PLCopen

for efficiency in automation

PLCopen - Technical Committee 5
Safety Software

Technical Specification

Part 2: User Examples

Version 1.01 – Official Release

DISCLAIMER OF WARANTIES

THIS DOCUMENT IS PROVIDED ON AN “AS IS” BASIS AND MAY BE SUBJECT TO FUTURE ADDITIONS, MODIFICATIONS, OR CORRECTIONS. PLCOPEN HEREBY DISCLAIMS ALL WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, FOR THIS DOCUMENT. IN NO EVENT WILL PLCOPEN BE RESPONSIBLE FOR ANY LOSS OR DAMAGE ARISING OUT OR RESULTING FROM ANY DEFECT, ERROR OR OMISSION IN THIS DOCUMENT OR FROM ANYONE’S USE OF OR RELIANCE ON THIS DOCUMENT.

Date: February 26, 2008 / July 7, 2008

Total number of pages: 42

User Examples with PLCopen Safety Functions

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

Leo Schratt	3S Smart Software Solutions, Kempten, Germany
Joachim Greis	Beckhoff, Verl, Germany
Franz Kaufleitner	B&R, Eggelsberg, Austria
Michael Huelke	BGIA, Sankt Augustin, Germany
Jochen Ost	Bosch Rexroth, Lohr, Germany
Michael Mühlbauer	Bosch Rexroth, Lohr, Germany
Thomas Janzer	HIMA, Brühl, Germany
Hans Russo	Infoteam Software, Bubenreuth, Germany
Marko Lachmann	Infoteam Software, Bubenreuth, Germany
Andreas Otto	KW Software GmbH, Lemgo, Germany
Frank Bauder	Omron Europe, Nufringen, Germany
Olaf Ruth	Phoenix Contact, Blomberg, Germany
Michael Pistorius	Phoenix Contact, Blomberg, Germany
Johannes Kalhoff	Phoenix Contact, Blomberg, Germany
Armin Wenigenrath	Schneider Electric, Seligenstadt, Germany
Andreas Höll	Sick, Waldkirch, Germany
Bernard Mysliwiec	Siemens, Nuremberg, Germany
Martin Gottwald	Siemens, Nuremberg, Germany
Erich Janoschek	TÜV Rheinland, Cologne, Germany
Eelco van der Wal	PLCopen, Gorinchem, Netherlands

Change Status List:

Version Number	**Date**	**Change Comment**
V 0.1	January 20, 2006	First draft generated by EvdWal based on existing docs
V 0.2	May 31, 2006	As results of meeting at Kempten
V 0.3	August 18, 2006	As result of meeting at Lemgo
V 0.4	October 18, 2006	As result of meeting at Waldkirch
V 0.5	December 21, 2006	All feedback merged into this version
V 0.6	January 10, 2007	As result of the meeting in Bruehl
V 0.7	February 28, 2007	As result of work done by M. Huelke and E. van der Wal on Ch. 2
V 0.8	May 10, 2007	As result of the meeting in Bubenreuth
V 0.9	June 14, 2007	As result of the meeting at Neuremberg and editing by EvdW
V 0.99	February 5, 2008	As result of feedback of group and provided solutions via email
V 1.0	February 26, 2008	Last changes accepted by Chairman, and document released
V 1.01	July 7, 2008	Missing information in Figure 12 in the pdf file. New pdf version generated

Table of Contents

1 Introduction

In 2006 the members of the PLCopen Technical Committee 5 - Safety released the safety specification Part 1 – Concepts and Functions Blocks. With this Part 1 and the corresponding corrigendum, the safety aspects can be transferred to a software tool, which is integrated in the software development tools. This helps developers and users 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.

Since this Safety Part 1 is more focused to the implementers of the functionalities, PLCopen realized it needed to do more to help OEMs to add the safety functionality to their machines' application software. For this reason the PLCopen TC5 Safety group continued with a Part 2 – User Guidelines.

This document Part 2 is focused to the users of the PLCopen functionalities and demonstrates the ease of use of the defined function blocks in real life applications. The PLCopen safety functionalities are focused to the machine building industry, as well as related system integrators.

The combination of logic, motion, and safety in one environment provides the user with a harmonized view of the total application within one environment. And with multiple implementations, this is also valid across platforms. This means less educational efforts, and simpler transfer of knowledge and application software between different controls. Also, it tackles the 'not-invented-here syndrome', which often is a cause of errors and additional costs. By using tested functionality, and support in the programming environment, including language definition with subsets of functionality, one is able to create safety related application programs for easy commissioning.
With this, a major contribution to the acceptance and usage of safety related functionality is made. This will take away several hurdles as they now exist, especially for the machine building industry.

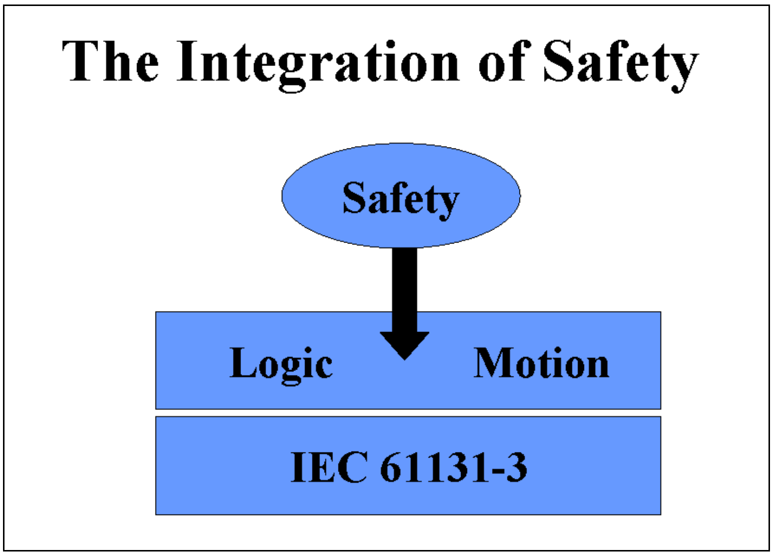


Figure 1: Merging three environments on one platform

The examples in Chapter 4 - Application Examples show possible, but simplified application programs of typical safety functions. However, the selection of the right safety hardware components, the wiring and coupling to the safety software function blocks in accordance with the risk analysis is required. Furthermore, the user has the responsibility to take additional measures to comply with the safety requirements.

2 General overview

This chapter gives a short overview of the safety functionalities as defined in the document "PLCopen - Technical Committee 5 – Safety Software - Technical Specification Part 1: Concepts and Function Blocks", as first officially released in 2006. For a complete specification of the representation of these functionalities in the function blocks, as well as the concepts for the developments environments check Part 1 and the corresponding corrigendum. The following overview is provided to ease the understanding of the examples.

2.1. Creating a Safety Plan

The safety plan is the bracket for all activities required for the realization of a safety function (SF) by a programmable electronic system (PES) on a machine.

Objective of the step
The Machine Directive requires a systematic procedure when realizing a SF by a PES. This includes the documentation of all activities in the safety plan:

- from the hazard analysis and risk assessment of the machine
- and the design and realization of the SF
- to the validation.

The safety plan always has to be updated with each step of the realization of the SF.

Procedure
The following topics and activities are documented in the safety plan.

- Planning and procedure of all activities required for the realization of a SF. Examples:
  - Developing the specification of the safety-related control function.
  - Designing and integrating the PES

Validating the SF with the PES

  - Preparing the PES user documentation
  - Documentation of all relevant information on the realization of the SF (project documentation)

- Strategy how the functional safety is to be achieved.
- Responsibilities for execution and review of all activities

2.2. Terms and Definitions

AOPD	Active opto-electronic protective device
EDM	External device monitoring signal, which reflects the state transition of an actuator.
ESPE	Electro-sensitive protective equipment (for example a light curtain)
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.
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
PES	Programmable Electronic System (see IEC 61508)
PLC	Programmable Logic Controller
Process control	Control signal from the functional application for process control.
SAFEBOOL	Data type to identify safety-related BOOLEAN signals.
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 function	Function of the machine which failure can result in an immediate increase of the risk(s) [ISO 12100-1:2003, clause 3.28]
Stop category	Machine stop in accordance with stop category 0, 1, 2 (cf. IEC 60204).

2.3. *Example of safety functionalities in a production line*

As an example of the usage of safety functions, the following production line is used. It is state-of-the-art to use a programmable electronic system (PES) to realize the safety functions (versus hard-wired). The PLCopen FBs described below can be used to easily realize the safety application program.

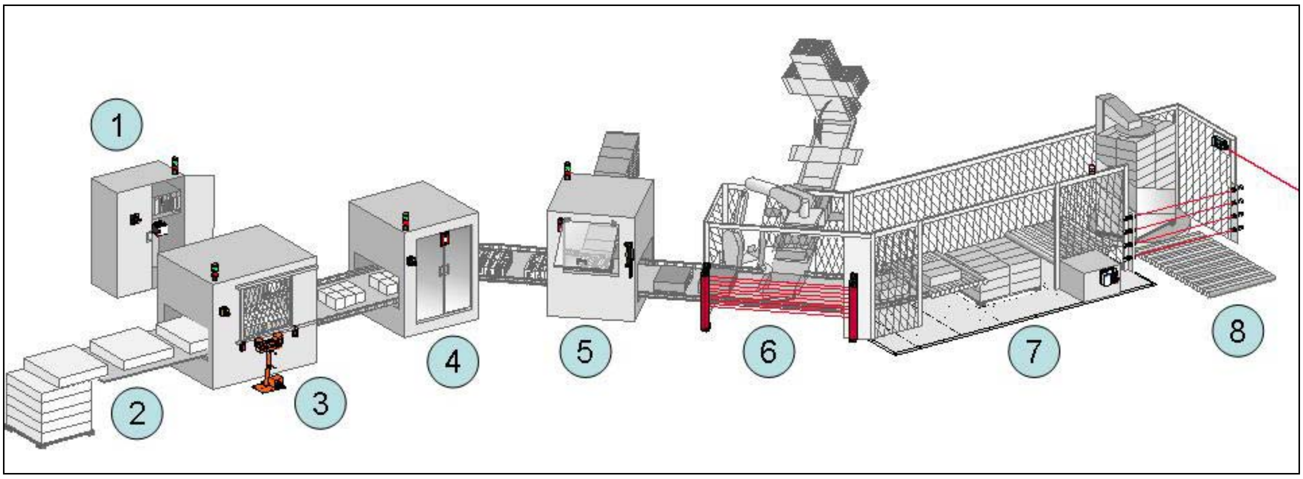


Figure 2: Example of safety functionalities in a production line

This production line consists of the following functionalities:

1. Centralized control cabinet, including the safety related part of the control system where the safety related function blocks are running.
2. Infeed of material. In this example no special safety related functions are used. However, safety functionalities like muting to separate between products and persons could be used.
3. Cutting of the material. For manual control a two hand control safety function (unit is in front of the machine) is added combined with a 2-fold door monitoring system (attached to the door on the machine)
4. Automatic printing station, with door monitoring as safety function in case of service access (attached to door on the machine)
5. First cartonning machine with door monitoring as safety function in case of service access (attached to door on the machine). Sometimes manual operation is necessary. In this case the operator can run the machine with a safely limited speed controlled by an enabling device which, when released, initiates a safe stop.
6. Second cartonning machine, guarded by an electro-sensitive protective equipment, ESPE. In this case a light curtain.
7. Palletizing function, guarded by safety mats. This functionality could be coupled to the ESPE safety function.
8. Foil wrapping station of the palletized products, as well as exit of the production line. This area is safeguarded by several combined light beams, coupled to the ESPE safety function.

In addition, every station is equipped with an emergency stop.

2.4. Description of the PLCopen Function Blocks

SF_ModeSelector
The SF_ModeSelector handles a mode selector switch with up to 8 positions to distinguish different safety relevant operation modes. The FB basically monitors the switch in that way, that only one mode gets pre-selected. If for any reason the position signal is not definite (e.g. two SAFEBOOL input bits or more are HIGH for a longer time than the MonitoringTime), the output gets set to the fail-safe state value (no mode selected). Depending on the control bits, the output signals get activated either immediately or has to be confirmed by an additional input first.
Note: Most applications differentiate between two safety relevant operation modes only. Either the machine is in automatic mode or the machine is in a mode which requires additional functional safety.
Normally, this FB is followed by additional function blocks.

SF_EmergencyStop
This function block handles the emergency stop condition for the application, and ensures a restart inhibit.
The machine cannot be restarted until the emergency button is released and a reset signal is given by the user.
Normally, the output of this FB is not directly coupled to actuators but followed by additional function blocks, e.g. to initiate a machine stop in accordance with stop category 0, 1, 2.

SF_SafeStop1
This function block initiates and monitors a stop of a drive system in accordance with stop category 1.
If for any reason the drive does not confirm the requested safety function (e.g. monitoring time elapses), the FB output gets set to fail-state values and the error bit gets set. It can be assumed that the drive system performs an independent fail-safe reaction, since the communication to the drive is interrupted or a drive internal error occurred.
The FB can be restarted after the problem got fixed and a reset signal initiated.
Normally, this FB does not get linked directly to the processing input level. Mostly the FB follows the SF_EmergencyStop FB.
Note: The FB does not execute the stop function. The function itself needs to be encapsulated within the drive system. The signals to initiate the stop or to get the acknowledge signal back will be exchanged via a hidden system level interface. For details see 3.3 Information for the use of safe drives (with a hidden system level interface).

SF_SafeStop2
This function block initiates and monitors a stop of a drive system in accordance with stop category 2.
If for any reason the drive does not confirm the requested safety function (e.g. monitoring time elapses), the FB output gets set to fail-state values and the error bit gets set. It can be assumed that the drive system performs an independent fail-safe reaction, since the communication to the drive is interrupted or a drive internal error occurred.
The FB can be restarted after the problem got fixed and a reset signal initiated.
Normally, this FB does not get linked directly to the processing input level. Mostly the FB follows the SF_EmergencyStop FB.
Note: The FB does not execute the stop function. The function itself needs to be encapsulated within in the drive system. The signals to initiate the stop or to get the acknowledge signal back will be exchanged via a hidden system level interface. For details see 3.3 Information for the use of safe drives (with a hidden system level interface).

