Specification of Limit Values for Safely Limited Speed (SLS) from a non-safety HMI

Distributed Safety

Application Description • March 2013

Applications & Tools

Answers for industry.

SIEMENS

Siemens Industry Online Support

This document is taken from Siemens Industry Online Support. The following link takes you directly to the download page of this document:

http://support.automation.siemens.com/WW/view/en/67634251

Caution:
The functions and solutions described in this entry are mainly limited to the realization of the automation task. In addition, please note that suitable security measures in compliance with the applicable Industrial Security standards must be taken if your system is interconnected with other parts of the plant, the company's network or the Internet. More information can be found under entry ID 50203404.

http://support.automation.siemens.com/WW/view/en/50203404

SIEMENS

SIMATIC
SLS over HMI

Warranty and Liability

Note

The application examples are not binding and do not claim to be complete regarding configuration, equipment and any eventuality. The application examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible for ensuring that the described products are used correctly. These application examples do not relieve you of your responsibility to use sound practices in application, installation, operation and maintenance. When using these application examples, you recognize that we will not be liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these application examples at any time without prior notice. If there are any deviations between the recommendations provided in this application example and other Siemens publications – e.g. Catalogs – the contents of the other documents have priority.

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

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

Table of Contents

1 Task

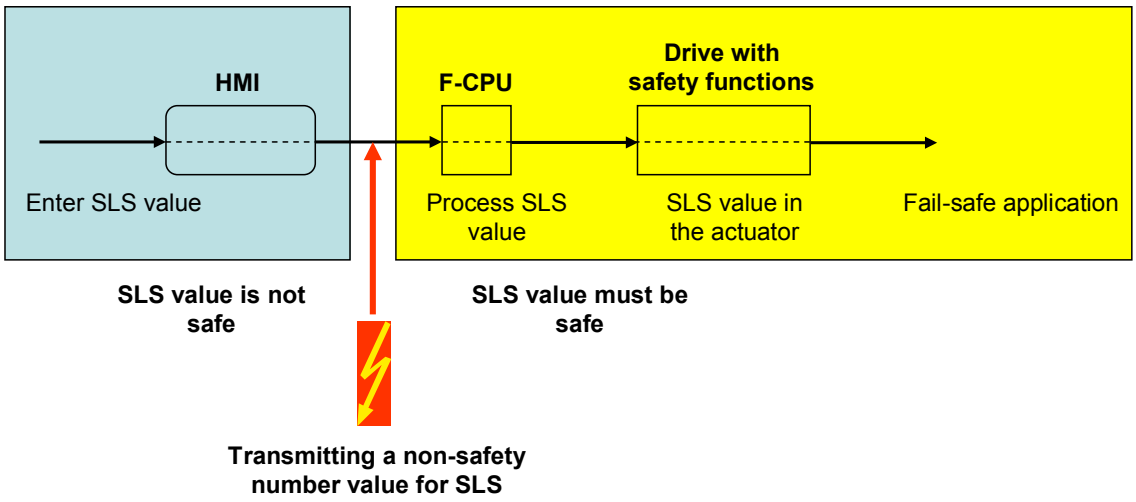
Introduction

Human Machine Interfaces (HMIs) are convenient and essential components in the practice of everyday industrial use. In order to use this convenience with operating and controlling processes and plants in the fail-safe applications as well, additional measures are required.

The application example on hand shows you how a non-safety HMI is directly involved in the safety function of an application.

Overview of the automation task

The figure below provides an overview of the automation task.



Description of the automation task

In a drive with safety functions, a number value (SLS value) shall be written as a limit for a safe speed (**S**afely **L**imited **S**peed (SLS)). During later operation of the drive, this means: when requesting the SLS, the actual speed value is compared with the SLS value. If the actual speed value exceeds the SLS value, the drive switches off safely.

The SLS value shall be specified via an HMI and reach the drive via an F-CPU. F-CPU and the drive with safety functions are certified safety components, i.e. an SLS value in the F-CPU safely reaches the drive with safety functions.

However, the F-CPU does not receive the SLS value via a non-safety HMI. This means, due to data corruption on the path from the HMI to the F-CPU an undesired SLS value might get to the F-CPU and subsequently reach the drive, which may cause hazardous situations.

The core task of this application example can be formulated as follows:

What could a safety concept look like which ensures that an SLS value transferred from HMI to the F-CPU is identical with the SLS value transmitted via a non-safety HMI?

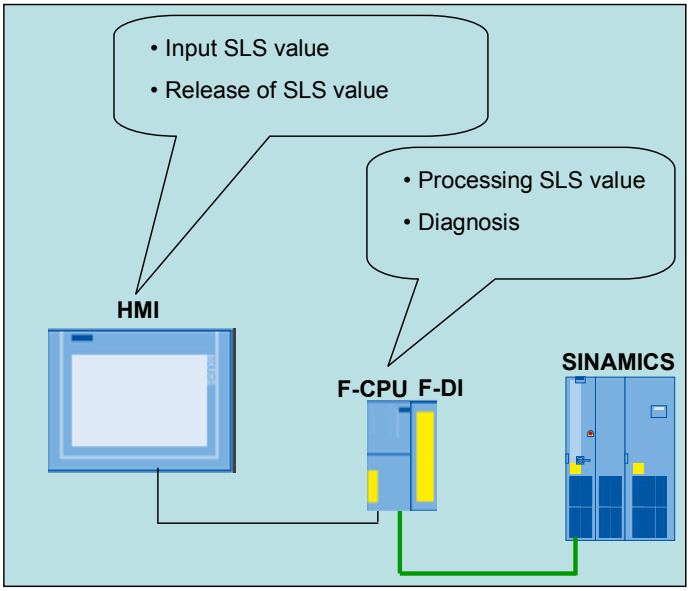
This application example on hand explains such a safety concept.

2 Solution

2.1 Overview of the general solution

Schematic layout

The following figure gives a schematic overview of the most important hardware components of the solution:



Advantages

The solution presented here offers you the following advantages:

- Fail-safe configuration data for the drive can be transferred user-friendly via non-safety HMI
- Safety concept can also be adopted for other tasks

Topics not covered by this application

The measures described in this document with regards to mastering the effects of errors due to the parameter transmission process, end with the presence of the SLS value transmitted by the HMI in the F program, after the plausibility check has been completed with a positive result (no data corruption detected).

Processing the SLS value in the drive (SINAMICS) is not further discussed in this application example. The following applies here:
If the correct SLS value exists safely in the F-CPU, it can also reach the drive safely via PROFIsafe mechanisms.

Assumed knowledge

The following basic knowledge is assumed:

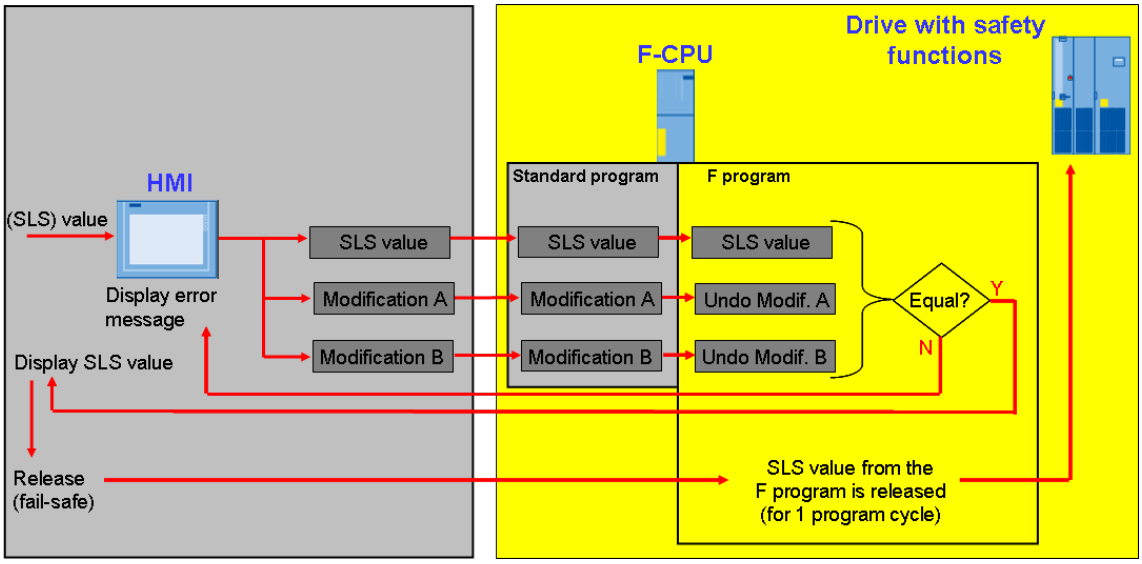
- STEP 7
- Distributed Safety
- WinCC flexible

2.2 Description of the core functionality

2.2.1 Overview screen and description

Overview

The following overview diagram shows in which hardware component a certain functionality is realized.



Description of the overview screen

The SLS value entered by the operator at the HMI is modified in WinCC flexible on purpose. The modification produces two additional values on top of the SLS value, which reach the standard program of the F-CPU via an interface DB. These modified values are used for diagnosis to exclude data corruption.

The three values (SLS value, modification A, modification B) are transferred to the F program via flags, where the modifications from WinCC flexible can be reversed. Subsequently, these three values must be equal, otherwise data corruption has occurred.

In the case of data corruption this information is displayed at the HMI. Only if the three values are equal (correct data transmission), is the SLS value displayed at the HMI to the user in order for it to be acknowledged once again. Acknowledgement at the HMI occurs fail-safe.

For one program cycle, the acknowledgement at the HMI by the user ensures that the SLS value, which has proven correct, is safely written to an area which represents the interface to the drive (here: a certain address in the F program). The mechanisms which bring the SLS value from there to the drive are all together fail-safe (PROFIsafe).

The mechanisms described here are discussed in detail in the chapter “Function mechanisms of this application” regarding the realization.

