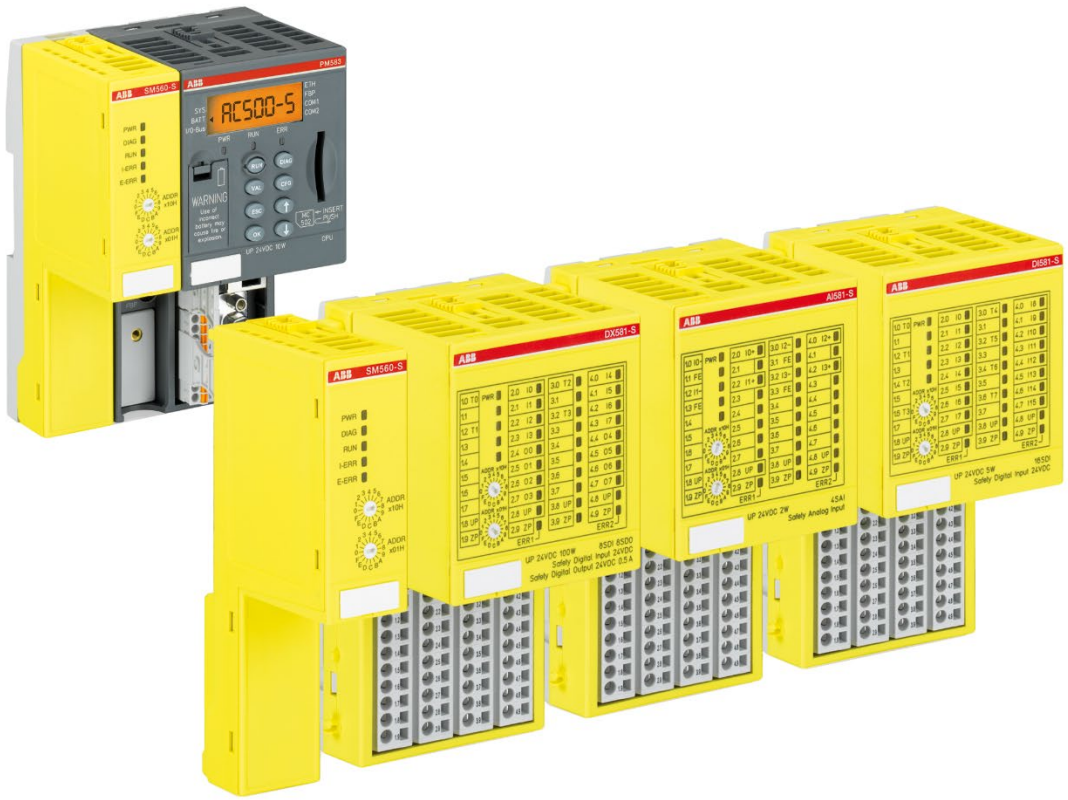
APPLICATION NOTE

AC500-S safety PLC

Cyclic non-safe data exchange between AC500-S safety CPU and PM5xx non-safety CPU



Contents

1. Introduction

1.1. Purpose

This application note describes an optional usage of cyclic non-safe data exchange via DPRAM between AC500-S safety CPU and PM5xx non-safety CPU to exchange process data between CPUs.

Motivation:
As requested by customers, a fast communication and/or big data amount transfer via DPRAM between AC500-S safety CPU and PM5xx non-safety CPUs is needed in some customer applications to synchronise process data on different CPUs. The DPRAM communication using existing FBs (SF_DPRAM_PM5XX_S_REC and SF_DPRAM_PM5XX_S_SEND, see [1.]) may be not fast enough and data amount (maximum 84 bytes) may be not big enough for some applications.

1.2. Document history

Rev.	Description of version / changes	Who	Date
D	Programming environment for safety devices was restyled and renamed to "AC500-S Programming Tool".	ABB	26.04.2023
C	Company name was changed. Various typos were corrected and various improvements in the texts and illustrations were made.	ABB	15.09.2021
B (V1.1.1)	Minor typos were corrected. Various improvements in the document: - Minor improvements in the text, e.g., "Danger" and "Warning" notices in the document. - chapter 5 includes a detailed overview of verification and validation actions. - Test example in chapter 7 was simplified for ease of use.	ABB	13.07.2017
A (V1.0.0)	First release	ABB	22.11.2013

1.3. Validity

The data and illustrations found in this documentation are not binding. ABB reserves the right to modify its products in line with its policy of continuous product development.