SF_ESPE
This function blocks handles the output signal switching device (OSSD) of an ESPE (Electro-sensitive protective equipment, e.g. light curtain) for the application, and ensures a restart inhibit.
The machine cannot be restarted until the ESPE is infringed and a reset signal is given by the user.
Normally, the output of this FB is not directly coupled to actuators but followed by additional function blocks, e.g. to initiate a machine stop in accordance with stop category 0, 1, 2.

SF_TestableSafetySensor
This function block can be used for a periodic testing of a safety sensor (e.g. type 2 ESPE).
By use with an ESPE, the function block simulates an intrusion in the protected field of the ESPE and monitors the maximum response time. Within this test, a detection of a hazardous fault is possible. During test the output of the block is muted. If no test is active and output signal of safety sensor switched off, also output of function block is switched off immediately. Reset is used only for error reset (e.g. exceeding of monitoring time).
Normally the output of this FB is not directly coupled to actuators but followed by additional function blocks SF_ESPE, e.g. to ensure the restart inhibit.

SF_MutingSeq, SF_MutingPar, and SF_MutingPar_2Sensor

Muting is the intended suppression of the safety function of e.g. a light curtain. This is required, e.g. when transporting material into the danger zone, guarded by a light curtain, without causing the machine to stop. The muting sequence is triggered by muting sensors. The correct installation of these sensors into the production sequence must ensure that no person can initiate the muting function. Active muting mode must be indicated by indicator lights.
Three function blocks are available for different muting sequences, depending on the arrangement and the amount of the used muting sensors.

- SF_MutingSeq intended for sequential muting with four muting sensors.
- SF_MutingPar intended for parallel muting with four muting sensors.
- SF_MutingPar_2Sensor intended for parallel muting with two muting sensors.

Each function block monitors the correct activation of the muting sensors, according to the used muting mode and resets the enable signal in case of:

- muting sequence is not initiated in the correct way
- light curtain is interrupted while not muted
- defect muting lamp

SF_GuardMonitoring

This function block monitors the relevant interlocking guard. There are two independent inputs for two switches at the safety guard coupled with a time difference (via the input MonitoringTime) for closing the 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.

SF_GuardLocking

In order to secure a hazardous area, it can be fenced. In order to enter this area, a door is included. This door is monitored and protected by a lock, which can be a mechanical locked switch. In normal operation, the door is closed and locked.
A person who wants to enter the hazardous area has to request this. The door (or guard) can only be unlocked and opened if 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 by the safety functionality.

SF_TwoHandControlTypeII and SF_TwoHandControlTypeIII

These function blocks provide the two-hand control functionality, and start the machine to operate as long the buttons are pressed (set the relevant output) under certain conditions.
Type II checks if the switching of the inputs (the result of pushing the knobs) is in the correct sequence.
Type III has additional timing constraints for setting the two knobs. The maximum timing difference allowed is 500 msec.

SF_SafelyLimitedSpeed

This function block initiates and monitors an actor system to perform either a SafeStop2 or to get enabled by an enabling switch to move slower than the permitted safely limited speed. The functions can be activated only if the normal operation mode is <u>not</u> activated. In the other case no functional safety function is active and has to be ensured by other measures, like a guard.
If for any reason the drive does not confirm the requested safety function (e.g. monitoring time elapses), the FB output gets set to fail-state values and the error bit gets set. It can be assumed that the drive system performs an independent fail-safe reaction, since the communication to the drive is interrupted or a drive internal error occurred.
The FB can be restarted after the problem got fixed and a reset signal initiated.
Normally, this FB does <u>not</u> get linked directly to the processing input level. Mostly the FB follows the SF_EnableSwitch, SF_DoorMonitoring, and SF_ModeSelector FB.
<u>Note</u>: The FB does not execute this function. The function itself needs to be encapsulated within in the drive system. The signals to initiate the function or to get the acknowledge signal back will be exchanged via a hidden system level interface. For details see 3.3 Information for the use of safe drives (with a hidden system level interface).
<u>Note:</u> The command values are generated by the standard motion controller and require an additional start signal.

SF_SafeRequest

This FB is a generic FB and is used with generic actors which could perform safety functions on request.
The FB does not utilize a hidden system level interface. Furthermore the communication to the actor like a drive is done via an IO interface, either discrete or via a fieldbus. The requested safe mode gets acknowledged by a feedback signal.

If for any reason the drive does not confirm the requested safety function (e.g. monitoring time elapses), the FB output gets set to fail-state values and the error bit gets set. It can be assumed that the actor performs an independent fail-safe reaction, since the communication to the actor is interrupted or an actor internal error occurred.
The FB can be restarted after the problem got fixed and a reset signal initiated.
Normally, this FB does not get linked directly to the processing input level. Mostly the FB follows the SF_EnableSwitch, SF_DoorMonitoring, and SF_ModeSelector.

SF_EnableSwitch

This function block handles a 3 position enable switch. When activated, that means position 2 is detected, the FB allows a machine operation to be initiated by a separate start control. When de-activated, that means position 1 or 3 is detected, the FB initiates a stop function, and prevents initiation of machine operation.
The FB prevents that a machine operation could be enabled before a safe condition (e.g. safe stop) is active. In order to prevent an unexpected start, the FB ensures that the enable switch is not activated when the safe mode becomes active.
If for some reason the machine is not in a safe mode anymore (safety active acknowledge signal gets false), the (enabling of the) operation will be disabled.
This FB is dedicated to be used with an SF_SafelyReducedSpeed (or similar functionalities).
Note: The FB can be used only for 3 position enable switches which perform a specific signal behaviour, like shown in the table. For details see 3.1 PLCopen FBs and their Connection to the Periphery.

SF_OutControl

This FB controls a safety output with a signal coming from the functional application, which is enabled by a safety signal. This FB has an optional startup inhibit.

SF_EDM

This FB controls a safety output and monitors controlled actuators, e.g. subsequent contactors.
This FB monitors the initial state of the actuators via the feedback signals (S_EDM1 and S_EDM2) before the actuators are enabled by the FB. It monitors the switching state of the actuators (MonitoringTime) after the actuators have been enabled by the FB.
Two single feedback signals may be used for an exact diagnosis of the connected actuators. A single feedback signal must be provided to both inputs. When doing so, the user must connect this single signal to both parameter S_EDM1 and parameter S_EDM2. S_EDM1 and S_EDM2 are then controlled by the same signal.

3 General notes

3.1. PLCopen FBs and their Connection to the Periphery

Within the PLCopen examples the safety input devices and the connection of the peripheral connectors is shown exemplarily only. The contacts are drawn in their non-activated position. In most examples it is assumed that an intelligent safety input is used, which is capable to do a 1oo2[1] evaluation with the required fault-detection and provides one SAFEBOOL signal. However there might be safety input devices, which are capable to process the contactors separately as 1oo1[2]. The fault-detection is done for each channel separately. In this case additional measures might be necessary on the safety application level as well on the peripheral wiring (different test pulses, fault-exclusions, etc.).

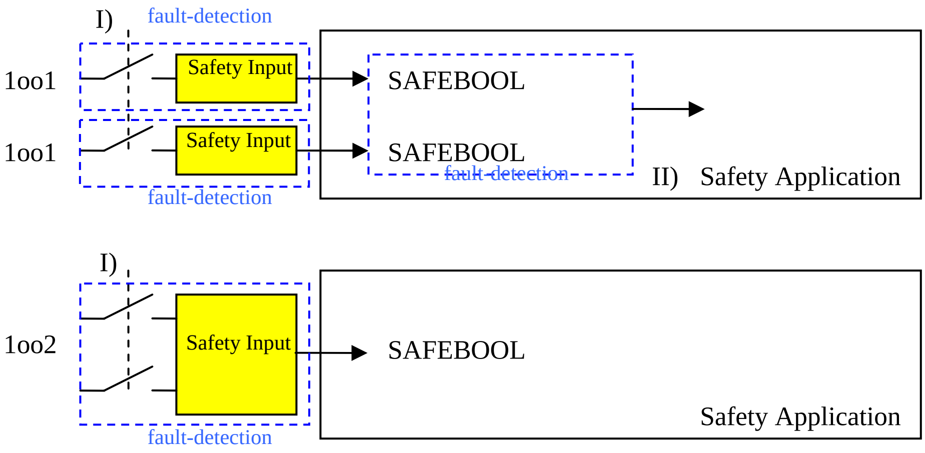


Figure 3: Different input connections to generate a SAFEBOOL

I. Contactors can be any combination out of normally closed and normally open regarding the requirements and the fault-detection capabilities of the safety input
II. Additional fault-detection has to be implemented. This is either implemented in the signal processing FB or could be implemented via the FB SF_Equivalent or SF_Antvialent separately.

SF_Equivalent:

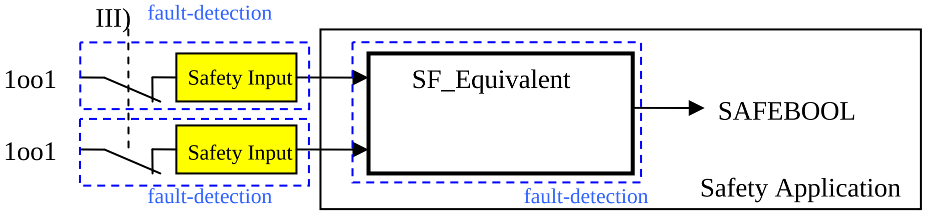


Figure 4: Use of SF_Equivalent

III. Instead of two “NO” (normally open) contactors, two “NC” (normally closed) contactors could be used.

This FB processes two SAFEBOOL inputs and monitors that the signal changed equivalent within a specified monitoring time.

[1] 1oo2: one out of two sensors must be working as expected to generate the SAFEBOOL representation. If one does not fulfil the expected behaviour the SAFEBOOL is set to the fail-state value.

[2] 1oo1: one out of one sensors must be working as expected to generate the SAFEBOOL representation.

SF_Antivalent:

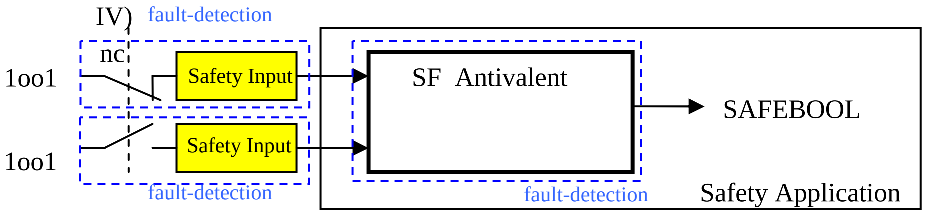


Figure 5: Use of SF_Antivalent

IV. It is required to connect the NC contactor to a specific FB input. (See Part 1 for more details)

This FB processes two SAFEBOOL inputs and monitors that the signal changed antivalent within a specified monitoring time.

Some FBs are able to deal with both ways of connecting the peripheral contacts. Therefore they have to be connected as shown.

SF_GuardMonitoring:

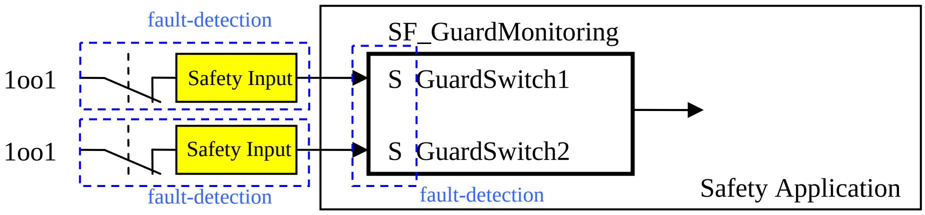


Figure 6: Use of SF_GuardMonitoring with two 1oo1 inputs

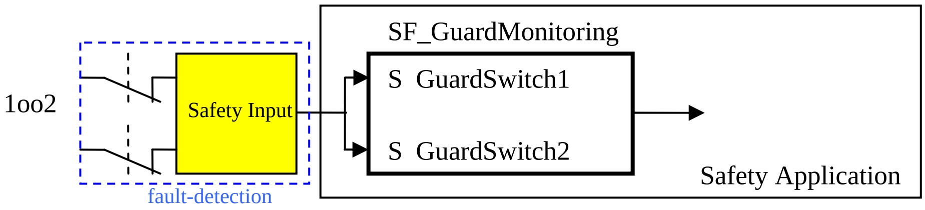


Figure 7: Use of SF_GuardMonitoring with two 1oo2 inputs

Some FBs require that each channel is connected directly such as SF_EnableSwitch:

V)

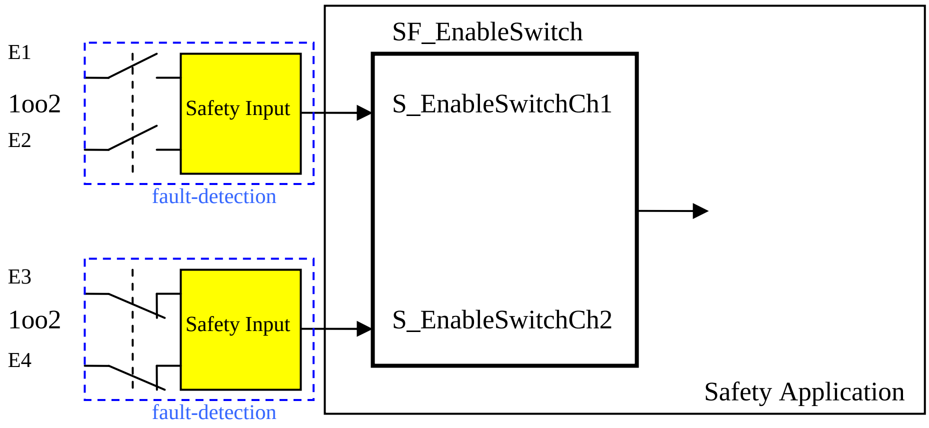


Figure 8: Use of inputs for SF_EnableSwitch

V. The FB can be only used for 3-position enable switches that perform following signal diagram.

	Pos 1	Pos 2	Pos 3
Ch1	0	1	0
Ch2	1	1	0
	←	▨	

There are many different enable switches available. Some of them have an internal electrical circuit; some of them have directly accessible contactors.