2.2.2 Description and overview of the user interface

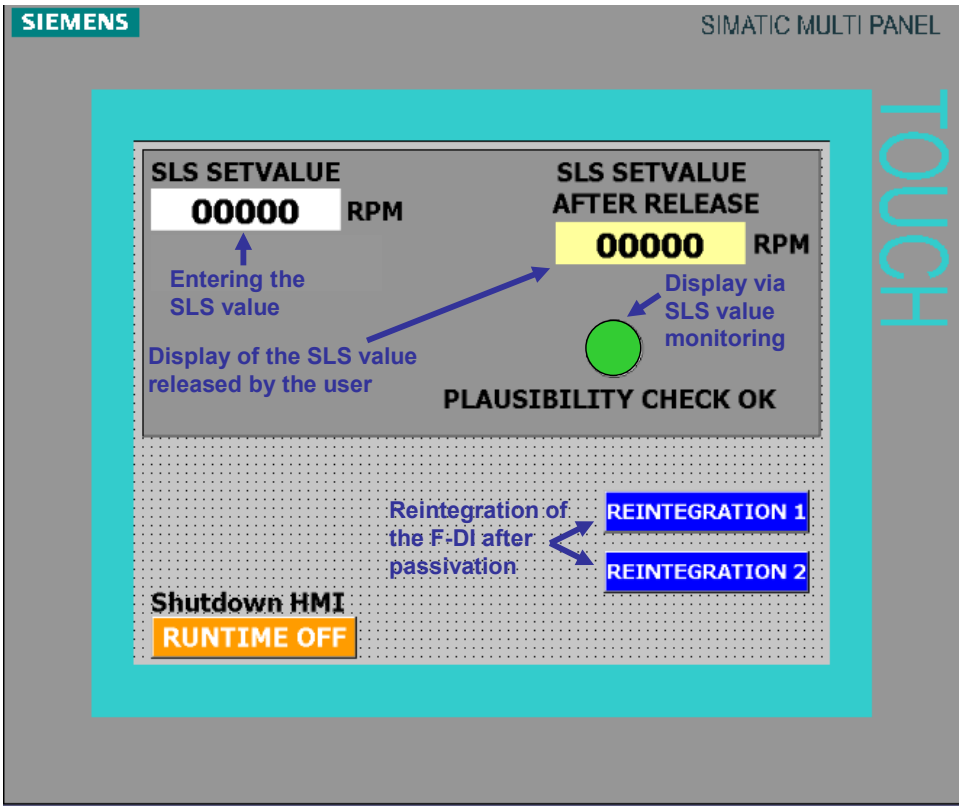
The visualization user interface configured with WinCC flexible consists of two screens. Both screens are necessary for the operation. Switchover between the screens is performed automatically.

The operation at the HMI contains the following:

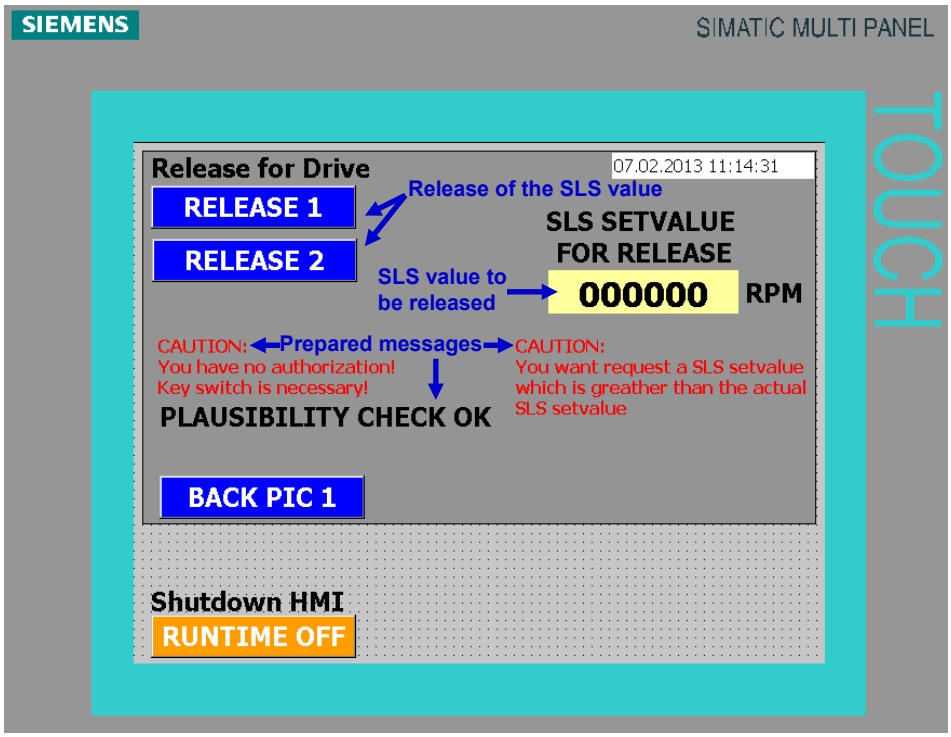
- **Screen 1**
  - Manual input of a speed value for the safe velocity (SLS value)
- **Screen 2**
  - Releasing the speed value displayed at the HMI for the safe velocity (SLS value)

Note Button RELEASE 2 will be shown after pressing button RELEASE 1.

Screen 1 of the visualization user interface



Screen 2 of the visualization user interface



2.3 Hardware and software components used

The application document was generated using the following components:

Hardware components

Component	Qty.	MLFB/order number	Note
Power Supply	1	6S7307-1EA00-0AA0	Or similar hardware components
CPU 317F-2 PN/DP Firmware V3.1	1	6ES7317-2FK14-0AB0	
F-DI 24 X DC 24V	1	6ES7326-1BK02-0AB0	
MULTI PANEL MP 277 10” Touch Image Version: 1.1.3.0 Bootloader Version: 1.01	1	6AV6643-0CD01-1AX0	
Key switch (1S) snaps into place	1		

Standard software components

Component	Qty.	MLFB/order number	Note
SIMATIC STEP 7 V5.5	1	6ES7810-4CC10-0YA5	Floating license for 1 user
SIMATIC Distributed Safety V5.4 + SP5	1	6ES7833-1FC02-0YA5	Floating license for 1 user
S7 F Configuration Pack V5.5 + SP8	1		Is part of SIMATIC Distributed Safety
SIMATIC WinCC flexible 2008 SP2	1	6AV6613-0AA51-3CA5	Floating License For the configuration of SIMATIC panels and WinCC flexible 2008 Runtime

Sample files and projects

The following list includes all files and projects used in this example.

Component	Note
67634251_sls_over_hmi.zip	This zip file contains the STEP 7 and WinCC flexible project.
67634251_ sls_over_hmi_en.pdf	This document

2.4 Achievable SIL or PL

The safety concept described here is suitable for reaching SIL 2 according to IEC 62061 or a PL d according to ISO 13849-1.

3 Functional Mechanisms of this Application

3.1 Program overview

Below, you are provided with an overview of the STEP 7 program.

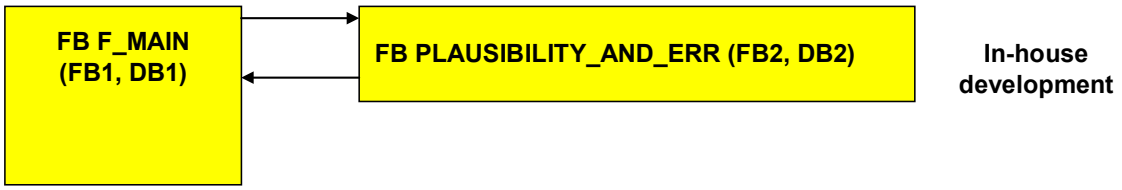
Standard program

The standard program basically consists of the following blocks:

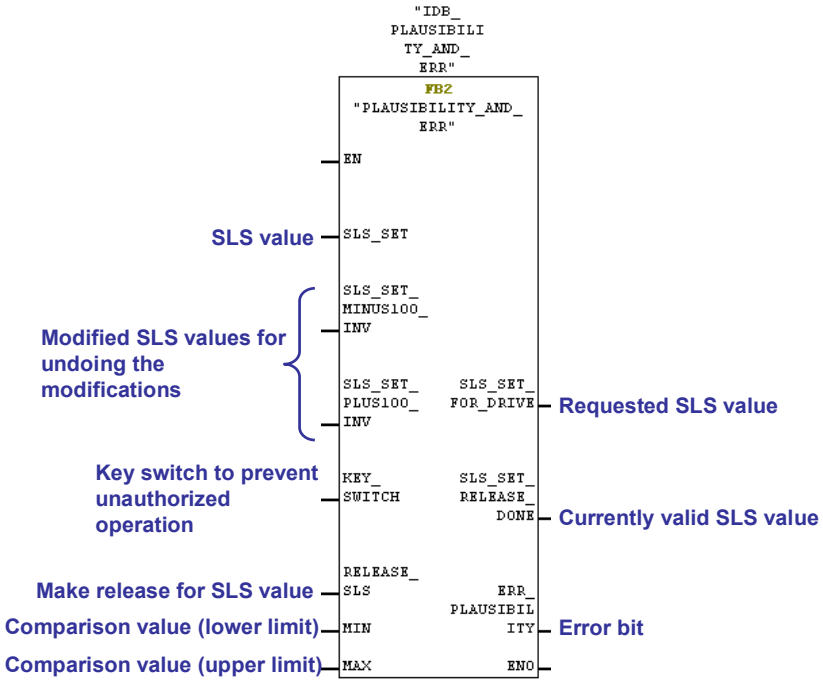
- OB100: preset
- OB35: call of F-CALL
- OB82, OB83, OB85, OB86, OB121, OB122: error-OBs, which generally should be downloaded into a SIMATIC CPU.

F program

The F-CALL block calls FB F_MAIN (FB1, DB1) as F program block, which in return calls FB PLAUSIBILITY_AND_ERR (FB2, DB2), which from a program point of view represents the main component for realizing the safety concept.