ABB assumes no liability or responsibility for any consequences arising from the im-proper use of this document information. ABB is in particular in no way liable for missed profits, loss of income, loss of use, loss of production, capital costs or costs associated with an interruption to operation, the loss of expected savings or for indirect or follow up damages or losses no matter of what kind.

1.4. Important user information

This documentation is intended for qualified personnel familiar with functional safety. You must read and understand the safety concepts and requirements presented in AC500-S safety User Manual [1.] as well as further referenced documents prior to operating AC500-S safety PLC system.

The following special notices may appear throughout this documentation to warn of potential hazards or to call attention to specific information.

⚠ DANGER

The notices referring to your personal safety are highlighted in the manual by this safety alert symbol, which indicates that death or severe personal injury may result if proper precautions are not taken.

NOTICE

This symbol of importance identifies information that is critical for successful application and understanding of the product. It indicates that an unintended result can occur if the corresponding information is not taken into account.

1.5. Definitions, expressions, abbreviations

AC500-S	ABB safety PLC for applications up to SIL3 (IEC 61508 ed. 2 and IEC 62061) and ISO 13849
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check. A number derived from and stored or transmitted with a block of data in order to detect data corruption.
DevDesc	Device Description in PS501 Control Builder Plus or Automation Builder
DPRAM	Dual-ported Random Access Memory
IEC	International Electro-technical Commission Standard
EN	European Norm (European Standard)
FB	Function Block
PL	Performance Level according to ISO 13849
PLC	Programmable Logic Controller
POU	Program Organization Unit
RAM	Random Access Memory
SIL	Safety Integrity Level (IEC 61508)
TÜV	Technischer Überwachungs-Verein (Technical Inspection Association)

1.6. References / related documents

[1.] AC500-S safety User Manual, 3ADR025091M0205 (or newer)

[2.] AC500 User Documentation, Automation Builder 1.0.1 (or newer)

[3.] Creation of safety-oriented applications with CoDeSys V2.3 – Document version 1.8

2. Installation

In this chapter, a step-by-step explanation is provided on how to install additional function blocks for cyclic non-safe data exchange between AC500-S safety CPU and PM5xx non-safety CPU.

NOTICE

Non-safe cyclic data transfer configuration is a part of the standard installation of Automation Builder V1.0.1 or higher.

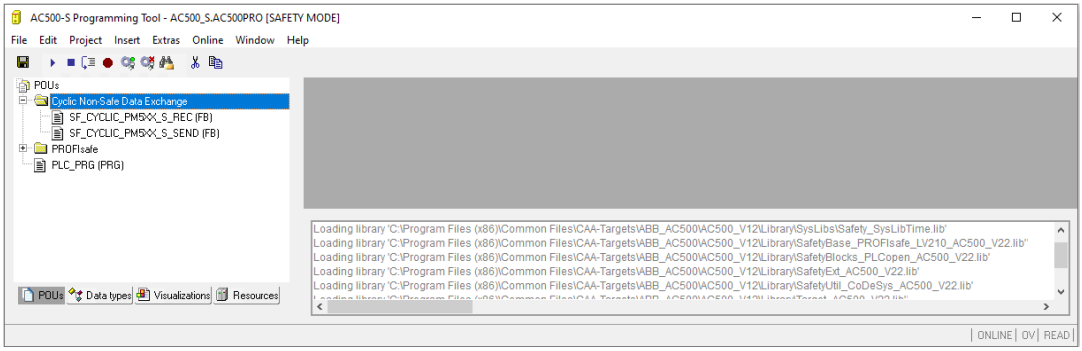
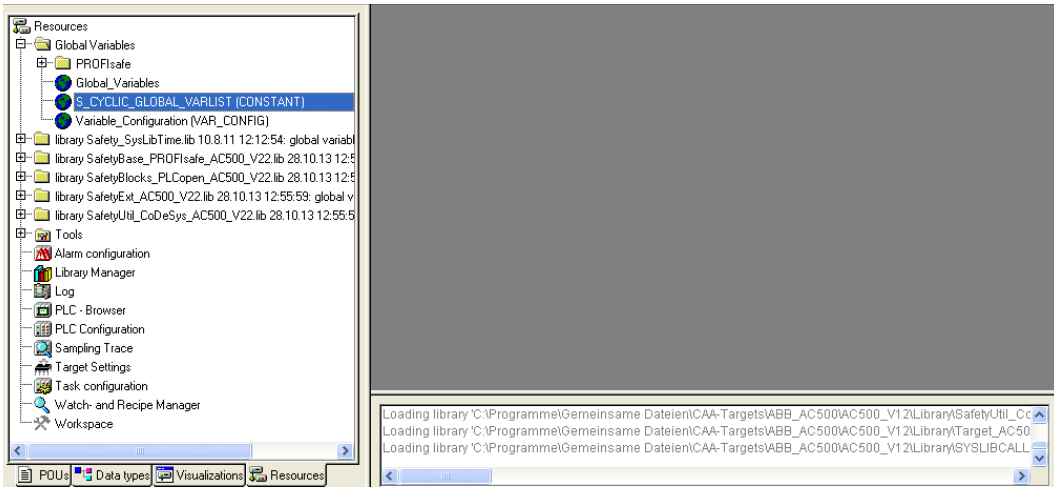
The export file "SAFETYCYCLICDATA_AC500_V23.EXP", which includes two function blocks SF_CYCLIC_PM5XX_S_SEND and SF_CYCLIC_PM5XX_S_REC, is not a part of the standard installation environment of Automation Builder.