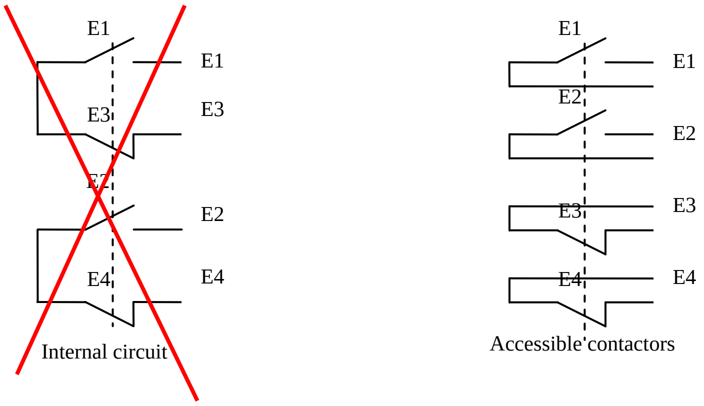


Figure 9: Examples of enable switch configurations (left one cannot be used with SF_EnableSwitch)

3.2. Information to the graphical overview of the safety application examples

Figure 10 shows on the left side the separation between the Safety and Functional Application as already defined in Part 1. However from the safety function view (right side), safety and standard variables effect the processing of the Safety FBs, even though the standard variables do not affect the safety at all since the safety application has to enable any safety related action.

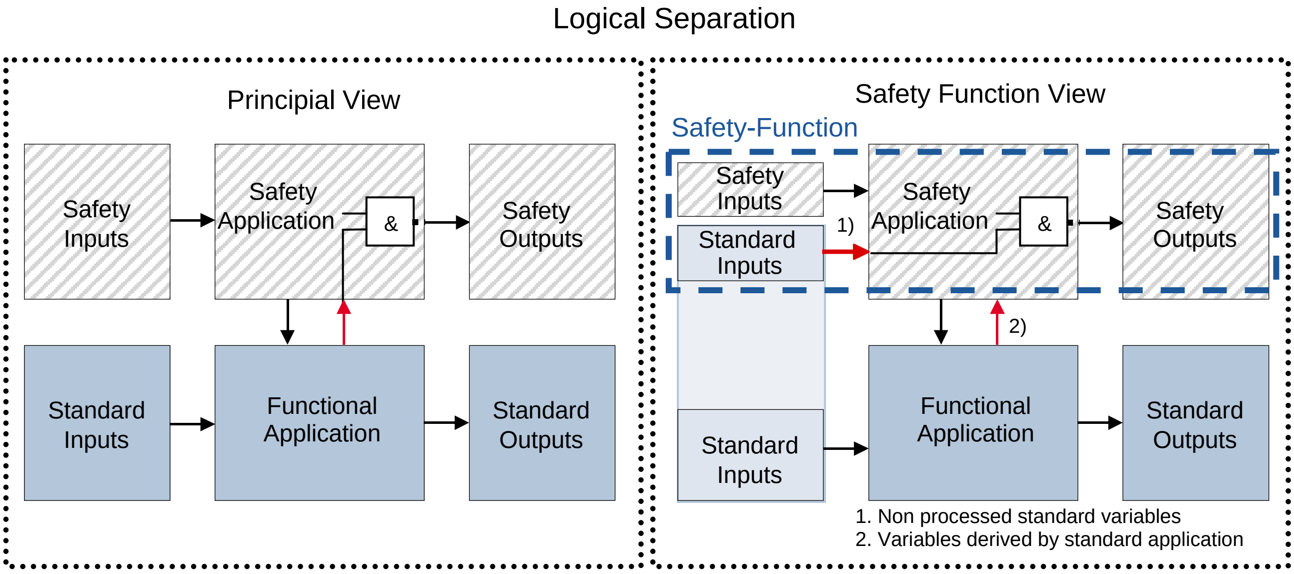


Figure 10: Architectural model

To recognize the corresponding signals easier, in the graphical interface overview of Part 2 – Users Examples, the Input Signals get sorted in correspondence to the safety functions and get assigned directly to the safety application regardless if they belong to the functional or safety application, as shown on the right side in the figure above. Most of these standard signals (1) do not get manipulated by the Functional Application anyway. Some systems are able to address the Standard Inputs/Outputs directly from the Safety Application; others provide these signals through the Functional Application.
However the separation to the different physical input/output blocks and the assignment to the functional or safety application still match with the basic architectural model as defined in part 1. The signals (2), processed by the Functional-application and which do not belong to specific safety function such as reset, diagnosis, are simplified illustrated at the interface between the two applications.
Note: Within the examples, the use of contactors and their connection to the input and output units is just illustrated simplified and cannot in general be transferred one-to-one into real applications.

The real installation depends on the required level of safety integrity, the applicability of the sensors/actors, the capability of the system and the fault exclusions.
These may vary between different safety control systems. Therefore, the supplier’s documents, standards, and further related information have to be considered.

3.3. Information for the use of safe drives (with a hidden system level interface)

Since the drive internal state machine is complex and vendor specific, the control and status information to handle it will be exchanged between the drive and the control system via a hidden system level interface. So the user does not need to program every single bit. The FBs provide the necessary interface.
The hidden interface could utilize a discrete output device however will be mostly a digital fieldbus interface. Since there are no transparent variables in the application program, in the following graphical overviews of the examples the data exchange will be represented by the FB instance and the actor as shown in figure Figure 11: Overview of hidden interface for drives below.

Safe Drives are drives with built-in safety functions. Normally the drive differentiates between modes where no safety functions are active and where safety is active. In the safety mode there are different safety functions possible. The transitions between the safety functions normally get handled by a drive internal state machine and might underlay different priorities. The behaviour of the state machine might depend on a set of drive internal parameters as well.

The corresponding PLCopen FBs provide an interface to control this state machine. They do not perform the safety functions at all.

To select the different safety functions or to enable movement mostly a system level interface gets used. That means that the control and status bits are not visible like it is possible if they get transmitted via a discrete IO interface. Such FBs are

- SF_SafeStop1
- SF_SafeStop2
- SF_SafelyLimitedSpeed.

However a discrete I/O interface could be utilized by using the SF_SafetyRequest FB instead.

In any case, the interface to the drive uses the idle current principle. This means that if for some reason the communication is lost, the drive performs independently a fail-safe reaction (e.g. a safe stop in accordance with stop category 1 EN60204). If the FB detects any error, the safe state will be requested automatically. Such an error can be a corrupted acknowledge signal, or the communication is lost.

Safely Limited Speed
Due to the requirement, using an enabling device to allow machine operation, the machine needs to be stopped safely when the enabling switch is released or a panic situation is detected. The SF_SafelyLimitedSpeed has the SafeStop functionality incorporated. The SafeStop function could be parameterized either as a SafeStop1 or SafeStop2. As long as safety is required, e.g. due to an operator who wants to get access to the hazardous area to clear a material jam, a SafeStop needs to be performed. To get the material out of the machine, the operator might need to get the machine in operation. Therefore he uses an enabling switch, which activates the real SafelyLimitedSpeed function. Or with other words, if the enabling switch is not actuated the permitted velocity is zero. After the enable switch is activated a speed lower than the safely limited speed can be run. However the speed command has to be activated separately by the motion controller and has nothing to do with the safety function at all. It does influence the availability only, since in the case that the command value is higher than the permitted speed, the drive would perform a fail-safe reaction. (See Part 1 – SF_SafelyLimitedSpeed for more information)

Note: To provide a better understanding of the following examples, the hidden system signals get represented in the interface overview by the FB instances. They are not represented by variables.

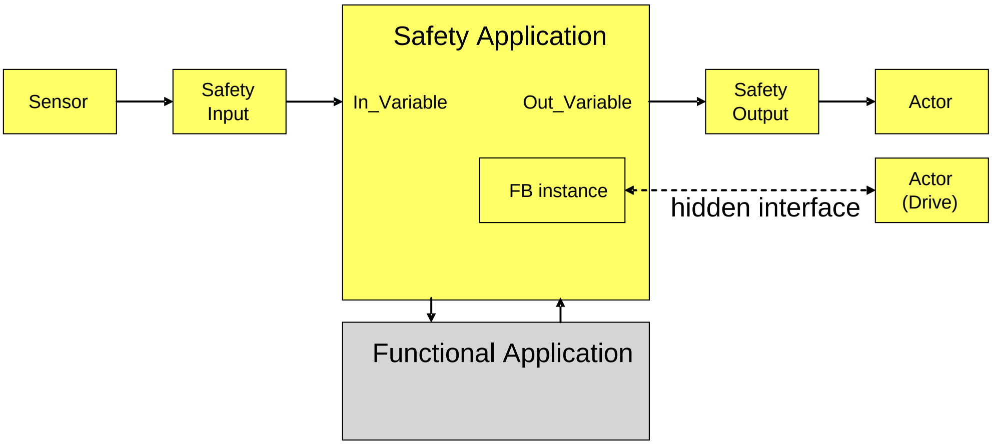


Figure 11: Overview of hidden interface for drives

4 Application Examples

The following application examples are provided within this document:

4.1 Diagnostics concept
4.2 Safe Motion with hidden interfacing
4.3 Muting
4.4 Safe Motion with I/O interface
4.5 Two-Hand Control

The usage of defined FBs of Part 1 in the examples

Name FB / Example	***4.1***	***4.2***	***4.3***	***4.4***	***4.5***
SF_Equivalent	x				
SF_Antivalent					
SF_ModeSelector		x		x	
SF_EmergencyStop	x	x	x	x	x
SF_ESPE	x				
SF_SafeStop1	x	x			
SF_SafeStop2					
SF_GuardMonitoring		x	x		
SF_SafelyLimitedSpeed		x			
SF_TwoHandControlTypeII					x
SF_TwoHandControlTypeIII					(1)
SF_GuardLocking		x		x	
SF_TestableSafetySensor			x		
SF_MutingSeq			x		
SF_MutingPar					
SF_MutingPar_2Sensors					
SF_EnableSwitch		x		x	
SF_SafetyRequest				x	
SF_OutControl			x		x
SF_EDM			x		x

(1) Example 4.5 can also be used with the SF_TwoHandControlTypeIII.

4.1. Diagnostics concept

This example shows the usage of the diagnostic concept, with a daisy chain from the FB parameters Activate and Ready (with perhaps a pre-evaluation of hardware errors), and a coupling to a HMI screen with error IDs and messages through the functional application. For clarity sake of the safety function the other examples will not show the diagnostic connections.
The safety functionality is to stop a drive in accordance with stop category 1 of IEC 60204-1 initiated by an Emergency stop or by interrupting the light curtain. The equivalent monitoring of the 2 connectors of the emergency switch is done in the safety application.
In this example both possibilities of input evaluation are shown: once via intelligent safety input and once via the equivalent function block.

4.1.1. Functional description of safety functions

This example uses the following safety functions:

- Issuing the emergency stop (via SF_EmergencyStop) or interrupting the light beam in the light curtain (via SF_ESPE) stops the drive in accordance with stop category 1.
- The stop of the electrical drive within a predefined time is monitored (via SF_SafeStop1).
- The Safe Status of the drive is indicated by the S_Stopped variable, connected to the functional application
- If the stop is performed by the Emergency Switch, a manual reset is required (via SF_EmergencyStop)
- If a monitoring time violation is detected (via SF_SafeStop1), manual error acknowledge is required to allow a reset.
- The 2 channel connectors of the emergency stop are monitored. An error is detected when both inputs do not have the same status once the discrepancy time has elapsed (via SF_EQUIVALENT)
- The functional stop in this example is performed as a safe stop issued from the functional application. A restart interlock for this stop is not necessary.

4.1.2. Graphical overview of the safety application interface

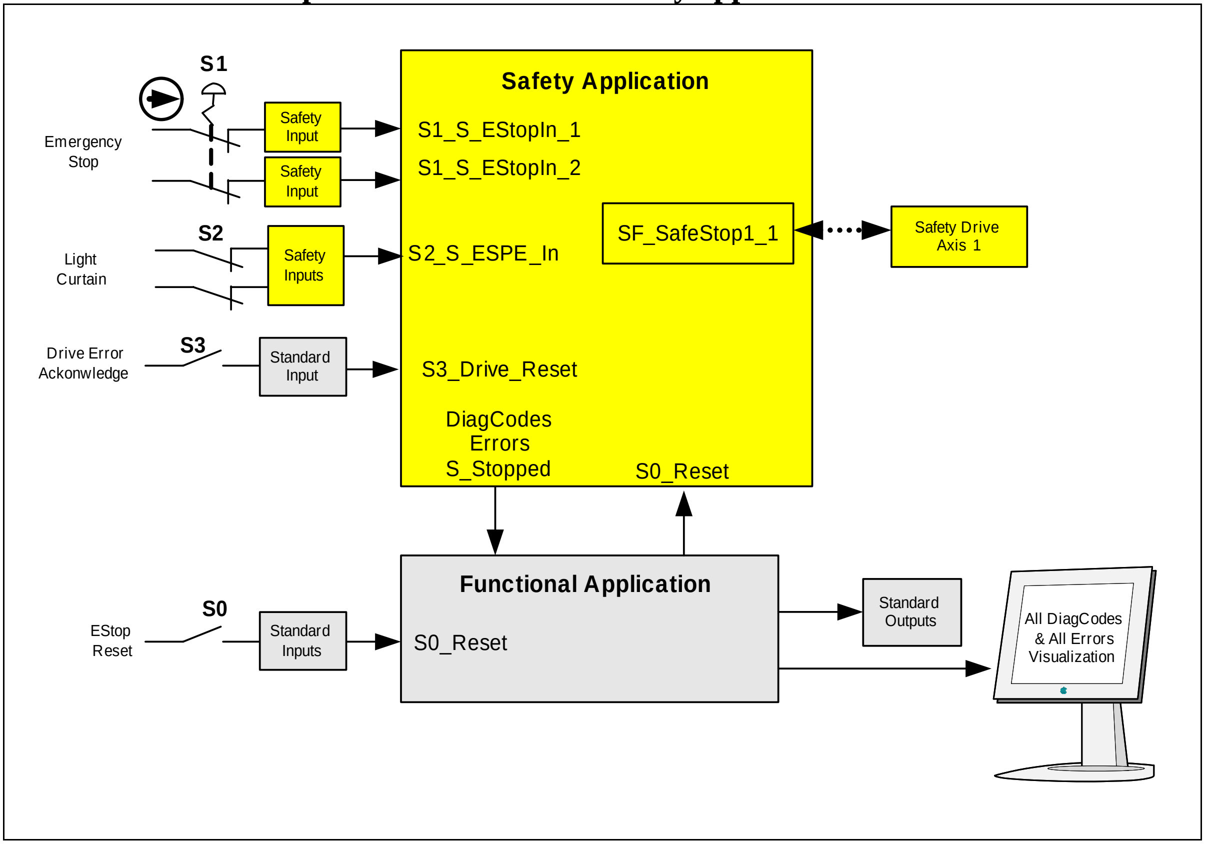


Figure 12: Graphical overview of the example with emergency stop

Note: the symbol ⊖ represents a direct opening action (cf. IEC 60947-5-1).