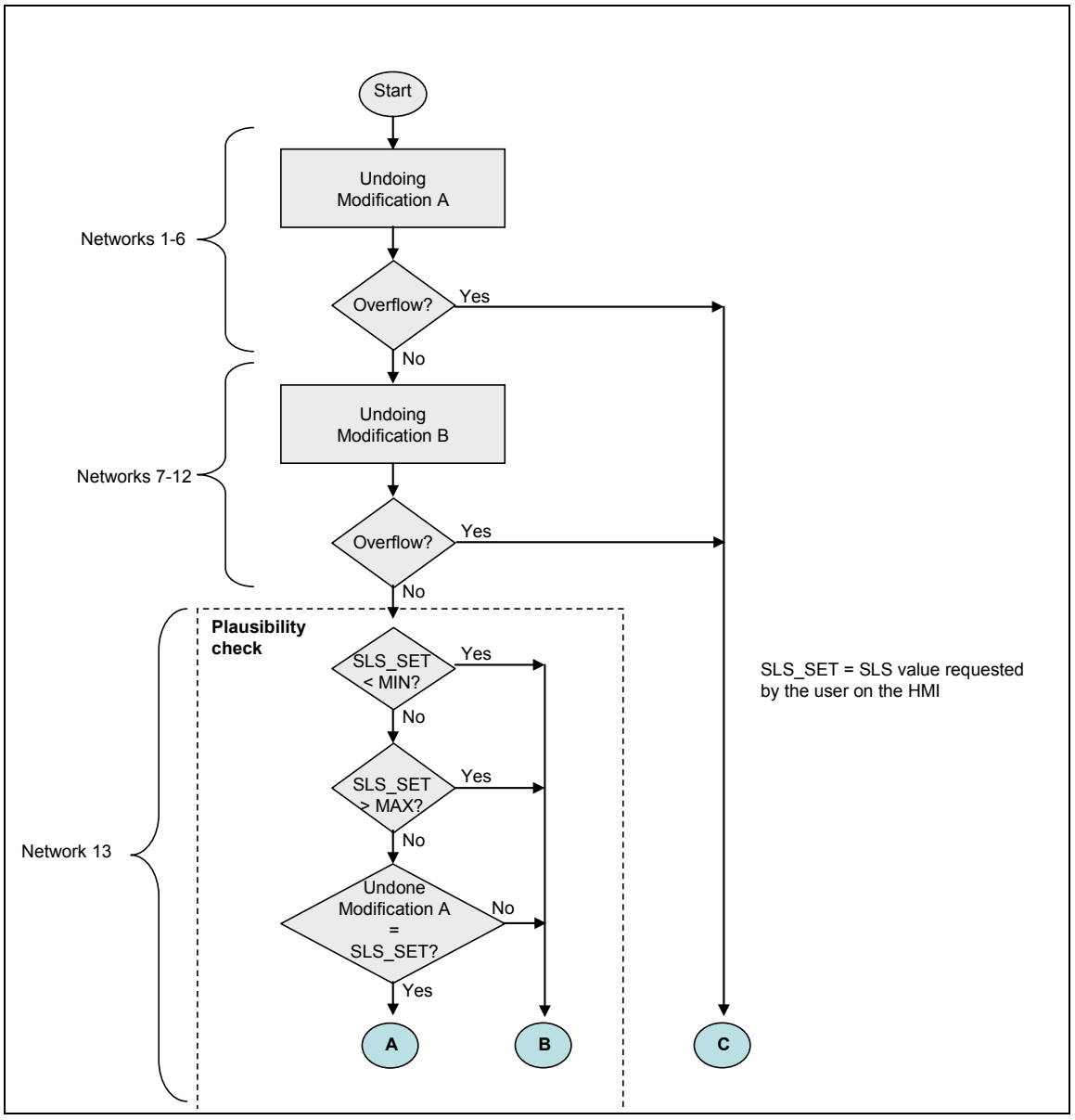
FB PLAUSIBILITY_AND_ERR (FB2, DB2)

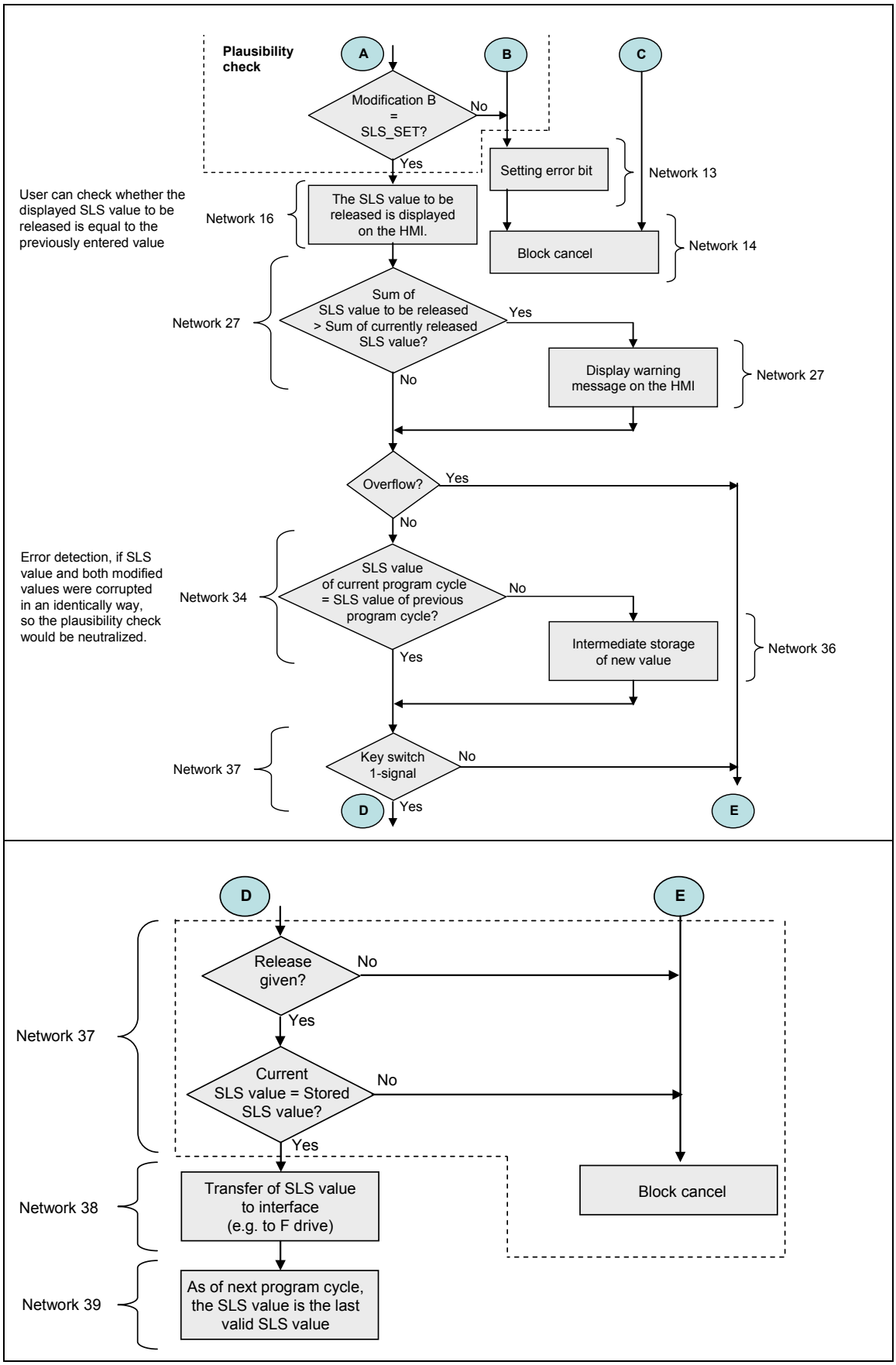


3.2 Concept of FB PLAUSIBILITY_AND_ERR (FB2, DB2)

The flow chart illustrates the solution concept of FB PLAUSIBILITY_AND_ERR (FB2, DB2).

Only if the signal of the key switch is pending (KEY_SWITCH = 1) **and** the SLS value displayed at the HMI has been released (RELEASE_SLS = 1) **and** the plausibility check has not detected an error, is FB2 not cancelled (RET = 0), i.e. a program sequence is run through for one program cycle which offers the released SLS value to the drive.





3.3 Access protection

F-CPU access protection

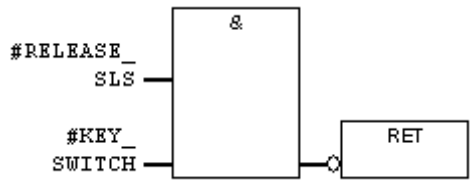
The F-CPU and the access to the F program require a password. The prepared password is: **siemens**

Access protection at the HMI

Only authorized personnel must specify SLS values at the HMI. To ensure this, a key switch is required for the operation. The signal of the key switch is assigned as 1-channel to an F-DI. In the F program it is checked, whether the key switch signal exists:

Signal of the key switch	Evaluation in the F program	Response
0	There is **no** authorization for transferring an SLS value.	**F program:** Entered SLS value is not processed. **HMI:** Display information that no authorization exists.
1	There is an authorization for transferring an SLS value.	SLS value displayed at the HMI can be released.