Contact ABB technical support to obtain this export file.

Since function blocks SF_CYCLIC_PM5XX_S_SEND and SF_CYCLIC_PM5XX_S_REC are NOT a part of the standard installation of Automation Builder, these steps for export file SAFETYCYCLICDATA_AC500_V23.EXP have to be done once before the first usage.

NOTICE

Although AC500-S Programming Tool provides mechanisms to detect and overwrite already existing POUs or variable lists, it shall be ensured that no elements of previous imports of "Cyclic Non-Safe Data Exchange" are located in the target project to ensure a clean and consistent setup.

Step	Action
1	Check if the export file SAFETYCYCLICDATA_AC500_V23.EXP has already been imported: • Open the safety project of AC500-S safety CPU by double-clicking the node “AC500_S” in the configuration tree (please refer to AC500-S safety User Manual [1.] for details). • Check if the folder “Cyclic Non-Safe Data Exchange” exists. • Check if the global variable list “S_CYCLIC_GLOBAL_VARLIST (CON-STANT)” exists. If not present, proceed with step 2. Otherwise, make sure that original FBs are present in the project, e.g., compare the source code of existing FBs with original ones from SAFETYCYCLICDATA_AC500_V23.EXP.  
2	Import SAFETYCYCLICDATA_AC500_V23.EXP file using standard import functionality (see AC500 User Documentation [2.] for details) in safety project. Re-check if the import was successful (see step 1).

3. Configuration

This chapter describes how to configure the AC500-S safety CPU within Automation Builder to use cyclic non-safe data exchange between AC500-S safety CPU and PM5xx non-safety CPU.

3.1. General

For the configuration of cyclic non-safe data exchange, tab „Data exchange configuration“ of the AC500-S safety CPU is available in Automation Builder, as illustrated in the next figure.

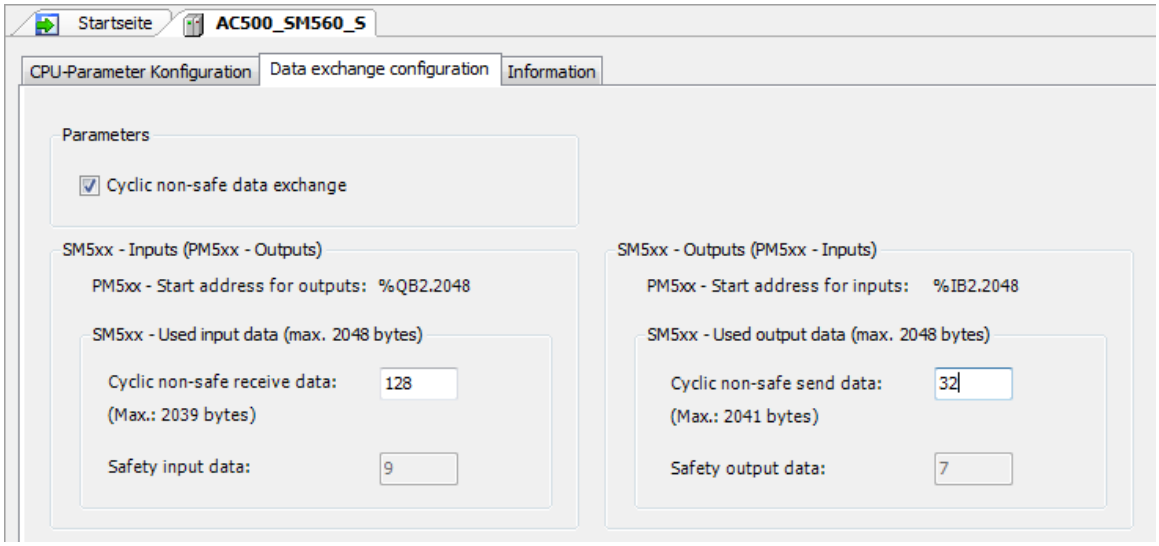


Figure 1: Editor „Data exchange configuration“

Steps to activate “Cyclic non-safe data exchange”:

1. Enable cyclic non-safe data exchange (activate check box “Cyclic non-safe data exchange”).
2. Define the length of cyclic non-safe receive data and, respectively, cyclic non-safe send data by editing the corresponding boxes.
3. The data size of the configured PROFIsafe modules is displayed as “Safety input data” respectively “Safety output data”.