4.1.3. Declaration of the used variables

Inputs

Name	Data type	Description
S1_S_EstopIn_1	SAFEBOOL	Emergency Stop Channel 1
S1_S_EstopIn_2	SAFEBOOL	Emergency Stop Channel 2
S2_ESPE_In	SAFEBOOL	Light curtain signal
S0_Reset	BOOL	Reset Emergency Stop and ESPE
S3_Drive_Reset	BOOL	Reset Drive Error

Outputs

Name	Data type	Description
S_Stopped	SAFEBOOL	Indication of Safe Stop of drive
All Errors	BOOL	Errors of SF_Function Blocks
All Diagcodes	BOOL	Diag codes of SF_Function Blocks

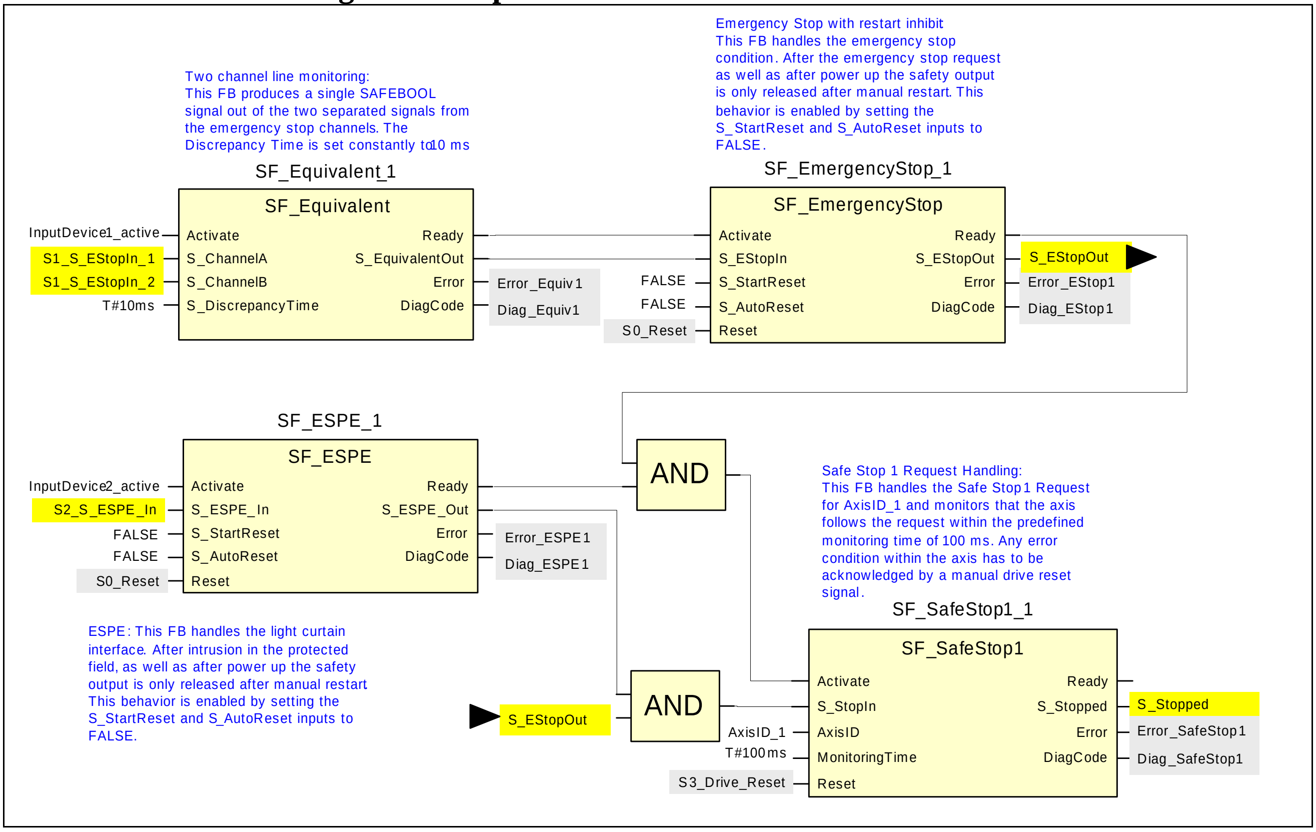
Hidden Interface of FB instances towards drives (Vendor specific)

Name	Description
SF_SafeStop1_1	Connection to Drive 1

Local variable

Name	Data type	Description
S_EStopOut	SAFEBOOL	Emergency stop request
InputDevice1_active	BOOL	Status of the relevant input device as provided by the system
InputDevice2_active	BOOL	Status of the relevant input device as provided by the system

4.1.4. Program example



Two channel line monitoring: This FB produces a single SAFEBOOL signal out of the two separated signals from the emergency stop channels. The Discrepancy Time is set constantly to 10 ms

Emergency Stop with restart inhibit This FB handles the emergency stop condition. After the emergency stop request as well as after power up the safety output is only released after manual restart. This behavior is enabled by setting the S_StartReset and S_AutoReset inputs to FALSE.

ESPE: This FB handles the light curtain interface. After intrusion in the protected field, as well as after power up the safety output is only released after manual restart This behavior is enabled by setting the S_StartReset and S_AutoReset inputs to FALSE.

Safe Stop 1 Request Handling: This FB handles the Safe Stop1 Request for AxisID_1 and monitors that the axis follows the request within the predefined monitoring time of 100 ms. Any error condition within the axis has to be acknowledged by a manual drive reset signal.

Figure 13 – Program example – Emergency stop with safe stop & equivalent monitoring

4.1.5. Additional Notes

This example uses different reset signals to acknowledge the emergency stop situation and to acknowledge the monitoring violation situation of the drive. If the safety requirement specification of the application allows the acknowledgement of both situations with the same signalling device, the identical signal from the functional application may be used to reset the FB SF_EmergencyStop_1 as well as to reset the FB SF_SafeStop1_1.

Information on the diagnostics concept
The representation of the diagnostics concept is for information only. For the safety functionality the dedicated safety inputs and outputs shall be used.

Daisy chain from Activate and Ready:
The connection of the Ready output to an Activate input of the following FB ensures that no irrelevant diagnostic information is generated if a device is disabled. The daisy chain from Activate and Ready avoid subsequent error messages of related function blocks.

Pre-evaluation of Hardware errors:
If the target system supports an error signal e.g. InputDevice_active, which represents the status (Active or Not Active) of the relevant safety device, this signal can be used to disable the safety function blocks. This ensures no irrelevant diagnostic information is generated if a device is disabled. If no such error signal is provided by the target system, a static TRUE signal must be assigned to the Activate input.

Evaluation of the diagnostic information:
The Error signals and DiagCodes of each safety function block are transferred to the standard application. Diagnosis information might be processed and displayed by an attached visualization. There are different possibilities to realize the evaluation of the diagnostic information:

- Transfer these values into the visualization and realize the diagnostic evaluation in the visualization.
- Realize the diagnostic evaluation in the standard logic and transfer the results to the visualization

Because of the various possibilities and the differences in the target system to realize diagnostic processing, there is no special example showed here. Further diagnostic processing could be:

- Display of the error status for each safety function block
- Providing an error overview which is linked to function block specific error displays
- Detection and display of the last error of the used safety function blocks in the safety application.

Information on the used function block parameters

Function Block	**Input**	**Constant Value**	**Description**
SF_Equivalent_1	S_DiscrepancyTime	10 ms	Maximum monitoring time for discrepancy status of both inputs.
SF_EmergencyStop_1	S_StartReset	FALSE	Manual reset when PES is started (warm or cold).
	S_AutoReset	FALSE	Manual reset when emergency stop button is released.
SF_SafeStop1_1	AxisID	AxisID_1	Drive address, supplier specific value
	MonitoringTime	100 ms	Time until the drive shall be stopped.
SF_ESPE	S_StartReset	FALSE	Manual reset when PES is started (warm or cold).
	S_AutoReset	FALSE	Manual reset after safety demand condition is cleared.

4.2. Safe Motion with hidden interfacing

4.2.1. Functional description of safety functions

The example describes a machine with two electric drive systems within a working area where an operator needs access to, e.g. for process diagnosis, set-up activities or to clear a material jam.
The access to the working area is provided by an interlocking guard with guard locking. The locking is required due to the fact that the operator could get access to the hazardous area before a stop of the drive system is be performed completely.
In emergency situation the drive systems needs to be stopped in accordance with stop category 1 (EN60204).
A mode selector is used to switch the machine between automatic and set-up mode.
Within the set-up mode the guard door can be opened and the drive systems enabled to move with a safely limited speed by using an enabling device.

The emergency stop (via SF_EmergencyStop) acts superimposed to all other safety functions and puts the drive systems into a safe standstill (via SF_SafeStop1) in accordance with stop category 1 of EN60204-1.

After an emergency stop, the restart of the machine is only possible after the emergency button is released and a reset signal is given (via SF_EmergencyStop).

The (normal) operation of the machine is only possible within the automatic mode (via SF_ModeSelector) and the guard door closed (via SF_GuardMonitoring) and locked (via SF_GuardLocking).

The guard door lock can be released within the set-up mode (via SF_ModeSelector) or after an emergency stop (via SF_EmergencyStop) as soon as the drive systems are performing a safe standstill (via SF_SafelyLimitedSpeed or SF_SafeStop1).

In the set-up mode (via SF_ModeSelector) the drive systems can be switched with the enabling device (via SF_EnableSwitch) into a mode where movement with safely limited speed is allowed (via SF_SafelyLimitedSpeed). The drive system can be moved by the motion controller via the standard command values. The drive itself has to guarantee safely that the speed limit gets not exceeded. (Note: If the motion control command values are greater than the parameterized limit the drive system performs a fail-safe reaction independently.)

Without an enable signal (SF_EnableSwitch) the drive system stays in a safe standstill mode, where the speed gets monitored to be zero (SF_SafelyLimitedSpeed) as long as set-up mode is selected. (Note: In this case the SF_SafelyLimitedSpeed FB puts the drive into a safe operational stop in accordance with stop category 2 (EN60204) and acts like the SF_SafeStop2 FB.)

4.2.2. Graphical overview of the safety application interface

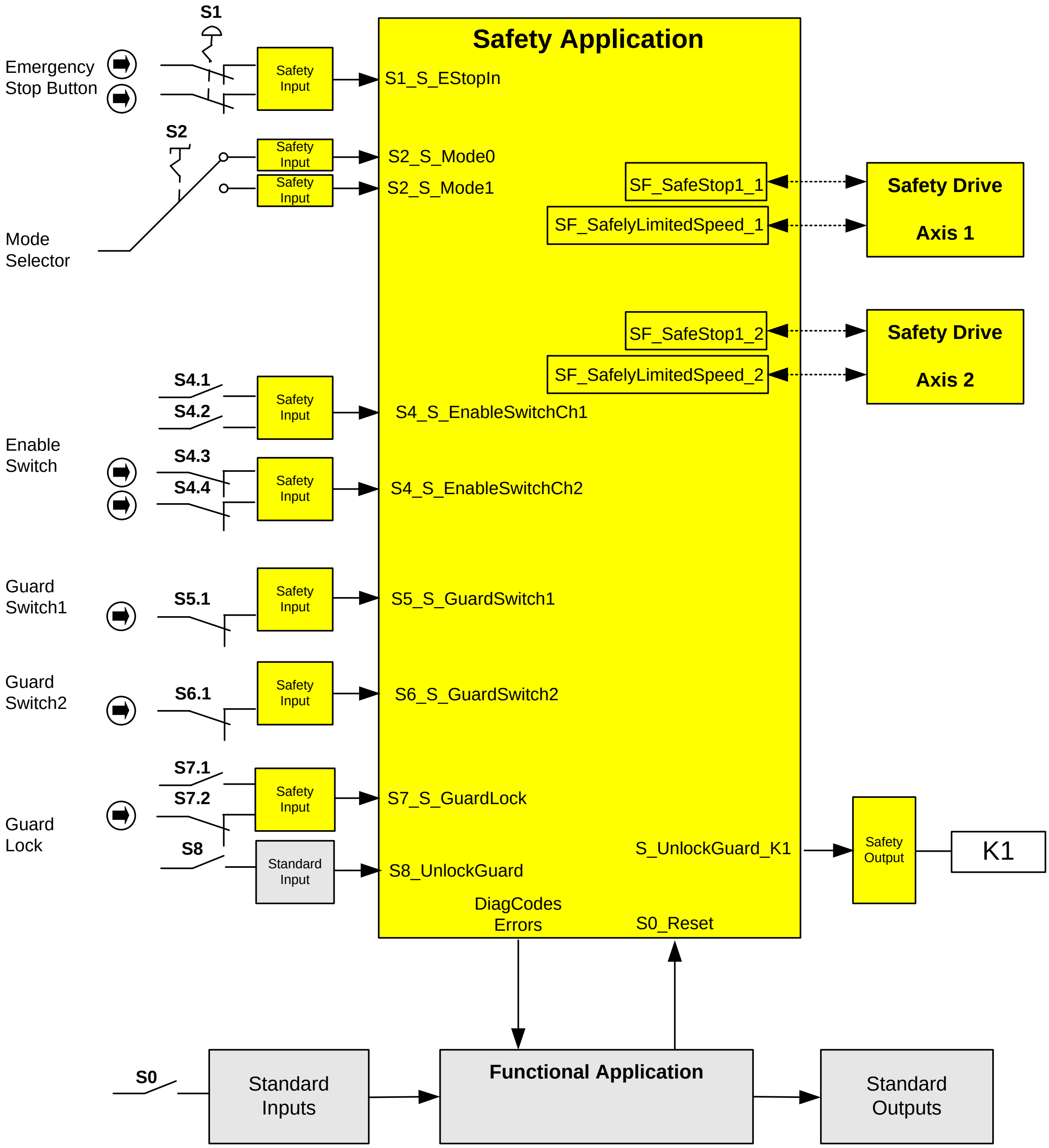


Figure 14 - Graphical Overview of Safe Motion Application

4.2.3. Declaration of the used variables

Inputs

Name	Datatype	Description
S1_S_EStopIn	SAFEBOOL	E-Stop
S2_S_Mode0	SAFEBOOL	Automatic Mode
S2_S_Mode1	SAFEBOOL	Setup Mode
S4_S_EnableSwitchCh1	SAFEBOOL	Enabling Device E1+E2
S4_S_EnableSwitchCh2	SAFEBOOL	Enabling Device E3+E4
S5_S_GuardSwitch1	SAFEBOOL	Guard Monitoring
S6_S_GuardSwitch2	SAFEBOOL	Guard Monitoring
S7_S_GuardLock	SAFEBOOL	Guard Lock Monitoring
S0_Reset	BOOL	Reset
S8_UnlockGuard	BOOL	Request to unlock Guard

Outputs

Name	Datatype	Description
S_UnlockGuard_K1	SAFEBOOL	Unlock Guard

Hidden interface of FB instances towards the drives (Vendor specific)

Name	Description
SF_SafeStop1_1	Connection to Drive 1
SF_SafelyLmitedSpeed_1	Connection to Drive 1
SF_SafeStop1_2	Connection to Drive 2
SF_SafelyLmitedSpeed_2	Connection to Drive 2

Note: If the drive does not support a hidden interface, external signals must be used instead. Therefore the FB SF_SafeRequest shall be used.