The statements of the above table are realized in FB PLAUSIBILITY_AND_ERR (FB2):



3.4 Passivation and reintegration of the F-DI

Consequences of passivation of the F-DI

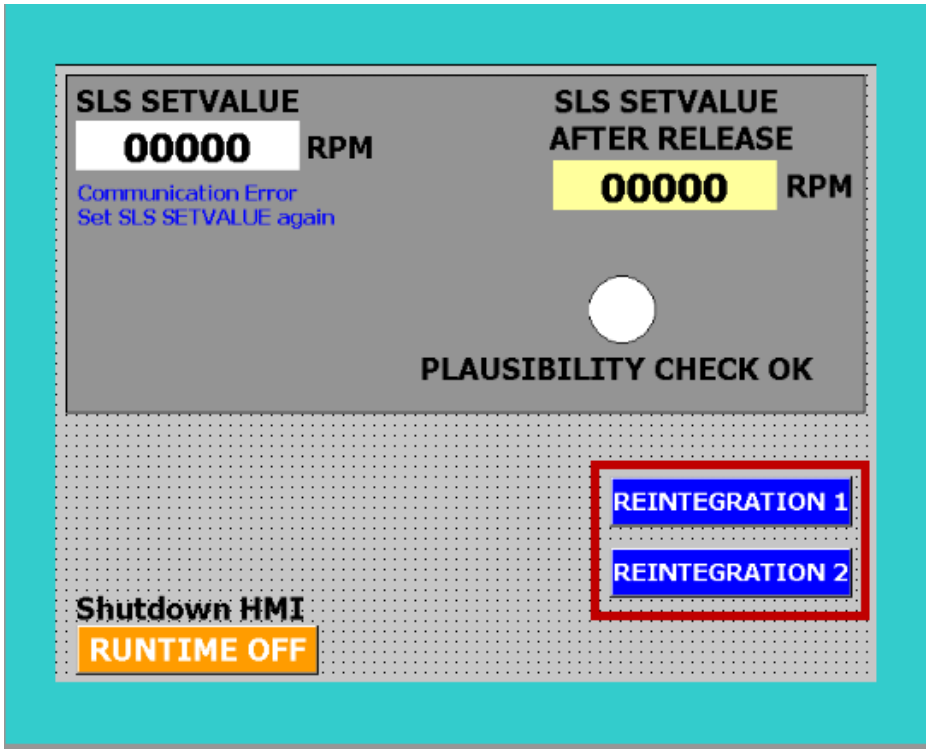
In the case of a passivation of the F-DI, it goes to a fail-safe state. All inputs of the F-DI come with signal “0”. Since the signal of the key is pending at the F-DI, this signal is also “0”, therefore, no SLS values can be specified since a “0” signal of the key switch is interpreted as missing authorization.

When the cause of the passivation of the F-DI is removed, the F-DI can be reintegrated (or depassivated).

Measures for reintegration

The requirement for reintegrating the F-DI, is removing the error that caused the passivation.

In order to perform the reintegration, the buttons at HMI “REINTEGRATION 1” and “REINTEGRATION 2” must be pressed in succession.



After pressing the REINTEGRATION 1 button, you need to wait at least 1 second, however, no longer than 1 minute before pressing REINTEGRATION 2. Otherwise, the F-DI remains passivated.

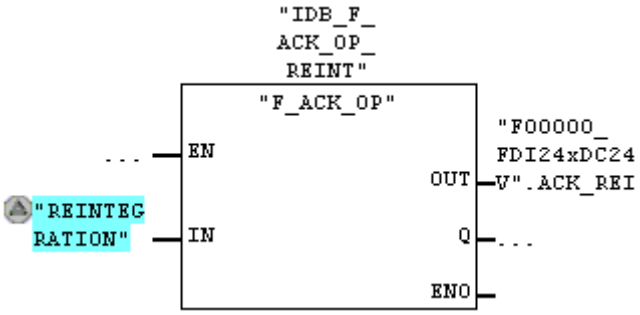
Realizing the reintegration

The section below describes how the reintegration was realized.

The F-DI requires a positive edge at ACK_REI to be reintegrated. This was realized by using the F_ACK_OP from the block library of Distributed Safety. Using FB F_ACK_OP an acknowledgement can be performed fail-safe from the HMI.

Pressing the “REINTEGRATION1” button writes a 6 to FB ACK_OP, pressing the “REINTEGRATION 2” button a 9. If the 6 and the 9 arrive within the specified time, output OUT is set to “1” for one program cycle.

The bit of F-DI ACK_REI is located at output OUT. With the rising edge (ACK_REI = 1), the F-DI is reintegrated and supplies process values again.



WARNING	**In your application, ensure that a reintegration does not automatically cause hazardous machines or other applications to start up.**

3.5 Explanation of the safety concept

What will you learn here?

The following section discusses the underlying safety concept. For a better overview, we follow the overview diagram in chapter 2.2.1.

It is the aim of this chapter to outline the realization of the safety concept applied here, which allows a non-safety HMI to participate in the safety function.

3.5.1 From the input at the HMI to the F program

SLS value and modified values

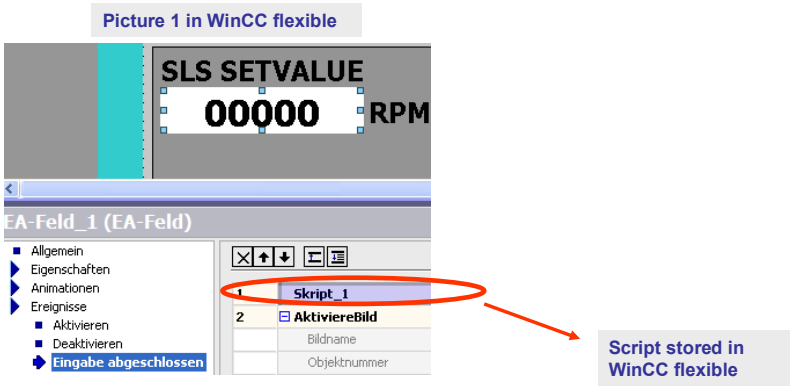
Via the input field of the HMI, the user enters the desired SLS value. As a safety supporting measure, two additional values supplied with an offset are generated from the SLS value.

The offset is later cancelled (in the F program) and it is checked whether the expected values exist.

Initially, three (non-safety) values exist:
The SLS value and the two resulting modified values with offset. All three values are later compared with each other in the F program, which will uncover any theoretically possible data corruption.

How are the two values modified?

After the completed input of the SLS value at the HMI, a script in WINCC flexible runs in the background. In this script, the two modified values are generated.



Modification A

The first value modified from the SLS value is formed by

- subtraction of 100 from the SLS value
- subsequent inversion of all bits of this integer value

Modification B

The second value modified from the SLS value is formed by

- adding 100 to the SLS value
- subsequent inversion of all bits of this integer value

Modification in the script of WinCC flexible

```
12 SmartTags("HMI_INTERFACE.SLS_SET")=SmartTags("INTERN_SLS_VAL")
13 SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV")=SmartTags("INTERN_SLS_VAL")-100
14 SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV")=SmartTags("INTERN_SLS_VAL")+100
15 ActivateScreen "PIC2", 0
16
17 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 0
18 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 1
19 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 2
20 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 3
21 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 4
22 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 5
23 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 6
24 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 7
25 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 8
26 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 9
27 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 10
28 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 11
29 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 12
30 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 13
31 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 14
32 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_MINUS100_INV"), 15
33
34 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 0
35 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 1
36 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 2
37 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 3
38 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 4
39 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 5
40 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 6
41 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 7
42 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 8
43 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 9
44 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 10
45 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 11
46 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 12
47 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 13
48 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 14
49 InvertBitInTag SmartTags("HMI_INTERFACE.SLS_SET_PLUS100_INV"), 15
```

SLS value

Modification A (subtraction)

Modification B (addition)

Modification A (inverting the bit)

Modification B (inverting the bit)

How do the three values get into the F program?

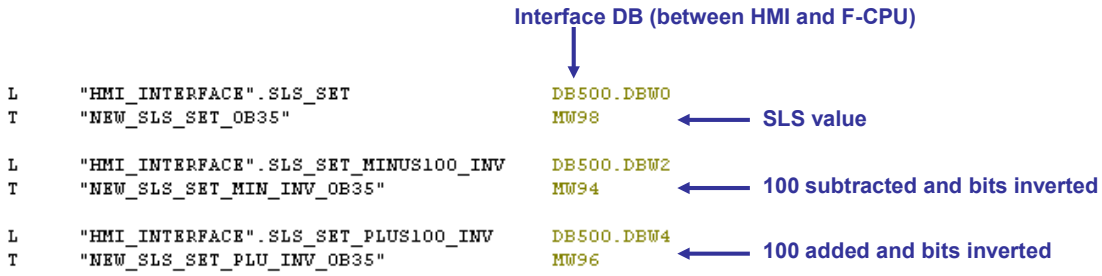
The interface from the HMI to the F-CPU is formed by data block DB HMI INTERFACE (DB500), in this document also referred to as interface DB. The tree values are hence located in the standard program of the F-CPU.

Adresse	Name	Typ
0.0		STRUCT
+0.0	SLS_SET	INT
+2.0	SLS_SET_MINUS100_INV	INT
+4.0	SLS_SET_PLUS100_INV	INT
=6.0		END_STRUCT

SLS value

Modified values

The transition from the standard program to the F program occurs, as usual, via flags in OB35 prior to calling the F-CALL (see picture below).



The tree values are now located in the F program. To have the SLS value evaluated as safe, a plausibility check is performed below.

3.5.2 Check for plausibility (undoing the modification)

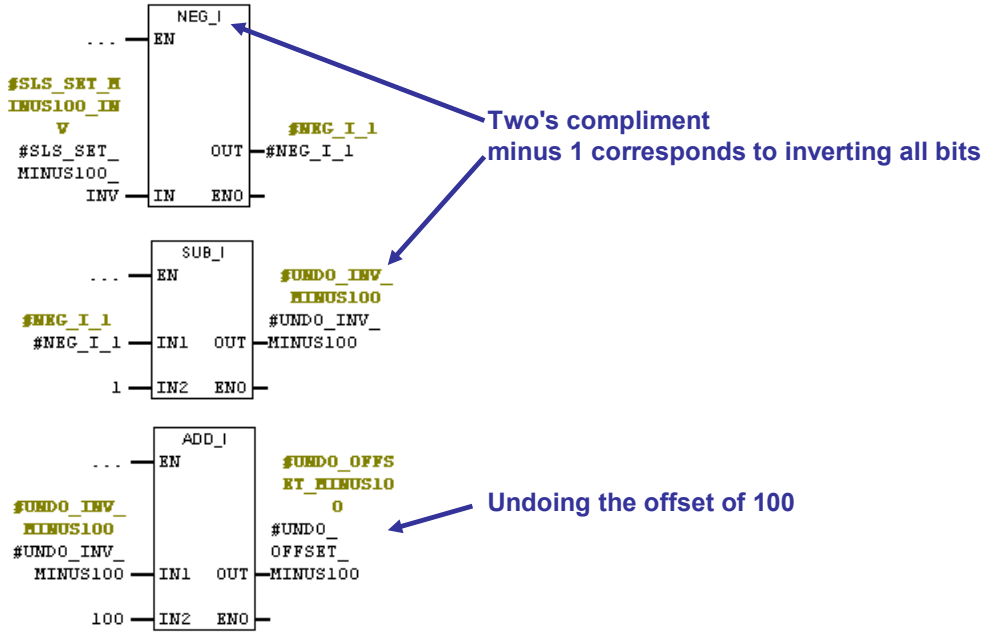
The check for plausibility (undoing the modification) consists of undoing the modification and subsequent comparison for equality with the SLS value. The check is performed in the F program.