NOTICE

Cycle time of the safety CPU may increase by up to 10 ms (maximum payload of 2 kByte for both directions) if cyclic non-safe data exchange is used.

Table 1 provides an overview of parameter configurations of used PM5xx and safety CPU for performance tests with cyclic non-safe data exchange input and output size = 512 Bytes.

Table 1: Overview of parameter configurations of used PM5xx and safety CPU for performance tests

	Con-fig 1	Con-fig 2	Con-fig 3	Con-fig 4	Con-fig 5	Con-fig 6	Con-fig 7	Con-fig 8	Con-fig 9
PM5XX cycle time	1ms	1m	1ms	1ms	1ms	1ms	1ms	1ms	1ms
Safety CPU cycle time	3ms	3ms	3ms	5ms	5ms	5ms	10ms	10ms	10ms
Safety CPU mini-mal up-date time *	0ms	2ms	5ms	0ms	3ms	7ms	0ms	5ms	15ms

Note:

* Safety CPU minimal update time is defined as 0%, 50% and 150% of safety CPU cycle time

Table 2 provides an overview of performance results for PM5xx and safety CPU configurations listed in Table 1.

1000 measurements were done for each configuration. The measured reaction time is defined as a data transfer from PM5xx → safety CPU and then back from safety CPU → PM5xx.

Table 2: Overview of performance results for PM5xx and safety CPU configurations listed in Table 1

	Measured reaction time		
	Minimum	**Maximum**	**Average**
Config 1	4 ms	8 ms	6 ms
Config 2	5 ms	10 ms	7 ms
Config 3	6 ms	14 ms	9 ms
Config 4	8 ms	13 ms	12 ms
Config 5	8 ms	13 ms	12 ms
Config 6	8 ms	17 ms	12 ms
Config 7	18 ms	28 ms	26 ms
Config 8	18 ms	28 ms	27 ms
Config 9	18 ms	38 ms	27 ms

3.2. Input data (AC500-S safety CPU)

The area for the input data of the AC500-S safety CPU is maximum 2048 bytes. The used input data is displayed at „Safety input data“. The remaining data bytes up to the maximum value (2048 - „Safety input data“) can be used as „Cyclic non-safe receive data“.

⚠ DANGER

The user has to take care about the total used size, which must not exceed 2048 bytes, as the sum of both safety input data and cyclic non-safe receive data.

A possible data size overflow will be indicated from Automation Builder (error message box arises).

It is, however, still mandatory that the final check (verification and validation) is performed by users as described in chapter 5 to ensure the correct configuration of the data exchange and avoid unexpected system behaviour.

NOTICE

For performance reasons, it is useful to limit „Cyclic non-safe receive data“ size to the needed length.

⚠ DANGER

The length of the „Cyclic non-safe receive data“ must be equal to the input value passed to input DATA_LEN of FB SF_CYCLIC_PM5XX_S_REC in the safety program.

For this reason, it is highly recommended to use SIZEOF() with the used variable to provide the data length programmatically (see code example in chapter 7.3 for further details).

It is, however, still mandatory that the final check (verification and validation) is performed by users as described in chapter 5 to ensure the correct configuration of the data exchange and avoid unexpected system behaviour.

NOTICE

If the data length of „Cyclic non-safe receive data“ is a multiple of 4, the FB is working with DWORD access, otherwise with BYTE access is used.

The DWORD access is executed faster.

For the PM5xx project, no FBs are needed. The PM5xx application uses the area with the appropriate start address %QBx.2048 for accessing the data (x = slot number of AC500-S safety CPU):

- slot 1 = %QB1.2048
- slot 2 = %QB2.2048
- slot 3 = %QB3.2048
- slot 4 = %QB4.2048.

⚠ DANGER

Make sure that the PM5xx task, which writes to data area starting from %QBx.2048, has the highest priority out of all PM5xx tasks.

Otherwise, inconsistent data may be written to AC500-S safety CPU, which may lead to the unexpected system behaviour.