Local Variables

Name	Datatype	Description
S_SafeStopAxis1	SAFEBOOL	Stop of Axis 1
S_SafeStopAxis2	SAFEBOOL	Stop of Axis 2
S_SafeMoveAxis1	SAFEBOOL	Safely Limited Speed Axis 1
S_SafeMoveAxis2	SAFEBOOL	Safely Limited Speed Axis 2
S_SafetyActive	SAFEBOOL	All axis are in any safe mode
S_ModeAutoActive	SAFEBOOL	Mode Automatic active
S_ModeSetSel	SAFEBOOL	Mode Set-up selected
S_SLS_Enable	SAFEBOOL	Enable of SafelyLimitedSpeed (output of Safety Enable Switch)
S_GuardLocked	SAFEBOOL	Indication of the status of the interlock

4.2.4. Program Example

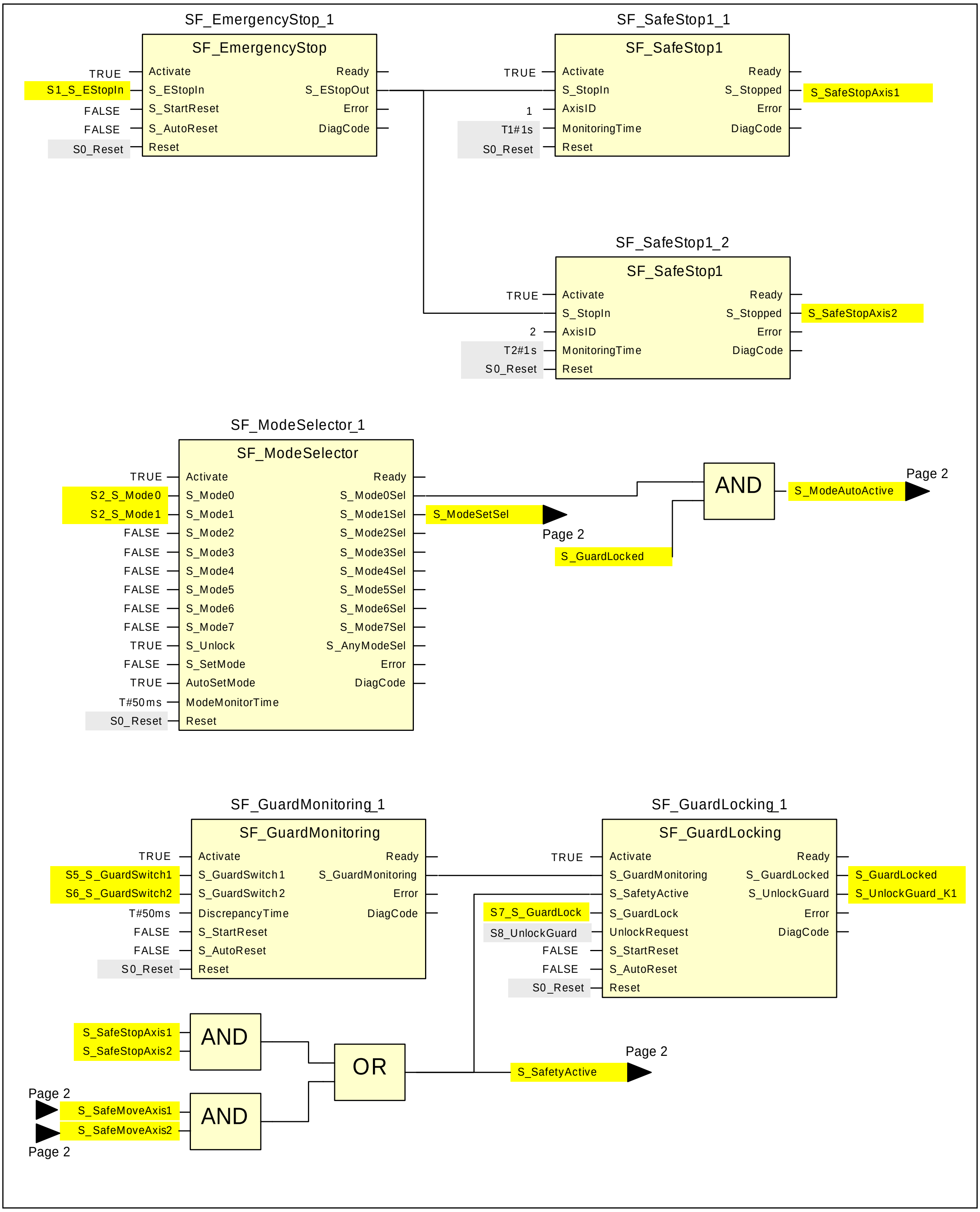


Figure 15 – Safe Motion with hidden interfacing - Application Program Page 1

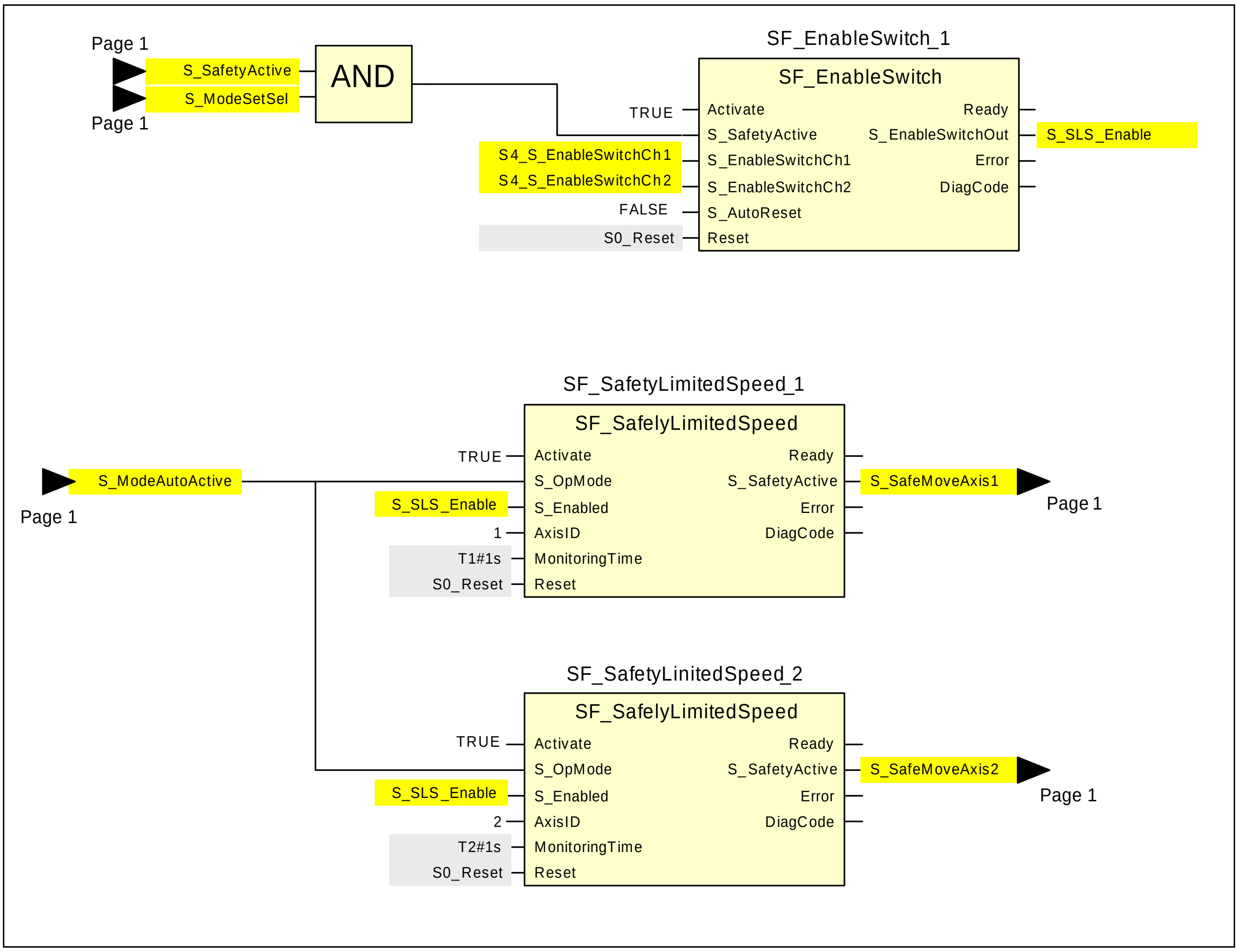


Figure 16 - Safe Motion with hidden interfacing - Application Program Page 2

4.2.5. Additional Notes

In this example an input device has been used to map the two channels to one SAFEBOOL.

The diagnostic information retrieval has not been covered in this example. For this, refer to example 4.1.5 Additional Notes.

The input Activate has been set to TRUE via its input for clarity sake. However in an application, this can be replaced by a variable.

There are only two operation modes regarding safety. In Automatic mode no safety functions have to be active. In any other modes safety has to be ensured. Therefore the safety functions are not derived on a specific output of the mode selector. Furthermore the condition is, if automatic is not selected, safety has to be ensured. The drive FBs differentiate only between a safe and a non-safe (operation) mode. In order of the current idle principle the automatic mode output of the mode selector could be linked directly to the S_OpMode input of the drive FBs.

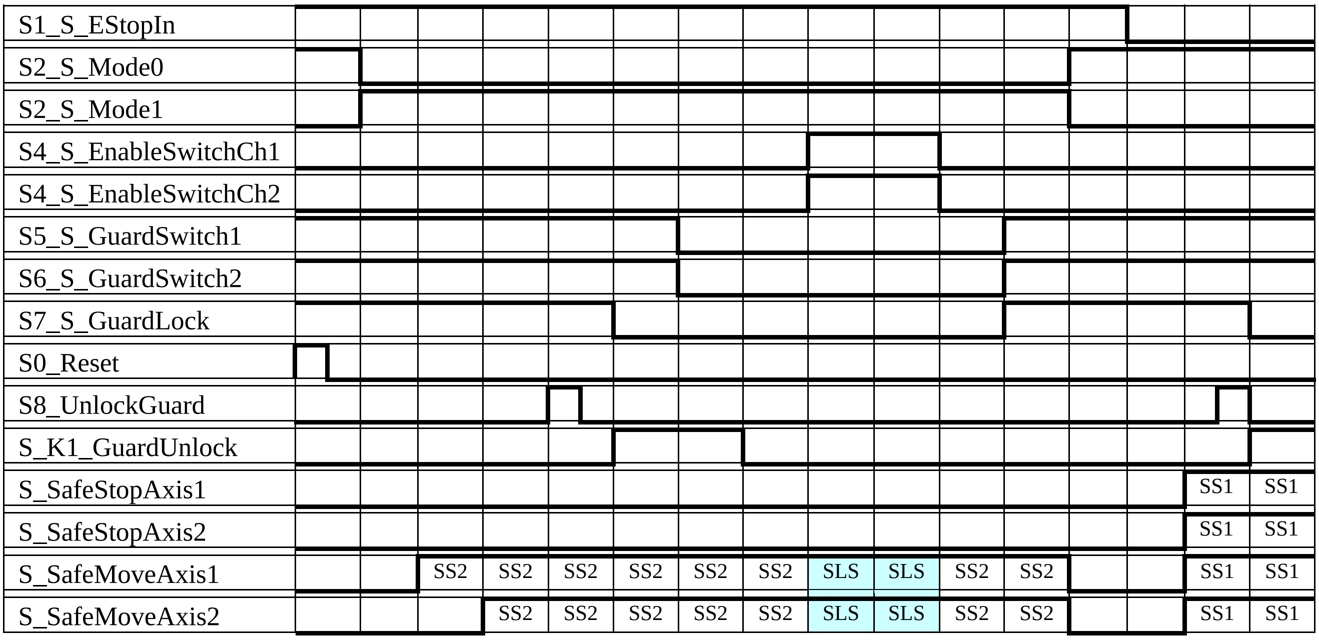
If a non-automatic mode is selected and the enable switch is not activated the SF_SafelyLimitedSpeed acts as a SF_SafeStop2. (See chapter 3.3 Information for the use of safe drives (with a hidden system level interface) for more information about the use of SF_SafelyLimitedSpeed.)

To open the guard after a stop under normal operation conditions a non-automatic mode has to be selected by the mode selector switch first. Also the user has to request the release the interlock by setting the S8_UnlockGuard signal.