Undoing the modification

In FB PLAUSIBILITY_AND_ERR (FB2, DB2) of the F program, the modification is undone (next picture).

Undoing the modification consists of

- inverting all bits of the modified integer value
- undoing the offset by addition or subtraction with 100.



Note

For the operations displayed in the picture (NEG_I, SUB_I, ADD_I) an overflow may occur theoretically, which is safely prevented in the F program by means of a query after each of these blocks. This query is not listed in the picture for a clearer overview.

An overflow causes an error bit to be set.

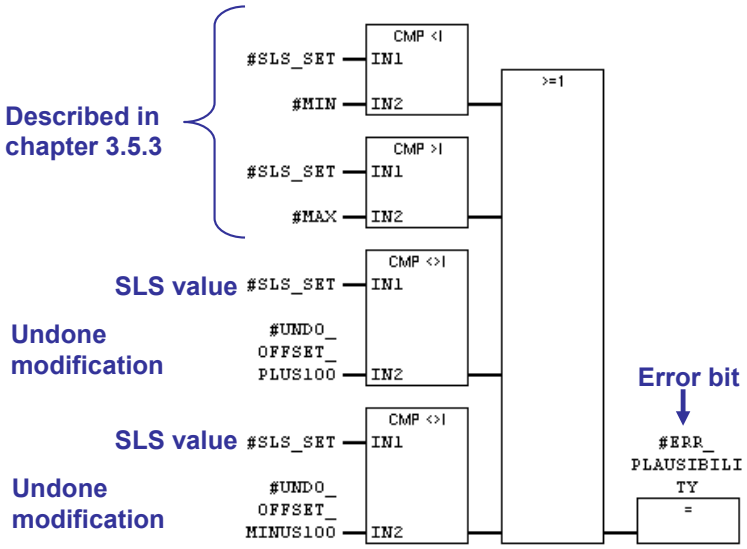
Note

Undoing the offset of 100 in the picture above is shown for the modified value for which 100 was subtracted from the SLS value in WinCC flexible.

For the second modified value for which 100 was added to the SLS value in WinCC flexible, SUB_I is used instead of ADD_I at the location shown in the picture (see code in the F program).

Comparison for equality

In the error-free case, undoing the modification means that both respective values are equal to the SLS value. This plausibility check is performed below:



In the case of an error (ERR_PLAUSIBILITY=1), the FB is cancelled at this point. As a result, no faulty value is displayed to the user on the HMI for release. An accidental release is therefore not even possible.

WARNING

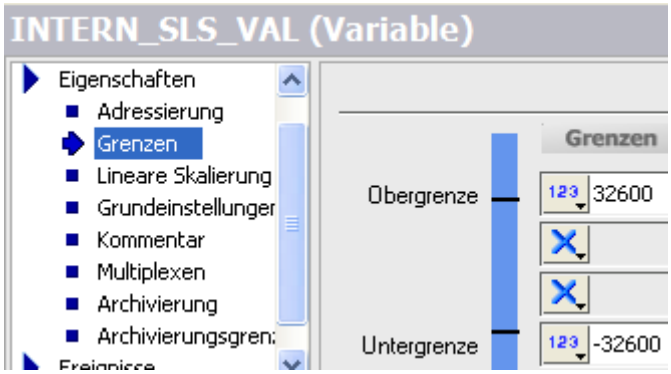
At a going error (ERR_PLAUSIBILITY changes from “1” to “0”) an acknowledgement is not necessary in this example. For applications beyond the scope of this application example, you need to reassess whether this causes possible hazards.

3.5.3 Check for plausibility (comparison for MIN/MAX)

The check for plausibility (comparison for MIN/MAX at the FB PLAUSIBILITY_AND_ERR (FB2, DB2)) consists of the comparison of the desired SLS value with an upper and lower limit. The check is performed in the F program.

Number range for MIN/MAX

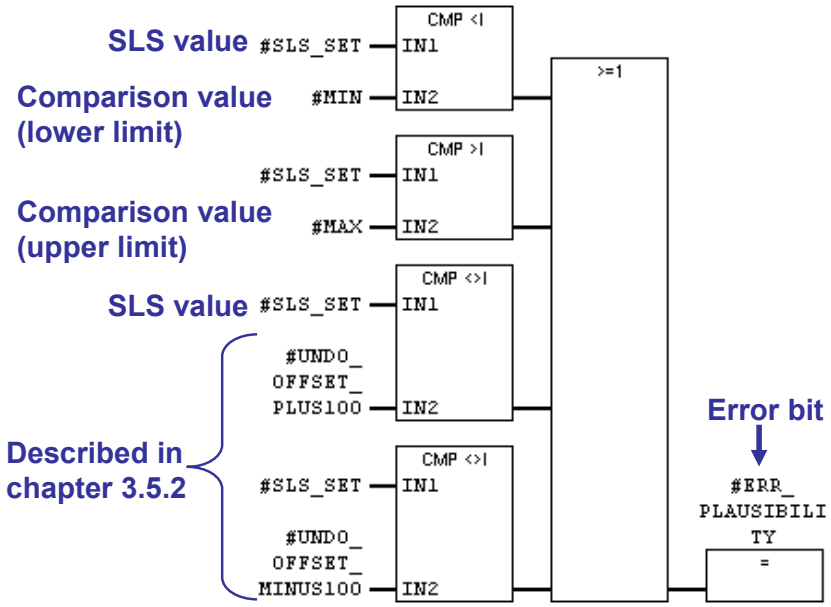
You can use a number range from -32600 to +32600 for MIN/MAX. The limits are already stored in WinCC flexible.



The MIN-/MAX-values self will only be given as input-parameters at the FB PLAUSIBILITY_AND_ERR (FB2, DB 2) in the F-program.

Comparison for MIN/MAX

The desired SLS value must be larger (or equal) than a MIN comparison value and smaller (or equal) than a MAX comparison value. If this condition is not fulfilled, the error bit is set (ERR_PLAUSIBILITY=1).



For ERR_PLAUSIBILITY=1 the FB is cancelled at this point. As a result, no faulty value is displayed to the user on the HMI for release. An accidental release is therefore not even possible.

WARNING	**At a going error (ERR_PLAUSIBILITY changes from "1" to "0") an acknowledgement is not necessary in this example. For applications beyond the scope of this application example, you need to reassess whether this causes possible hazards.**

Note

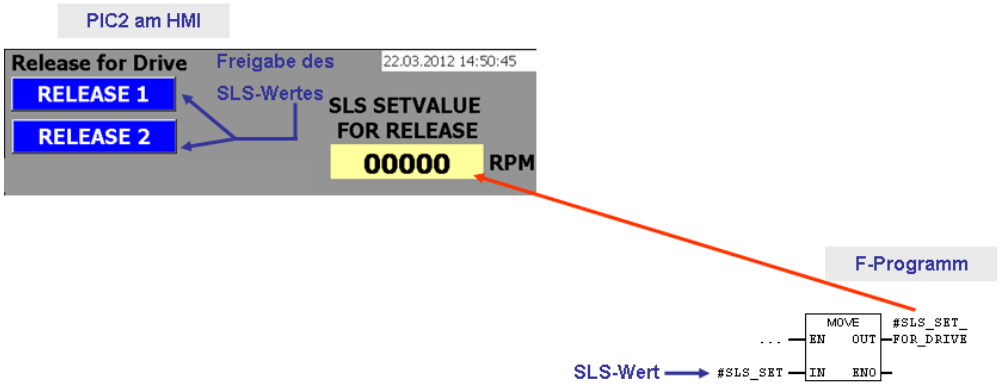
Please note that MIN and MAX are signed number values; the absolute values are not compared!

If SLS_SET shall lie between -100 and -500, the following applies: MIN = -500 and
MAX = -100.

Should you mix up MIN and MAX (i.e. MIN = -100; MAX = -500), the error bit is always set.

3.5.4 Releasing the SLS value displayed at the HMI

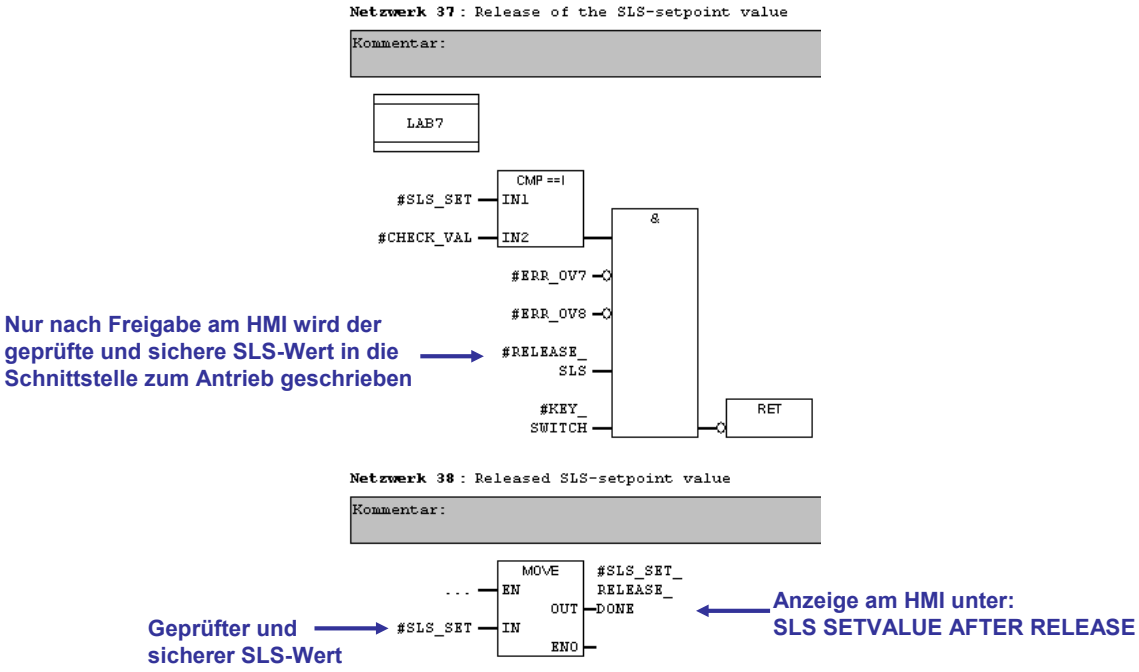
If no error has occurred (ERR_PLAUSIBILITY=0) the FB is further processed. The SLS value successfully checked for plausibility, hence considered safe, is offered to the user on the HMI for release.



