutControl. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C040	EDM Error 21	The signal at EDM1 is not valid in the initial actuator state. In state 8010 the EDM1 signal is FALSE and the monitoring time has elapsed. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C050	EDM Error 22	The signal at EDM2 is not valid in the initial actuator state. In state 8010 the EDM2 signal is FALSE and the monitoring time has elapsed. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C060	EDM Error 23	The signals at EDM1 and EDM2 are not valid in the initial actuator states. In state 8010 the EDM1 and EDM2 signals are FALSE and the monitoring time has elapsed. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C070	EDM Error 31	The signal at EDM1 is not valid in the actuator switching state. In state 8000 the EDM1 signal is TRUE and the monitoring time has elapsed. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C080	EDM Error 32	The signal at EDM2 is not valid in the actuator switching state. In state 8000 the EDM2 signal is TRUE and the monitoring time has elapsed. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C090	EDM Error 33	The signals at EDM1 and EDM2 are not valid in the actuator switching state. In state 8000 the EDM1 and EDM2 signals are TRUE and the monitoring time has elapsed. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
C111	Init Error	Similar signals at S_OutControl and Reset (R_TRIG at same cycle) detected (may be a programming error) Ready = TRUE S_EDM_Out = FALSE Error = TRUE

FB-specific status codes (no error):

0000	Idle	The function block is not active (initial state). Ready = FALSE S_EDM_Out = FALSE Error = FALSE

8001	Init	Block activation startup inhibit is active. Reset required. Ready = TRUE S_EDM_Out = FALSE Error = FALSE
8010	Output Disable	EDM control is not active. Timer starts when state is entered Ready = TRUE S_EDM_Out = FALSE Error = FALSE
8000	Output Enable	EDM control is active. Timer starts when state is entered Ready = TRUE S_EDM_Out = TRUE Error = FALSE

Appendix 1. Compliance Procedure and Compliance List

Listed in this Appendix are the requirements for the compliance statement from the supplier of the safety specification. The compliance statement consists of two main groups:

1. Reduction in programming languages and functionality (see "Appendix 1.2 Applicable reductions in the Development Environment").
2. The definition of a set of function blocks with safety-related functionality (see "Appendix 1.3 Overview of the supported Function Blocks").

The supplier must fill out the tables for their implementation, according to their product, committing their support to the specification itself.

By submitting these tables to PLCopen, and following approval by PLCopen, the list will be published on the PLCopen website (http://www.plcopen.org) as specified in "Appendix 2 The PLCopen Safety Logo and Its Use" below.

In addition to this approval, the supplier is provided with access and usage rights for the PLCopen Safety logo, as described in Appendix 2 The PLCopen Safety Logo and Its Use.

Appendix 1.1. Supplier Statement

Supplier name	
Supplier address	
City	
Country	
Phone	
Fax	
Website	
Product name	
Product version	
Release date	
Certified by	

I hereby state that the following tables as filled out and submitted correspond to our product and the accompanying user manual, as stated above.

Name of representative:

Date of signature (dd/mm/yyyy):

Signature:

Appendix 1.2. Applicable reductions in the Development Environment

Supported User Levels (See Section 4)	**Supported**	**Comments (< 48 Characters)**
Basic level		
Extended level		
System level		How is it supported?

Table 8: Supported user levels

Supported Programming Languages	**Supported**	**Comments (< 48 Characters)**
Function Block Diagram, FBD		
Ladder Diagram, LD		

Table 9: Supported programming languages

Supported Data Types	**Supported**	**Comments (< 48 Characters)**
SAFEBOOL		
BOOL		
INT		
DINT		
REAL		
WORD		
TIME		

Table 10: Supported data types

Supported Functions and FBs – Basic Level	**Supported**	**Comments (< 48 Words)**
AND		
OR		
Type Conversion functions		Specifiy which
TON		
TOF		
TP		
CTU		
CTD		
CTUD		
Others?		Specifiy which

Table 11: Supported Functions and Function Blocks at Basic Level

Supported Functions and FBs – Extended Level	Supported	Comments (< 48 Words)
AND		
OR		
XOR		
NOT		
ADD		
MUL		
SUB		
DIV		
GT, GE, EQ, LE, LT, NE		Specify which
Selection functions		Specify which
Type conversion functions		Specify which
Time functions		Specify which
TON		
TOF		
TP		
CTU		
CTD		
CTUD		
Bistable FBs		Specify which
Edge detection		Specify which
Others?		Specify which

Table 12: Supported Functions and Function Blocks at Extended Level

Appendix 1.3. Overview of the supported Function Blocks

Function Blocks	Supported	Comments (<= 48 Characters)
SF_Equivalent		
SF_Antivalent		
SF_ModeSelector		
SF_EmergencyStop		
SF_ESPE		
SF_SafeStop1		
SF_SafeStop2		
SF_SafetyGuardMonitoring		
SF_SafelyLimitedSpeed		
SF_TwoHandControlTypeII		
SF_TwoHandControlTypeIII		
SF_GuardLocking		
SF_TestableSafetySensor		
SF_MutingSeq		
SF_MutingPar		
SF_MutingPar_2Sensors		
SF_EnableSwitch		
SF_SafetyRequest		
SF_OutControl		
SF_EDM		

Table 13: Overview of the function blocks

Appendix 2. The PLCopen Safety Logo and Its Use

For quick identification of compliant products, PLCopen has developed a logo for the Safety Specification:

Figure 61: The PLCopen Safety logo

This logo is owned and trademarked by PLCopen.

In order to use this logo free of charge, the relevant company must meet all of the following requirements:

1. The company must be a voting member of PLCopen;
2. The company must comply with the existing specification, as specified by the PLCopen Technical Committee 5 - Safety, and as published by PLCopen, and of which this statement is a part;
3. This compliance is submitted in writing by the company to PLCopen, clearly stating the applicable software package and the supporting elements of all the specified tables, as specified in this document;
4. The company is aware that this compliance is only a statement of the supporting elements as specified in this document. In particular, the company is aware that this statement does not have any relationship to the implementation itself, nor the fulfillment of any requirements as specified in any safety standard, safety procedure, or development procedure, and does not state anything with regard to the quality of the product itself, nor certification procedures performed by a third party;
5. In the event of non-fulfillment, which must be decided by PLCopen, the company will receive a written statement to this effect from PLCopen. The company will have a period of one month to either adapt their software package in such a way that it is compliant, i.e., by issuing a new compliance statement, or removal of all reference to the specification, including the use of the logo, from all their specifications, be they technical or promotional material;
6. The logo must be used as is - i.e., in its entirety. It may only be altered in size as long as the original scale and color settings are maintained;
7. The logo must be used in the context of PLCopen Safety.