Information about the used Function Block Parameters

Function Block	Input	Constant Value	Description
SF_ModeSelector_1	S_Unlock	TRUE	A change is possible and is not frozen.
	AutoSetMode	TRUE	A change does not need to be confirmed
	S_SetMode	FALSE	Therefore S_SetMode is not used
	ModeMonitoringTime	50 ms	Maximum time for an ambiguous switch position
SF_EmergencyStop_1	S_StartReset	FALSE	Avoidance of an unintended restart required No automatic reset allowed when PES is started
	S_AutoReset	FALSE	Avoidance of an unintended restart required No automatic reset allowed
SF_EnableSwitch_1	S_AutoReset	FALSE	Avoidance of an unintended restart required No automatic reset allowed
SF_GuardMonitoring_1	S_StartReset	FALSE	Avoidance of an unintended restart required No automatic reset allowed when PES is started
	S_AutoReset	FALSE	Avoidance of an unintended restart required No automatic reset allowed
	DiscrepancyTime	50 msec	Maximum time for discrepancy status of both inputs
SF_GuardLocking_1	S_StartReset	FALSE	Avoidance of an unintended restart required No automatic reset allowed when PES is started
	S_AutoReset	FALSE	Avoidance of an unintended restart required No automatic reset allowed
SF_SafeStop1_1	MonitoringTime	1 s	Maximum time for the drive system to get into a safe stop
	AxisID	1	Drive 1
SF_SafeStop1_2	MonitoringTime	1 s	Maximum time for the drive system to get into a safe stop
	AxisID	2	Drive 2
SF_SafelyLimitedSpeed_1	MonitoringTime	1 s	Maximum time for the drive system to get into the safely limited speed state
	AxisID	1	Drive 1
SF_SafelyLimitedSpeed_1	MonitoringTime	1 s	Maximum time for the drive system to get into the safely limited speed state
	AxisID	2	Drive 2

Typical Timing Diagram



SS2 is equivalent to the state „Safe Operational Stop" of the FB "SF_SafelyLimitedSpeed"
SLS is equivalent to the state „Safely Limited Speed" of the FB "SF_SafelyLimitedSpeed"

4.3. Muting

This example describes the safety functions for the safeguarding of a production cell. Objects are transferred through an entry gate, which is guarded by a light curtain. This light curtain can be muted only for material transport into the cell. The cell may be entered by the operator through a safety door. The process inside the cell is controlled by the functional application and enabled by the safety circuit. In case of a safety demand or an error, all hazardous movements are stopped in accordance with stop category 0.

4.3.1. Functional description of safety functions

All hazardous movements are stopped in case of:

- an opening of the door,
- an error (e.g. invalid muting sequence),
- an interruption of the unmuted light curtain (e.g. by a person),
- pushing an emergency stop button.

By pushing an emergency stop button, the operator can also stop all hazardous movements in stop category 0 (via SF_EmergencyStop and subsequent FBs).

An infringement of the unmuted light curtain stops all hazardous movements. In this application, a light curtain type 2 is used, which requires a test by the FB SF_TestableSafetySensor.

For the described muting function, four muting sensors are applied sequentially (via SF_MutingSeq). Additionally, the muting phase is indicated by a lamp, which is monitored in this case (also via SF_MutingSeq).

An additional door for maintenance purposes is monitored by a door switch (via SF_GuardMonitoring).
By resetting buttons, the operator must acknowledge the detected demand of the safety functions and errors.
The initial state and the operational state of the connected actuator are checked by an external device monitoring. In case an error is detected, the control cannot become operational (via SF_EDM).

The process and related movements inside the production cell are controlled by the functional application. Within the safety application, this control is enabled by the above-described safety circuit (via SF_OutControl) and drives the actuator via a safety output.

4.3.2. Graphical overview of the safety application interface

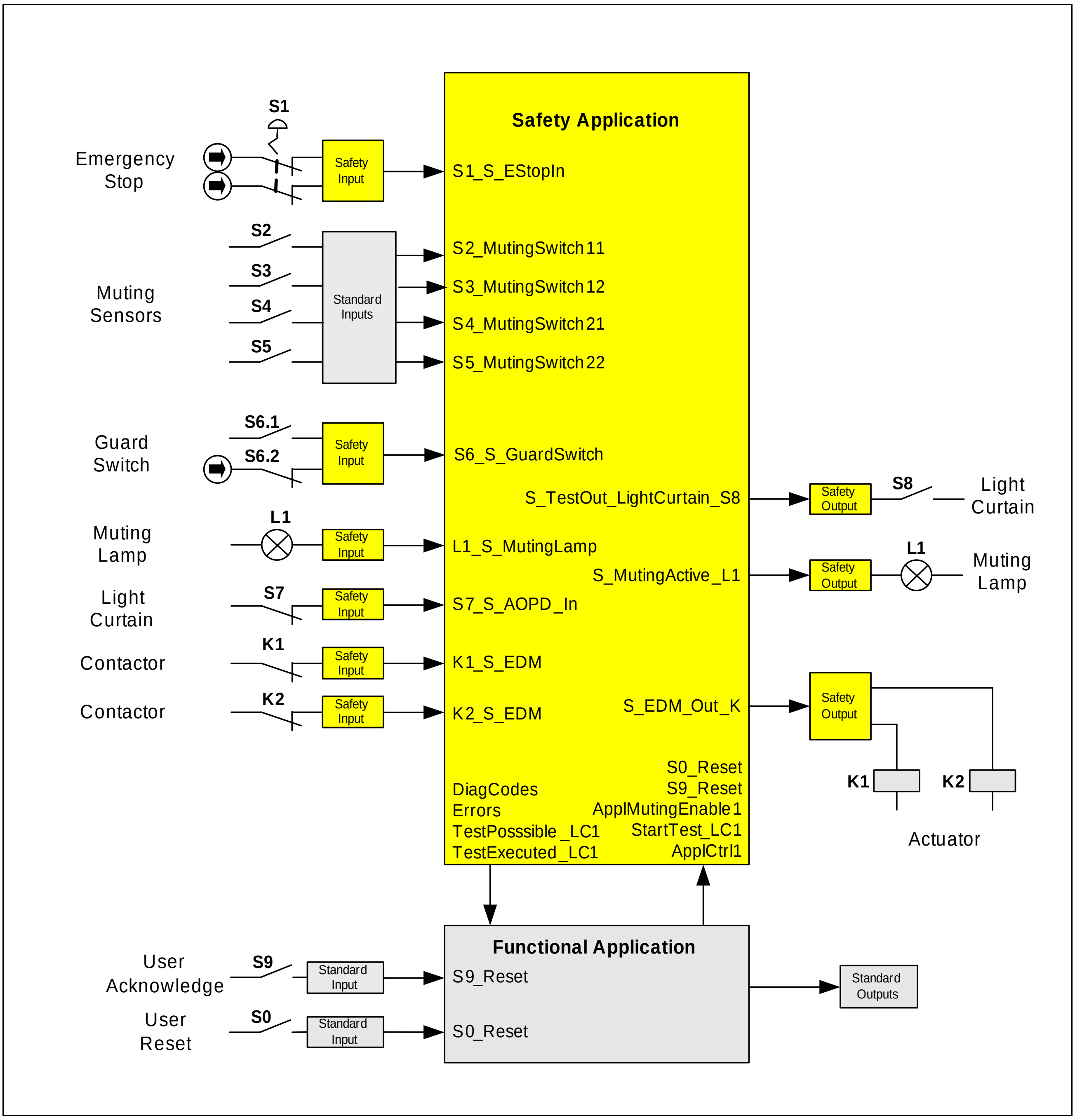


Figure 17: Graphical overview of the example access protection at a material gate

4.3.3. Declaration of the used variables

Inputs

Name	Data type	Description
S1_S_EStopIn	SAFEBOOL	Emergency stop button S1
S2_MutingSwitch11	BOOL	Muting sensor S2
S3_MutingSwitch12	BOOL	Muting sensor S3
S4_MutingSwitch21	BOOL	Muting sensor S4
S5_MutingSwitch22	BOOL	Muting sensor S5
S6_S_GuardSwitch	SAFEBOOL	Door switch S6 with two contacts
L1_S_MutingLamp	SAFEBOOL	Muting lamp monitor signal L1
S7_S_AOPD_In	SAFEBOOL	OSSD from light curtain S7
K1_S_EDM	SAFEBOOL	Feedback external device K1
K2_S_EDM	SAFEBOOL	Feedback external device K2
S9_Reset	BOOL	Reset safety demand by user S9
S0_Reset	BOOL	Reset error by user S0 (derived from functional application)
ApplCtrl1	BOOL	Signal controlling the actuator, enabled by safety loop (derived from functional application)
StartTest_LC1	BOOL	Signal starting test of light curtain S7 (derived from functional application)
ApplMutingEnable1	BOOL	Signal enabling start of the muting sequence (derived from functional application)

Outputs

Name	Datatype	Description
S_EDM_Out_K	SAFEBOOL	Drives actuator via K1 and K2
S_MutingActive_L1	SAFEBOOL	Drives Muting lamp L1
S_TestOut_LightCurtain_S8	SAFEBOOL	Test output for light curtain S8
All Errors	BOOL	Represents all error parameter of the used FB (connected to functional application)
All DiagCodes	WORD	Represents all diagnosis codes of the used FB (connected to functional application)
TestPossible_LC1	BOOL	Indicates to the functional application that an automatic sensor test of the light curtain is possible.
TestExecuted_LC1	BOOL	Indicates to the functional application the successful execution of an automatic sensor test of the light curtain.

Local variables

Name	Datatype	Description
S_SafeControl	SAFEBOOL	Indicates the status of the safety guards (TRUE = safety enabled)

4.3.4. Program example

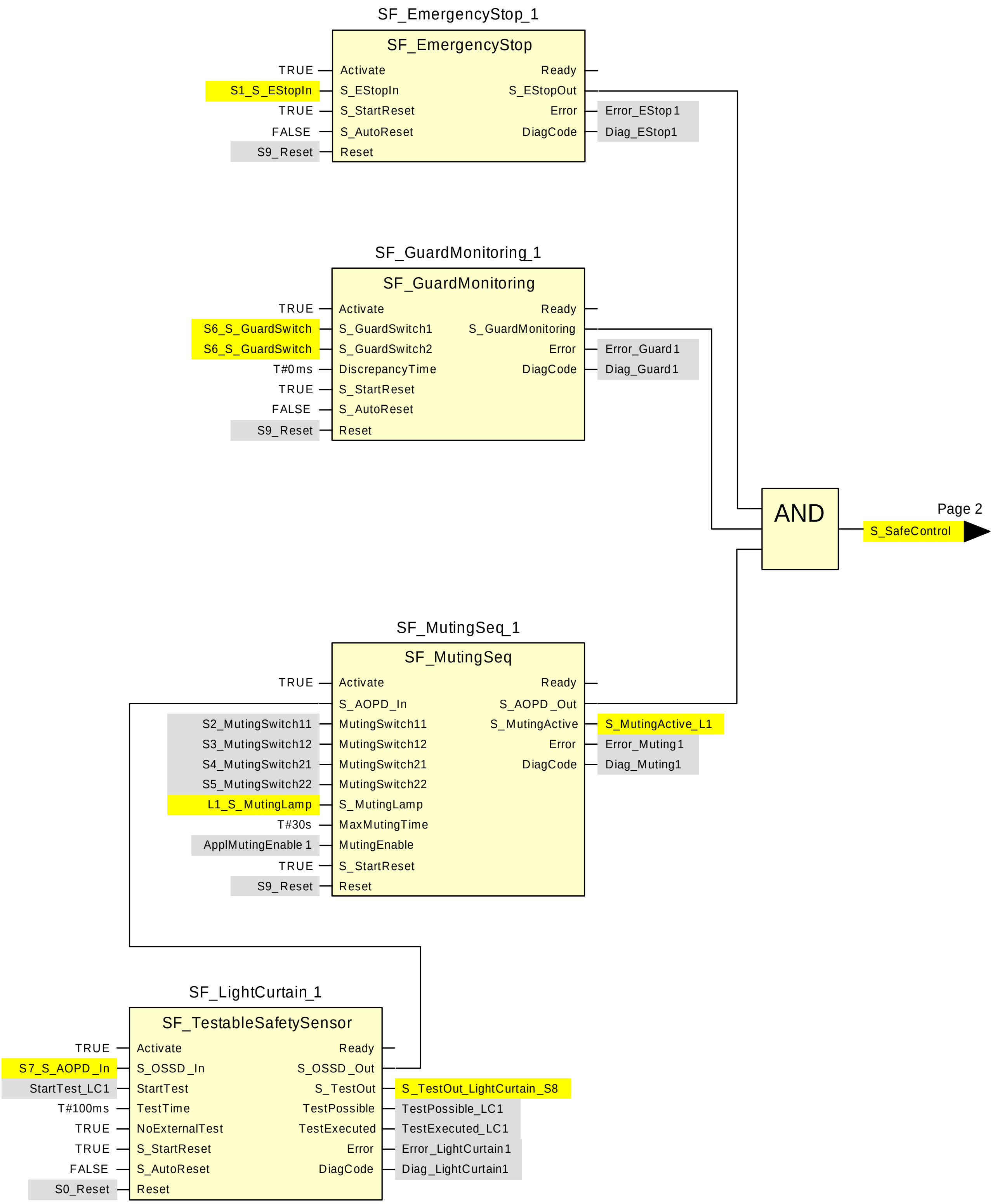


Figure 18 – Access protection at a material gate - Application Program Page 1

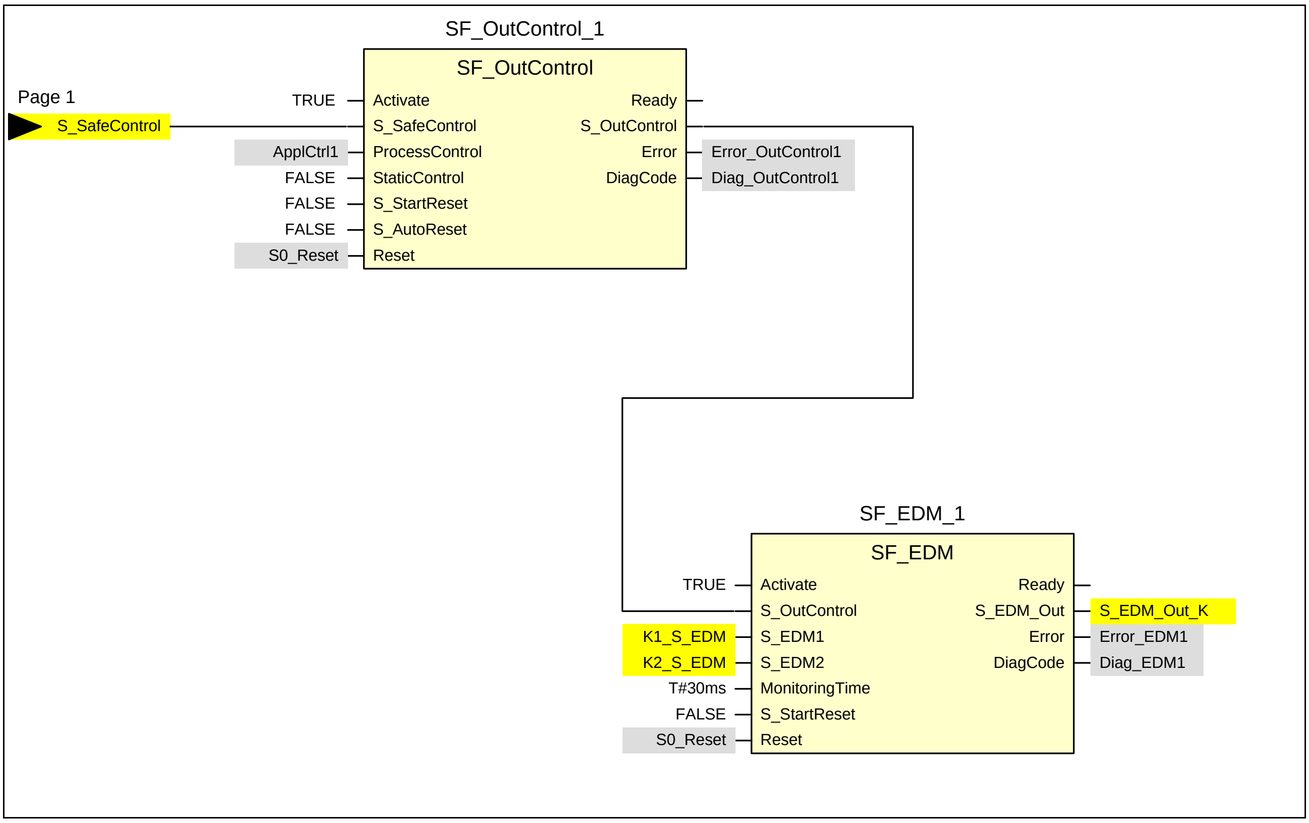


Figure 19 – Access protection at a material gate - Application Program Page 2

4.3.5. Additional Notes

In this example, the two contacts of the guard switch are connected to a safety input device, which realizes the error detection. The resulting SAFEBOOL signal is mapped to the two input channels of the SF_GuardMonitoring_1.