The release (via the RELEASE 1 and RELEASE 2 button) is based on using FB F_ACK_OP. This is a certified block from the block library of Distributed Safety. This makes the release signal safe.

After pressing the RELEASE 1 button, you need to wait at least 1 second, but no longer than 1 minute before pressing RELEASE 2. Otherwise the release is not accepted.

In FB PLAUSIBILTY_AND_ERR (FB 2, DB2) of the F program, the release signal ensures a verified SLS value evaluated as safe, which is displayed at the HMI (SLS SETVALUE AFTER RELEASE).



This SLS value considered safe is transferred to the drive (see also chapter 3.5.5).

This storage location can, only be accessed in the F program (for one program cycle) if

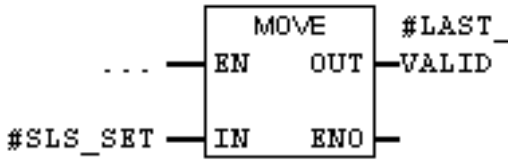
- the plausibility check has not detected any errors
- the user has released the desired SLS value

Note

Safety-related minimum and maximum values prepared here are determined by the user during the startup process. Possible critical values are intercepted by the SINAMICS drive through a setpoint value limitation.

3.5.5 SLS value in cyclical operation

As soon as the user sets the release at the HMI, a sequence is run through for one program cycle in the fail-safe FB PLAUSIBILITY_AND_ERR (FB2, DB2), in which the desired SLS value is stored safely as a static tag (LAST_VALID) (following the networks described in chapter 3.5.4).



In the F program (for one program cycle), this storage location can (as already described in chapter 3.5.4) only be accessed, if

- the plausibility check has not detected any errors and
- the user has released the desired SLS value.

If one of these conditions is not fulfilled, the last released SLS value LAST_VALID remains unchanged.

The value of SLS_SET_RELEASE_DONE from chapter 3.5.4, and the value of LAST_VALID are identical:

- The static LAST_VALID tag was introduced in order to display the last valid SLS value at the HMI.
- SLS_SET_RELEASE_DONE represents the prepared interface for the drive.

3.6 Measures against possible errors

What will you learn here?

This chapter describes an FMEA (failure mode and effect analysis) on the application example described here. The following errors are discussed here:

Communication error	Description	Chapter
Corruption	**Corrupted SLS value** The SLS value set at the HMI reaches the F-CPU corrupted.	3.6.1
	Corruption of SLS and modified values Apart from the SLS value, both modified values also reach the F-CPU corrupted.	3.6.2
	Freezing the individual bits A request at the HMI for the new SLS value is not accepted correctly.	3.6.3
Loss	**Voltage loss at the HMI**	3.6.4
	Voltage loss at the F-CPU	3.6.5

Basis

The FMEA described here follows DIN EN 61784-3, and there in particular Table 1 (“Overview of the effectiveness of measures against possible errors”).

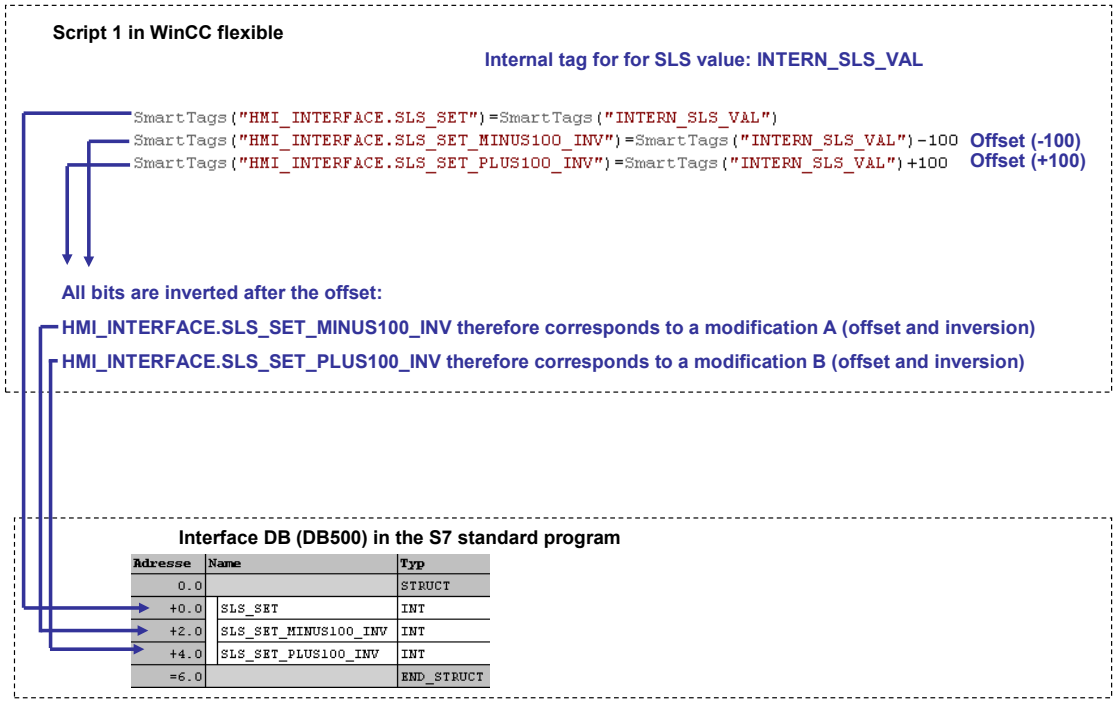
3.6.1 Corrupted SLS value

Error description

After entering the SLS value at the HMI, it is initially stored as internal tag in WinCC flexible and modified twice. Three values are given here:

- SLS value as internal tag
- Modification A of the SLS value
- Modification B of the SLS value

These three values are transferred at the interface DB (DB500) of the S7 standard program (see following picture).

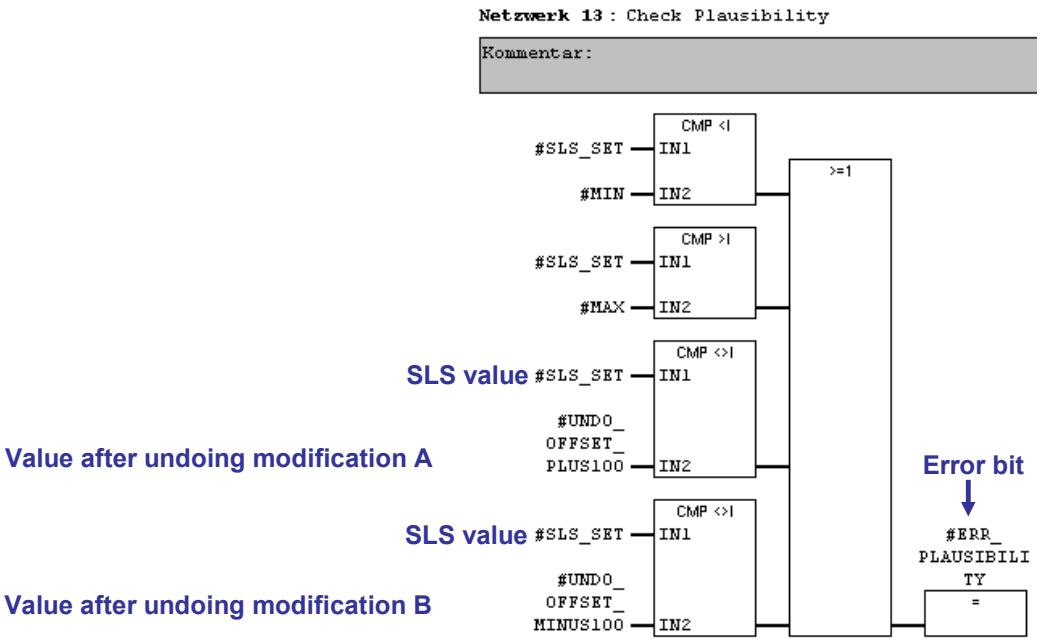


Due to a transmission error, the SLS value can be transferred faulty to the interface DB (DB500). The SLS value SLS_SET in DB500 (address 0) would hence (in OB35) be transferred faulty into the F program.

Error detection

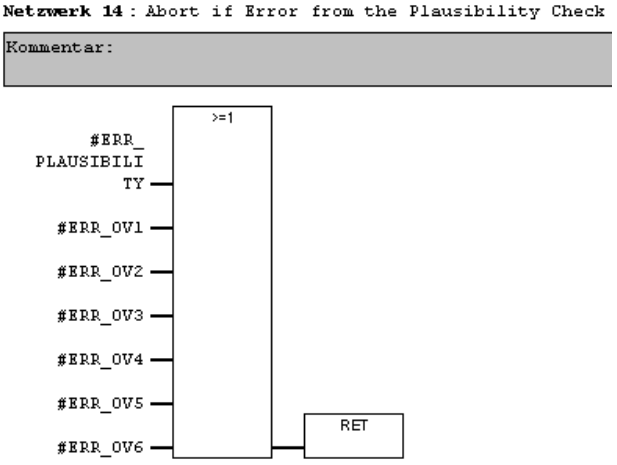
With modification A and modification B, two values are generated in script 1 of WinCC flexible, which are also transferred to the interface DB (DB500) (see address 2 and 4 in the picture above). Both of these values also reach the F program via OB35, where the modification is reversed. Then, both values must be equal to the SLS value.