The diagnostic information retrieval has not been covered in this example. For this, refer to Chapter 4.1.5 Additional Notes. The input parameter *Activate* for the dynamic FB activation has been set to TRUE for clarity sake. However, in an application this can be replaced by a variable.

Information about the used Function Block Parameters

Function Block	**Input**	**Constant Value**	**Description**
SF_EmergencyStop_1	S_StartReset	TRUE	Automatic reset allowed when PES is started
	S_AutoReset	FALSE	No automatic reset, user reset/acknowledge necessary
SF_GuardMonitoring_1	S_StartReset	TRUE	Automatic reset allowed when PES is started
	S_AutoReset	FALSE	No automatic reset, user reset/acknowledge necessary
	DiscrepancyTime	T#0ms	The discrepancy time between both safety inputs S_GuardSwitchX is not monitored, because they are identical and since the input unit provides one signal of type SAFEBOOL from the contactors.
SF_MutingSeq_1	S_StartReset	TRUE	Automatic reset allowed when PES is started
	MaxMutingTime	T#30s	The maximum muting time is monitored to be within 30 sec.
SF_LightCurtain_1	S_StartReset	TRUE	Automatic reset allowed when PES is started
	S_AutoReset	FALSE	No automatic reset, user reset/acknowledge necessary
	TestTime	T#100ms	The maximum test time is monitored to be within 100 msec
	NoExternalTest	TRUE	The external manual sensor test is not supported.
SF_OutControl_1	S_StartReset	FALSE	No automatic reset allowed when PES is started
	S_AutoReset	FALSE	No automatic reset, user reset/acknowledge necessary
	StaticControl	FALSE	A dynamic change of the signal ApplCtrl1 (rising edge) is required after block activation or a triggered safety function (S_SafeControl at FALSE).
SF_EDM_Contactor_1	S_StartReset	FALSE	No automatic reset allowed when PES is started
	MonitoringTime	T#30ms	The maximum response time of both the feedback signals S_EDM1 and S_EDM2 are monitored to be within 30 msec.

4.4. Safe Motion with I/O interface

This example describes a machine with a working area where an operator needs access to, e.g. for process diagnosis, set-up activities, or to clear a material jam. The machine is equipped with one safety drive. This example can be compared to example 2 above, however the drive system here does not utilize a hidden interface and is controlled via a discrete interface. The drive system can be moved by the motion controller via the standard command values.

The access to the working area is provided by a monitored guard door with interlock. The interlock is required because in the case the door is opened, a safety stop of the drive system cannot be finalized before the operator could enter the hazardous area. The door can be opened when the drive system was stopped safely; either after an emergency stop or after the operator selected the setup mode.

In an emergency situation the drive systems needs to be stopped in accordance with stop category 1. A mode selector is used to switch the machine between the 'automatic' and the 'set-up' mode. In the mode 'set-up' the drive systems could be enabled, in order to move with a safely limited speed, by using an enabling device. (Note: The command values will be generated by the standard motion controller)
It is assumed that the safety drive system supports the modes safe stop and safe motion, which could be either a safe operational stop or safely limited speed when the enable switch is activated. These states and the enabling are controlled by three binary safety inputs at the drive. For the feedback loop only two binary outputs of the drive are used since it is not necessary to differentiate between the Safe Operational Stop and the Safely Limited Speed. Both states are safe and will be represented by one feedback signal representing the safe state.

4.4.1. Functional description of safety functions

The emergency stop (via SF_EmergencyStop) acts superimposed to all other safety functions and puts the drive systems into a safe standstill (via SF_SafeRequest) in accordance with stop category 1.
After an emergency stop, the restart of the machine is only possible after the emergency button is released and a reset signal is given (via SF_EmergencyStop).

The normal operation of the machine is only possible in the automatic mode (via SF_ModeSelector) and the guard door is closed and locked (via SF_GuardLock).

In the set-up mode (via SF_ModeSelector) the drive system is stopped acc. Stop category 2 (via SF_SafeRequest). Out of safe standstill the drive can be switched with the enabling device into a mode where movement with a safely limited speed is allowed (via SF_EnableSwitch). If the enable device is released the drive performs again a stop category 2.

The guard door can be unlocked and opened after an emergency stop or setup mode is selected, however the drive must be stopped safely first.

4.4.2. Graphical overview of the safety application interface

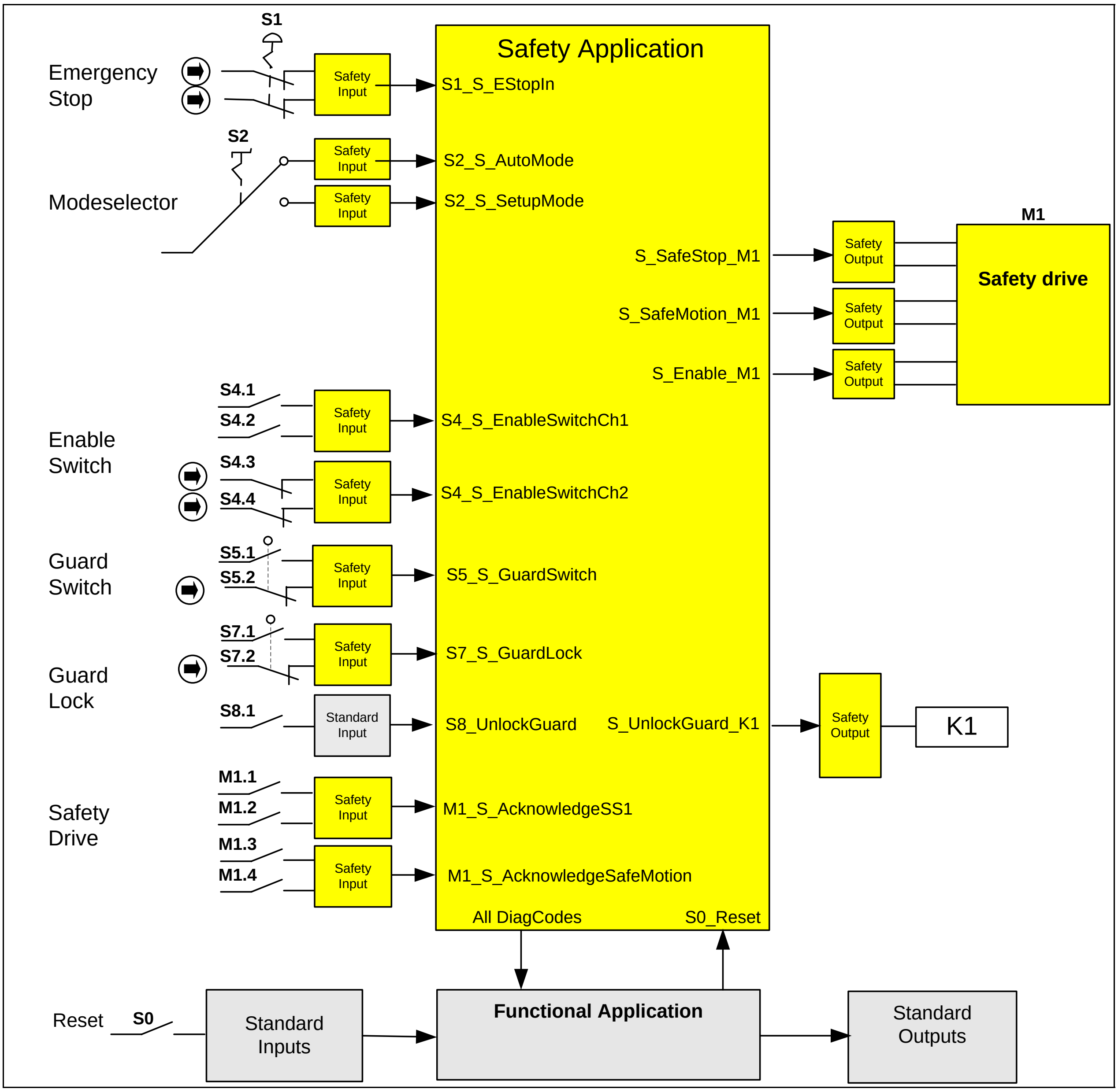


Figure 20 – Graphical overview of the example mode selector

4.4.3. Declaration of the used variables

Inputs

Name	Datatype	Description
S1_S_EStopIn	SAFEBOOL	E-Stop
S2_S_AutoMode	SAFEBOOL	Automatic Mode
S2_S_SetupMode	SAFEBOOL	Setup Mode
S4_S_EnableSwitchCh1	SAFEBOOL	Enabling Device Channel 1
S4_S_EnableSwitchCh2	SAFEBOOL	Enabling Device Channel 2
S5_S_GuardSwitch	SAFEBOOL	Guard Monitoring
S7_S_GuardLock	SAFEBOOL	Guard Lock Monitoring
S0_Reset	BOOL	Reset
S8_UnlockGuard	BOOL	Request to unlock Guard
M1_S_AcknowledgeSS1	SAFEBOOL	Acknowledge of Safe Stop 1
M1_S_AcknowledgeSafeMotion	SAFEBOOL	Acknowledge of Safe Motion

Outputs

Name	Data type	Description
S_SafeStop_M1	SAFEBOOL	Request SafeStop 1
S_SafeMotion_M1	SAFEBOOL	Request for Safe Motion Mode
S_Enable_M1	SAFEBOOL	Enable Signal to allow Safely Limited Speed
S_UnlockGuard_K1	SAFEBOOL	Unlock Guard

Local Variables

Name	Data type	Description
S_SetupMode	SAFEBOOL	Setup Mode
S_AutoMode	SAFEBOOL	Auto Mode enabled
S_SafeStandstill	SAFEBOOL	Drive System performs a safe Standstill
S_SafeMotionActive	SAFEBOOL	Drive Systems performs either an operational stop safely or moves safely slower than the permitted speed.

4.4.4. Program example

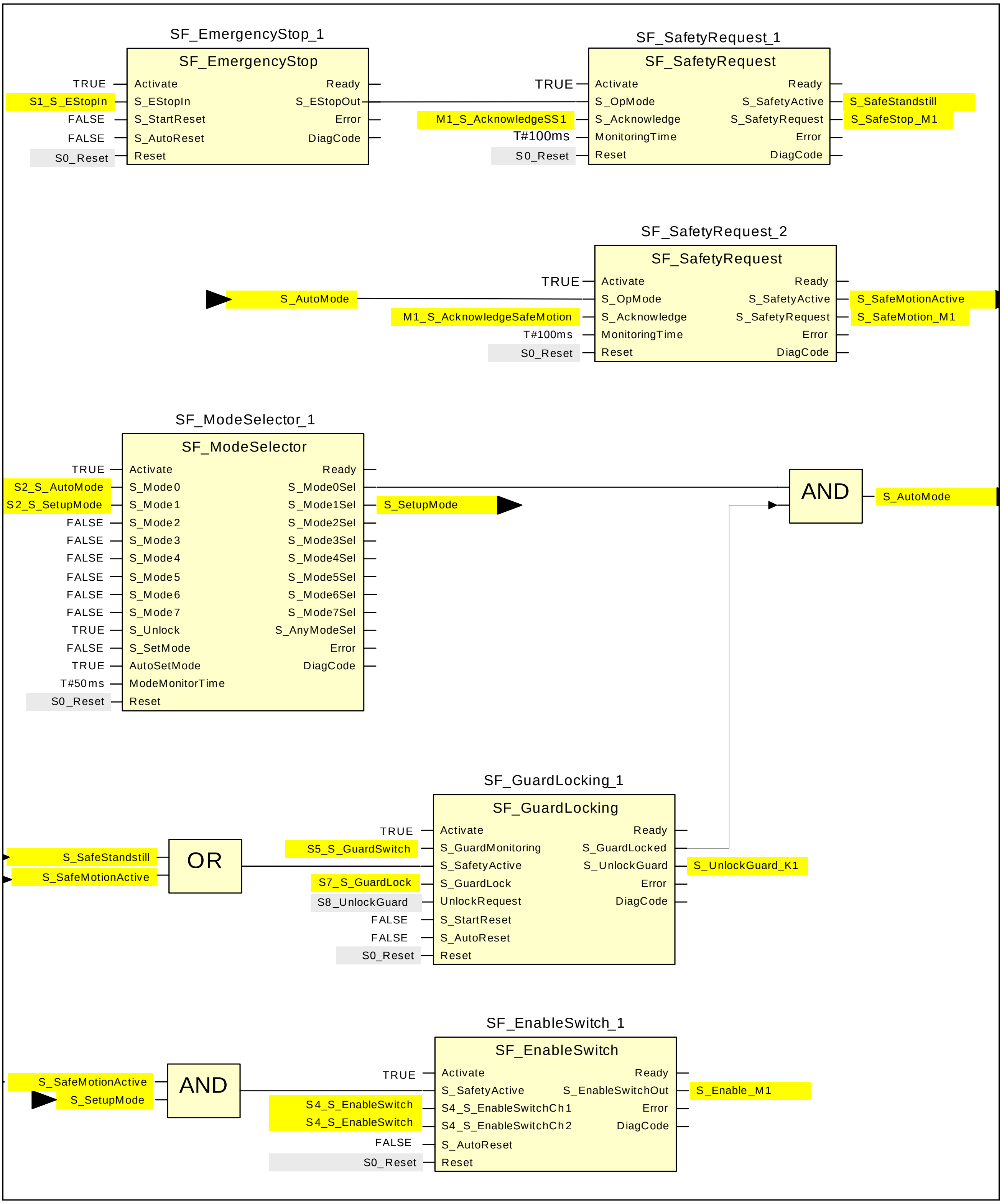


Figure 21 – Application Program mode selector

4.4.5. Additional Notes

In this example, the two contacts of the guard switch are connected to a safety input device, which realizes the error detection. The resulting SAFEBOOL signal is mapped to the input of the SF_GuardLocking_1.