The check for equality is performed in the F program. A corrupted SLS value would surely be uncovered in this way.



Error response

Error bit ERR_PLAUSIBILITY is set. This bit ensures, that the F program is safely aborted which prevents forwarding the SLS value to the interface of the drive.



Cancelling the F program if error bit ERR_PLAUSIBILITY or the overflow check for the operation (ADD_I, SUB_I,…) is set.

Release at the HMI takes no effect in the case of an error, since this program part is not reached and hence…

…no SLS value is written to the interface for the drive.

3.6.2 Corruption of SLS and modified values

Error description

Apart from the SLS value, both of the modified values are now also corrupted. Via the interface DB HMI_INTERFACE (DB500), three corrupted values now reach the F program of the F-CPU from the standard program (via flag in OB35).

In this error scenario, two variants are possible:

- **Variant 1**
  All three values (SLS value and both modified values) are **not** falsified by the same number, but a respective other number.

- **Variant 2**
  All three values (SLS value and both modified values) are falsified by the same number.

Error detection for variant 1

This case is surely uncovered in the diagnosis, irrespective of whether two or all three values were corrupted.

After undoing the modification and performing a comparison with the SLS value, the inequality between the values is recognized.

Example:

Assumption: SLS value 2000 is entered at the HMI.

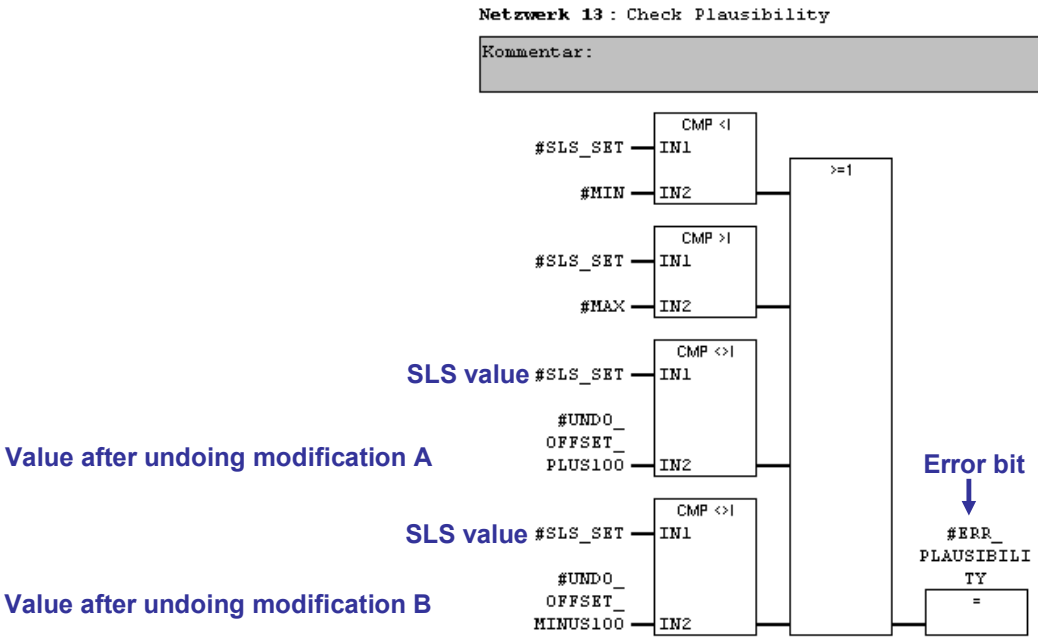
In the case of no errors, the following values would result:

SLS value	2000
Modification A	**HMI:** Subtraction of 100: 1900 (76C hex) Inverting all 16 Bits of 1900: F893 hex **F program:** Two's compliment of F893 hex: 076D hex Minus 1: 76C hex Addition with 100 (undoing the offset): **2000** **Modification A = SLS value**
Modification B	**HMI:** Addition of 100: 2100 (834 hex) Inverting all 16 Bits of 2100: F7CB hex **F program:** Two's compliment of F7CB hex: 835 hex Minus 1: 834 hex Subtraction with 100 (undoing the offset): **2000** **Modification B = SLS value**

In the case of no errors, for example, the following values would result:

SLS value	Falsified to 2001 (instead of 2000)
Modification A	**HMI:** Subtraction of 100: 1901 (76D hex) Inverting all 16 Bits except of Bit 1, for example: F890 hex **F program:** Two's compliment of F890 hex: F770 hex Minus 1: F76F hex Addition with 100 (undoing the offset): **63443** **Modification A unequal SLS value**
Modification B	**HMI:** Addition of 100: 2101 (835 hex) Inverting all 16 Bits except of Bit 2, for example: F7BF hex **F program:** Two's compliment of F7BF hex: 831 hex Minus 1: 830 hex Subtraction with 100 (undoing the offset): **1996** **Modification B unequal SLS value**

The comparison in the F program uncovers the error and the error bit is set (as in chapter 3.6.1):



Error response

The same statements as described in "Error response" in chapter 3.6.1 apply.

Error detection for variant 2:

Theoretically, the following case would be possible: The SLS value and both of the modified values are falsified, so that

- a faulty SLS value results **and**
- **both** differently modified values are falsified in **exactly** the way that after undoing the modification they are equal to the faulty SLS value.

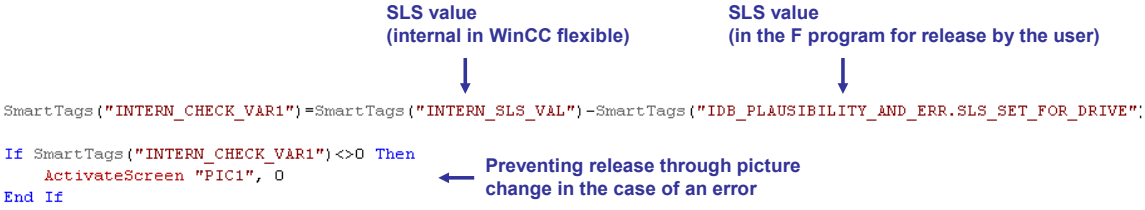
Additionally, this error can also occur in two variants:

Variant 2a: error occurs before REALESE 1 has been pressed

This case is nearly impossible. Nevertheless, this case can also be detected:

- The SLS value in the HMI is compared with the SLS value in the F program which the user must release (in Skript_2 of WinCC flexible).
- Furthermore, the SLS value is displayed to the user at the HMI for release.

Skript_2 of WinCC flexible is shown below. The SLS value in the HMI and the SLS value in the F program are subtracted. When the SLS value is corrupted, a value unequal zero results.



Error response

In this case, a jump into the other picture (PIC1) automatically occurs after pressing the RELEASE 1 button at the HMI. It is now no longer possible to release the SLS value by pressing the RELEASE 2 at the HMI.

In picture PIC1, the user is informed of a communication error:

Variant 2b: error occurs after REALESE 1 has been pressed

Here, the realization that this error is almost impossible also applies. However, FB PLAUSIBILTY_AND_ERR (FB2, DB2) also detects this scenario.

Error response

FB PLAUSIBILTY_AND_ERR (FB2, DB2) detects and reacts to this almost impossible error case in networks 28-36.

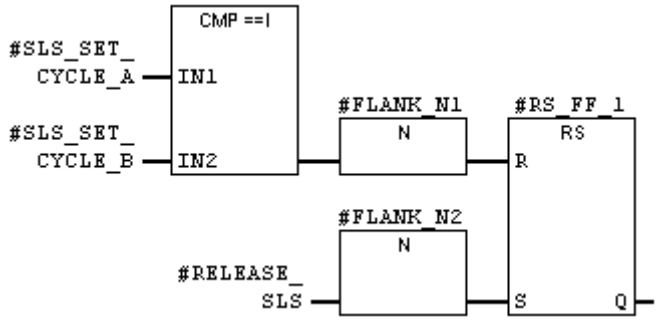
There, a bit changes in each program cycle. By means of this toggle bit, the currently requested SLS value in the current program cycle is compared with the value of the preceding program cycle. In this way, a value change can be detected and stored.

This stored value is now compared with the release at the RELEASE 2 button on the HMI with the currently valid SLS_SET value. If both values are unequal, no release is given for SLS_SET.

The figure below shows an extract from this F program.

The SLS value of the current program cycle is compared with the value of the preceding one (SLS_SET_CYCLE_A und SLS_SET_CYCLE_B). If the requested SLS value changes, the flip flop is reset and the new SLS value is saved (this occurs in the subsequent networks).

After a requested RELEASE_SLS, the flip flop is reset again.



3.6.3 Freezing individual bits

Error description

A new SLS value is entered at the HMI. However, the new SLS value is not correctly accepted due to an error.

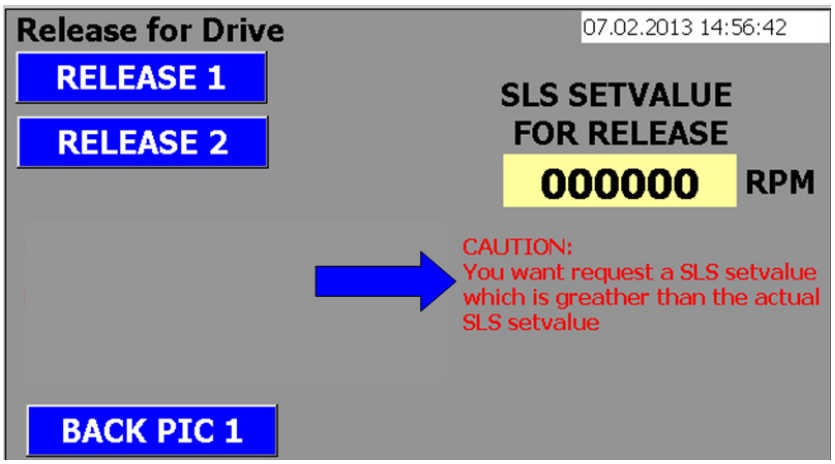
Example:

Display at the HMI	Note
Input: 245 Display: 245 (1 **1** 1 1 0 1 0 1)	**Old** SLS value
Input: 50 Input is not accepted correctly, since Bit 6 is not reset. Therefore, instead of 50 a higher value is now pending. Display: 114 (0 **1** 1 1 0 0 1 0)	**New** SLS value is entered, however, it is not accepted correctly since bit 6 in not inverted.