The diagnostic information retrieval has not been covered in this example. For this, refer to Chapter 4.1.5 Additional Notes. The input parameter *Activate* for the dynamic FB activation has been set to TRUE for clarity sake. However, in an application this can be replaced by a variable.

The application does not differentiate between the modes Safe Operational Stop or Safely Limited Speed. In both modes the machine is in a safe state. Therefore the “Enable” signal is connected to the drive without any feedback loop. Otherwise a third SF_SafetyRequest would be necessary.

Information about the used Function Block Parameters

Function Block	**Input**	**Constant Value**	**Description**
SF_EmergencyStop_1	S_StartReset	FALSE	Manual reset when PES is started
	S_AutoReset	FALSE	Manual reset when emergency stop button is released
SF_GuardLocking_1	S_StartReset	FALSE	Manual reset when PES is started
	S_AutoReset	FALSE	Manual reset when Guard is locked
SF_ModeSelector_1	S_Unlock	TRUE	A change is possible and is not frozen
	S_SetMode	FALSE	therefore S_SetMode is not used
	AutoSetMode	TRUE	A change does not need to be confirmed,
	ModeMonitorTime	T#50ms	Maximum time for an ambiguous switch position
SF_EnableSwitch_1	S_AutoReset	FALSE	Avoidance of an unintended restart required no automatic reset allowed
SF_SafetyRequest_1	MonitoringTime	T#100ms	Max. response time between the safety function request (S_OpMode set to FALSE) and the actuator acknowledgment (S_Acknowledge switches to TRUE)
SF_SafetyRequest_2	MonitoringTime	T#100ms	Max. response time between the safety function request (S_OpMode set to FALSE) and the actuator acknowledgment (S_Acknowledge switches to TRUE)

4.5. Two-Hand Control

This example describes a machine where a two-hand control initiates the dangerous movement as long as both push buttons on the two-hand control are pressed and the process provides an enabling signal.
The dangerous movement is initiated by the closing of two subsequent contactors, which are monitored via a feedback loop.

4.5.1. Functional description of safety functions

This example uses the following safety functions:

1. By pushing an emergency stop button all hazardous movements must be stopped. (via SF_EmergencyStop) Emergency stop has the highest priority. After releasing the EStop push button, a reset via S0_Reset is required.
2. By pressing both push buttons of the two-hand control, the safety output is activated. The release of any of the two-hand push buttons disables the safety output and stops the dangerous motion via the contactors K1 and K2. (via SF_TwoHandControlTypeII)
3. The initial state and the operational state of the connected contactors K1 and K2 are monitored and if an error is detected, the safety output cannot become operational. (via SF_EDM)
4. After power on of the safety or functional application, or after an emergency stop condition, the two-hand control must be released and re-operated in order to activate the safety output again (via SF_OutControl). In order to guarantee this for the functional application restart, the Process signal from the functional application is connected to the Activate input of the two hand control function block THC_S2_S3. (If the application process is restarted while the two hand control is activated, the FB goes to the state C0003 signalling an error that both buttons are pressed at the activation, prohibiting a restart.)

In this example, only one operation mode exists.

4.5.2. Graphical overview of the safety application interface

The safety inputs for the two-hand control (S2_S_Switch1 and S3_S_Switch2) are connected to the two-hand control type II.

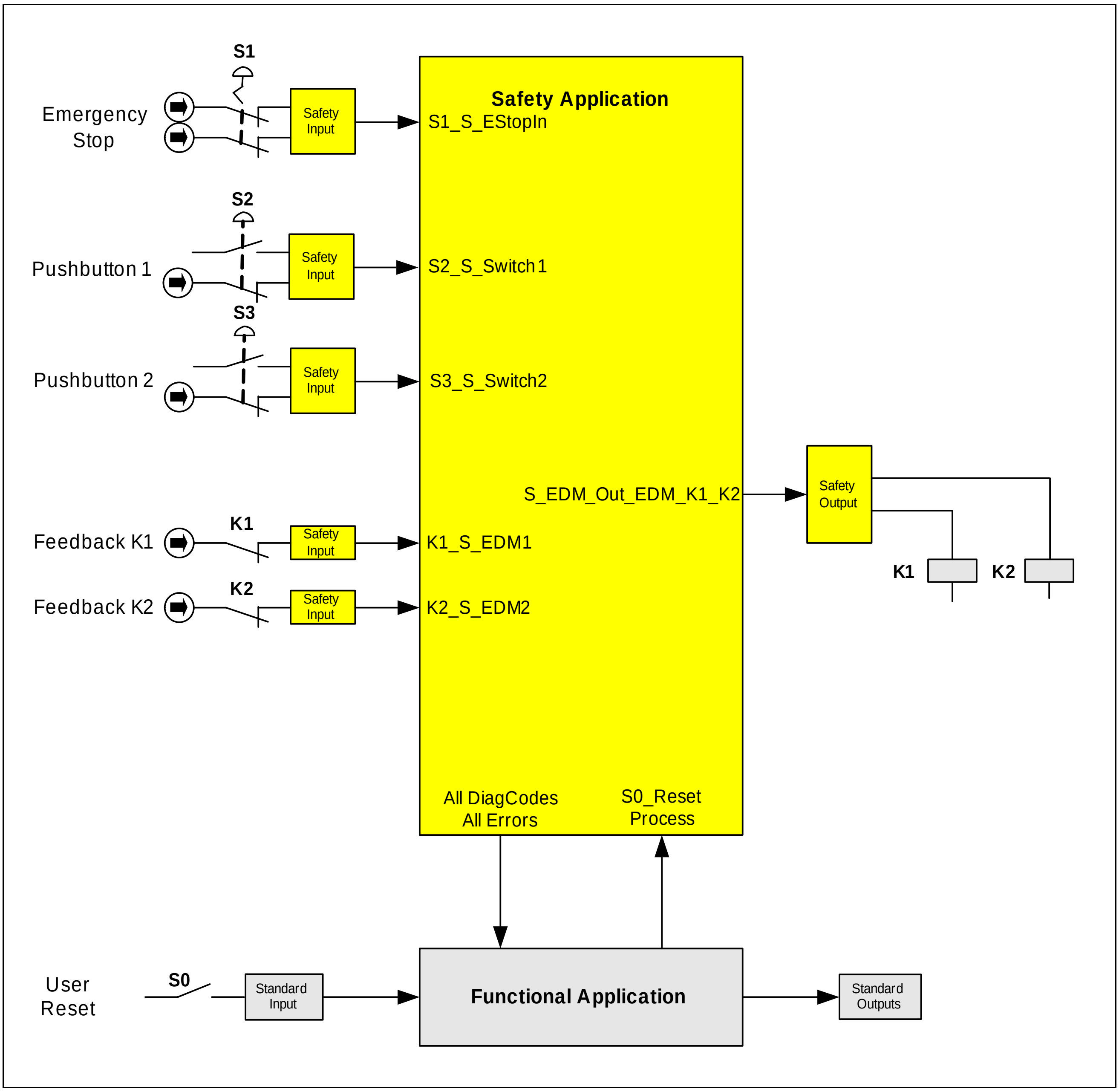


Figure 22 – Graphical overview of the example TwoHand Control with EDM

4.5.3. Declaration of the used variables

Inputs

Name	Datatype	Description
S1_S_EStopIn	SAFEBOOL	Emergency stop button S1
S2_S_Switch1	SAFEBOOL	Switch S2 related to push button 1 of two hand control
S3_S_Switch2	SAFEBOOL	Switch S3 related to push button 2 of two hand control
K1_S_EDM1	SAFEBOOL	Feedback external device K1
K2_S_EDM2	SAFEBOOL	Feedback external device K2
S0_Reset	BOOL	Reset by user via Switch S0 (derived from functional application)
Process	BOOL	Enabling motion by the process (derived from functional application)

Outputs

Name	Datatype	Description
S_EDM_Out_EDM_K1_K2	SAFEBOOL	Drives actuator via K1 and K2
All Errors	BOOL	Represents all error bools of the used FB (connected to functional application)
All DiagCodes	WORD	Represents all diagnostic codes of the used FB (connected to functional application)

4.5.4. Program example

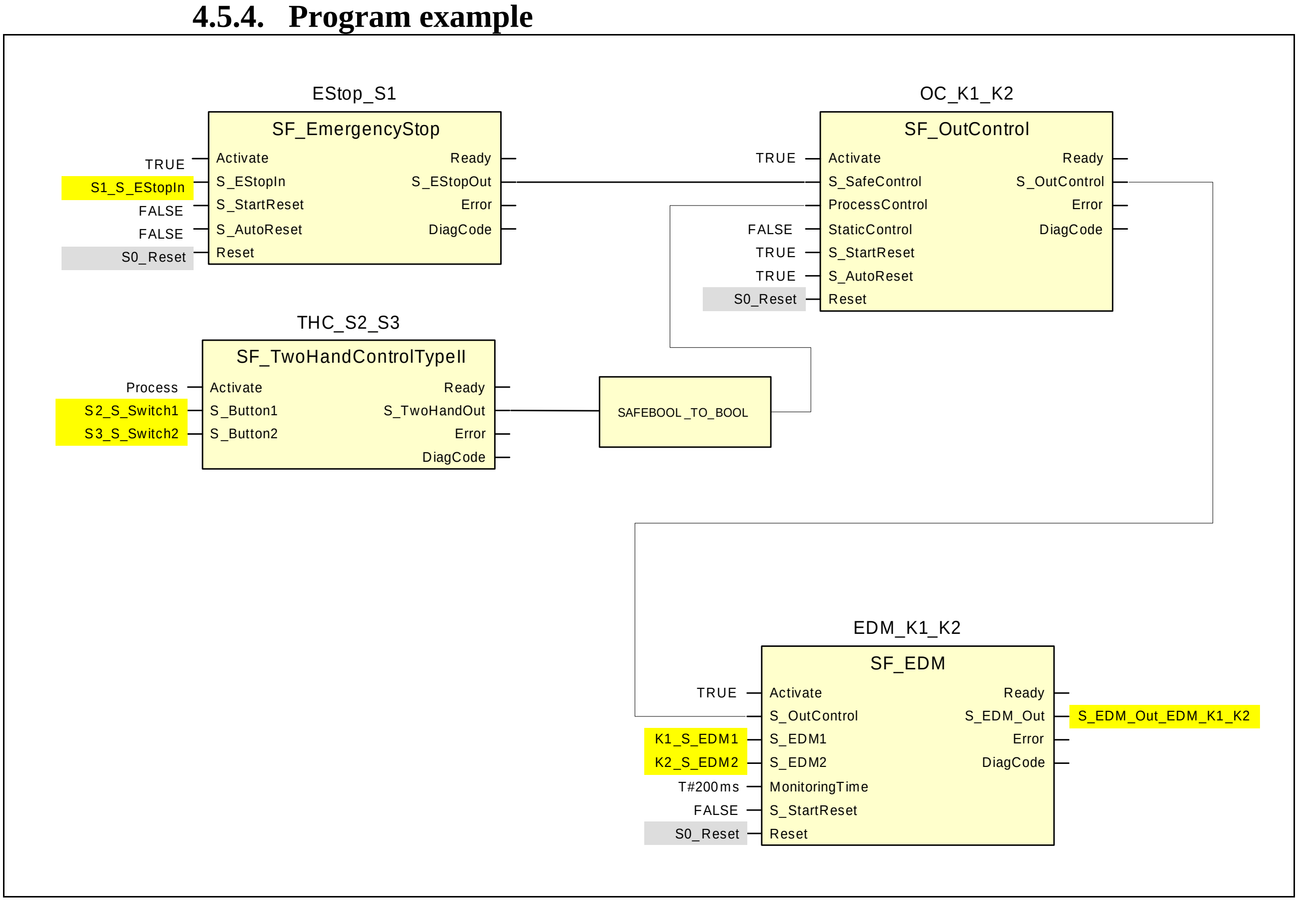


Figure 23 – Application Program TwoHand Control with EDM

4.5.5. Additional Notes

This example can also be used with the SF_TwoHandControlTypeIII.
The diagnostic information retrieval has not been covered in this example. For this, refer to Chapter 4.1.5 Additional Notes.The input Activate has been set to TRUE via its input for clarity sake. However, in an application this can be replaced by a variable.

Information about the used Function Block Parameters

Function Block	**Input**	**Constant Value**	**Description**
EStop_S1	S_StartReset	FALSE	No automatic reset when PES is started
	S_AutoReset	FALSE	No automatic reset, user reset/acknowledge necessary
OC_K1_K2	S_StartReset	TRUE	Automatic reset allowed when PES is started
	S_AutoReset	TRUE	Automatic reset, no user reset/acknowledge necessary
	StaticControl	FALSE	A dynamic change of the signal Appl_Control (rising edge) is required after block activation or a triggered safety function (S_SafeControl at FALSE)
EDM_K1_K2	S_StartReset	FALSE	No automatic reset when PES is started
	MonitoringTime	T#200ms	The maximum response time of both the feedback signals S_EDM1 and S_EDM2 are monitored to be within 200 msec.