Error detection

This error is extremely unlikely. The user is also responsible for the release of the SLS value in this case. However, in order to provide additional help, the following has been realized:

In the F program, the requested SLS value is compared with the previously valid SLS value. If the user has requested am SLS value larger than the currently valid SLS value, he is informed of this state by the blinking text at the HMI:



Error response

The respective note does not only appear for the scenario described above in “Error description”, but generally when the requested SLS value is larger than the currently valid SLS value.

In this way, the user is warned and can recognize the error case. Subsequently, he may not release the SLS value or, in the case of no error, release the larger SLS value on purpose.

3.6.4 Voltage loss at the HMI

Error description

The HMI fails due to a missing voltage supply.

Error detection

No picture exists at the HMI.

Error response

The STEP 7 program of the F-CPU still keeps working. The SLS value in the F-CPU remains valid. A hazardous state cannot occur, since a new SLS value can only be requested and released via the (currently unavailable) HMI.

3.6.5 Voltage loss at the F-CPU

Error description

The F-CPU fails due to a missing voltage supply.

Error detection

No program processing.

Error response

A hazardous state cannot occur, since all values go to safe mode. After voltage recovery and operating state RUN of the F-CPU, the SLS values must be requested again.

3.6.6 Diversity of the software modules

Requirement

The software modules used for coding/decoding within the transmission/retransmission process, and software modules used for displaying the safety-related parameters for the user, must use diversity for the functions at least in order to prevent system failures.

Implementation

The diversity occurs in the bit pattern, firstly, with the modification in the script of WinCC flex, and secondly, by remodification in the F program.

Preventing system errors is supported by the display at the HMI in different windows:

- window 1 for entering the SLS value
- window 2 for acknowledging the SLS value reported back

4 Installation

4.1 Used IP addresses

IP addresses

Hardware	IP address	Subnet mask
HMI	192.168.0.**1**	255.255.255.0
F-CPU	192.168.0.**2**	255.255.255.0

4.2 Hardware Installation

Connect the HMI to the F-CPU via the Ethernet interface. To start up this application example it is necessary to connect a drive.

4.3 Software installation

No.	Action
1	Install STEP 7
2	Install Distributed Safety
3	Install WinCC flexible

4.4 Setting the PG/PC interface

No.	Action	Remarks
1	In SIMATIC Manager: “Options >Set PG/PC interface”	
2	Select the TCP IP interface	Setting according to the communication card used

4.5 Installation of the example project

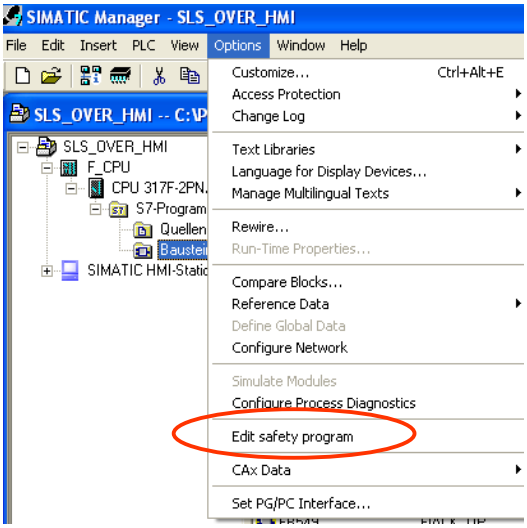
Retrieval

No.	Action
1	Load the zip-file provided on the HTML page to a local directory of the Window Explorer.
2	In the SIMATIC Manager go to “File -> Retrieve” and select the zip file. Follow the instructions.

Password

In all cases, the required safety password is: **siemens**

Download

No.	Action	Remarks
1	Set the mode switch of the S7-CPU to **STOP**.	
2	Load the HW Config to the F-CPU.	
3	Load the STEP 7 program from SIMATIC Manager to the F-CPU: • Select the block container • Menu: “Options > Edit safety program”	
4	Set the mode switch of the S7-CPU to **RUN**.	
5	Open WinCC flexible and load the project to the corresponding panel.	At the HMI you receive a note regarding a “PLAUSIBILITY CHECK ERROR”. This is correct at the case where no SLS value (SLS SETVALUE) has been entered, yet the plausibility check starts immediately after switching on.

5 Operation of the Application

5.1 Commissioning the example project

WARNING	**Using the example prepared by us does not release the user from the obligations to be fulfilled during startup or for fail-safe applications, such as ensuring that trained and authorized personnel only must perform the respective actions.**

WARNING	**You need to ensure that no unexpected actions, such as undesired starting of actuators, can occur due to the actions instructed below.**

Requirements:

- The hardware components are interconnected
- The actions from chapter 4.5 were performed.
- Key switch gives 1-signal from the F-DI

No.	Action	Remarks
1	Set the mode switch of the S7-CPU to **STOP**.	
2	In the SIMATIC Manager you open the FB F_MAIN (FB1, DB1) and specify the values for the parameters MIN and MAX at FB PLAUSIBILITY_AND_ERR (FB2, DB2).	
3	Save and close FB F_MAIN (FB1, DB1).	
4	In SIMATIC Manager: “Options > Edit safety program > Generate”	Select the “Blocks” container beforehand.
5	In SIMATIC Manager: “Options > Edit safety program > Download”	
6	Set the mode switch of the S7-CPU to RUN.	
7	At the HMI you enter an integer value, located within the MIN/MAX limits from No. 2, into the SLS SETVALUE input field.	You are automatically taken to screen 2 of the HMI
8	In SLS SETVALUE FOR RELEASE you check the displayed value for equality with the value entered at No. 7. If the values are equal (ok): continue with No. 9 If the values are unequal (Error): click BACK PIC 1 to go back to screen 1 of the HMI. Continue as specified at No. 7.	

No.	Action	Remarks
9	Consecutively click RELEASE 1 and RELEASE 2. The entered value and the safe value in the F program are both displayed. In the case of no errors, you receive the information PLAUSIBILITY CHECK OK	You only can see RELEASE 2 after pressing RELEASE 1. After clicking RELEASE 1, you need to wait at least 1s, but less than 1min before pressing RELEASE 2. Otherwise the release is not accepted.

5.2 Messages at the HMI

In the example project, the following messages have been prepared:

No.	Message	Cause / Note
1	PLAUSIBILITY CHECK OK	Values from the HMI are evaluated as plausible in the F program.
2	PLAUSIBILITY CHECK ERROR	The F program has detected an implausible state in the values of the HMI. Chap. 3.5
3	COMMUNICATION ERROR SET SLS SETVALUE AGAIN	Difference between the SLS value in the HMI and the SLS value in the F program. Chap. 3.6.2 (variant 2a)
4	CAUTION! You have no authorization! Key switch is necessary!	Without the 1-signal of the F-DI (key switch), no SLS value can be released. Chap. 3.3
5	CAUTION! You want to request a SLS setvalue which is greather than the actual setvalue!	Points out the fact, that the newly requested SLS value is larger than the previous one. Chap. 3.6.3
6	No release for required SLS SETVALUE Last released SLS SETVALUE still active! Change SLS SETVALUE and try again	As the case maybe, you have to change SLS SETVALUE on HMI once (with RELEASE 1 and RELEASE 2). Subsequent you can go on in the usual way.

6 Links & Literature

6.1 Literature

The following list is by no means complete and only provides a selection of appropriate information.

	Topic	Title
/1/	STEP7 SIMATIC S7-300/400	Automatisieren mit STEP7 in AWL und SCL (Automating with STEP7 in STL and SCL) Author: Hans Berger Publicis MCD Verlag ISBN: 978-3-89578-397-5
/2/	STEP7 SIMATIC S7-300/400	Automating with STEP 7 in LAD and FBD Author: Hans Berger Publicis MCD Verlag ISBN: 978-3-89578-296-1
/3/	STEP7 SIMATIC S7-300	Automating with SIMATIC S7-300 inside TIA Portal Author: Hans Berger Publicis MCD Verlag ISBN: 978-3-89578-357-9
/4/	STEP7 SIMATIC S7-400	Automating with SIMATIC S7-400 inside TIA Portal Author: Hans Berger Publicis MCD Verlag ISBN: 978-3-89578-372-2
/5/	STEP7 SIMATIC S7-1200	Automating with SIMATIC S7-1200 Author: Hans Berger Publicis MCD Verlag ISBN: 978-3-89578-355-5

6.2 Internet Links

The following list is by no means complete and only provides a selection of appropriate information.

	Topic	Title
\1\	Link to this document	http://support.automation.siemens.com/WW/view/en/67634251
\2\	Siemens Industry Online Support	http://support.automation.siemens.com
\3\	Distributed Safety - configuring and programming	http://support.automation.siemens.com/WW/view/en/22099875
\4\	SINAMICS S120 Function Manual Safety Integrated	http://support.automation.siemens.com/WW/view/en/59734511
\5\	SINAMICS S120: Controll9ng the integrated safety functions via TM54F and F-CPU	http://support.automation.siemens.com/WW/view/en/28424136
\6\	SINAMICS G120: Control via PROFINET, safety functions via PROFIsafe in cat. 3 (EN 954-1), SIL 2 (IEC 61508) and PLd (ISO 13849-1)	http://support.automation.siemens.com/WW/view/en/29585944
\7\	SINAMICS G120: Control via PROFIBUS with PROFIsafe, safety functions via PROFIsafe in cat. 3 (EN 954-1), SIL 2 (IEC 61508) and PLd (ISO 13849-1)	http://support.automation.siemens.com/WW/view/en/24093625

7 History

Version	Date	Revisions
V1.0	03/2013	First issue