NOTICE

It is not forbidden to use more than one instance of the function block SF_CYCLIC_PM5XX_S_REC in the safety program, but this is not useful because the received non-safe data are consistent within one application cycle.

3.3. Output data (AC500-S safety CPU)

The area for the output data of the AC500-S safety CPU is limited to 2048 bytes. The used output data are displayed at „Safety output data“, which are allocated in this area as well.

The remaining data bytes up to the maximum value (2048 - „Safety output data“) can be used as „Cyclic non-safe send data“.

⚠ DANGER

The user has to take care about the total used size which must not exceed 2048 bytes, as the sum of both safety output data and cyclic non-safe send data.

A data size overflow will be indicated from Automation Builder (error message box arises).

It is, however, still mandatory that the final check (verification and validation) is performed by users as described in chapter 5 to ensure the correct configuration of the data exchange and avoid unexpected system behaviour.

NOTICE

For performance reasons, it is useful to limit „Cyclic non-safe send data“ size to the needed length.

⚠ DANGER

The length of the „Cyclic non-safe send data“ must be equal to the input value passed to input DATA_LEN of FB SF_CYCLIC_PM5XX_S_SEND in the safety program.

For this reason it is highly recommended to use SIZEOF() with the used variable to provide the data length programmatically (see code example in chapter 7.3 for further details).

It is, however, still mandatory that the final check (verification and validation) is performed by users as described in chapter 5 to ensure the correct configuration of the data exchange and avoid unexpected system behaviour.

NOTICE

If the data length of „Cyclic non-safe send data" is a multiple of 4, the FB is working with DWORD access, otherwise with BYTE access. The DWORD access is executed faster.

For the PM5xx project, no FBs are needed. The data of the AC500-S safety CPU are available at the PM5xx application in the area with the start address %IBx.2048 (x = slot number of AC500-S safety CPU):

- slot 1 = %IB1.2048
- slot 2 = %IB2.2048
- slot 3 = %IB3.2048
- slot 4 = %IB4.2048.

⚠ DANGER

Make sure that the PM5xx task, which reads from data area starting at %IBx.2048, has the highest priority out of all PM5xx tasks.

Otherwise, inconsistent data may be read from AC500-S safety CPU, which may lead to the unexpected system behaviour.

NOTICE

It is not forbidden to use more than one instance of the function block SF_CYCLIC_PM5XX_S_SEND in the safety program, but this is not useful because the non-safe data of the last executed instance will be transferred and the data of the previous instances will be lost.

4. Function block description

File SAFETYCYCLICDATA_AC500_V23.EXP provides the following POUs:

- SF_CYCLIC_PM5XX_S_SEND (Sending cyclic non-safe data to PM5XX via DPRAM)
- SF_CYCLIC_PM5XX_S_REC (Receiving cyclic non-safe data from PM5XX via DPRAM).

4.1. SF_CYCLIC_PM5XX_S_SEND

4.1.1. Functionality

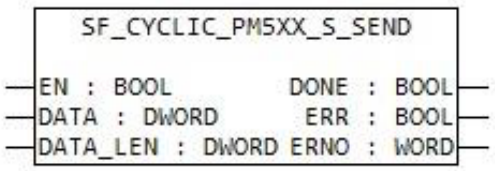


Figure 2: SF_CYCLIC_PM5XX_S_SEND

FB name	SF_CYCLIC_PM5XX_S_SEND

The FB SF_CYCLIC_PM5XX_S_SEND is used to send non-safety data cyclically to the PM5xx non-safety CPU. The data to be sent are available in the memory area, provided via ADR operator at the input DATA. The FB is enabled by a TRUE signal at the input EN. The FB remains active until input EN is set to FALSE. The length of the data is specified in bytes at the input DATA_LEN.

DONE = TRUE and ERR = FALSE indicate that the sending was successful. If an error was detected during FB processing, the error is indicated at the outputs ERR and ERNO.

NOTICE

Function block SF_CYCLIC_PM5XX_S_SEND is needed only if

“Cyclic non-safe send data” > 0.

⚠ DANGER

If FB SF_CYCLIC_PM5XX_S_SEND is used to send safety-critical data, then SIL3 (IEC 61508 and IEC 62061) and PL e (ISO 13849-1) safety requirements will not be fulfilled for sent data (in-dependently on application safety communication profile used), because only one microprocessor (no 1oo2 safety architecture in the background) on AC500-S safety CPU handles FB SF_CY-CLIC_PM5XX_S_SEND.

Contact ABB technical support on how to reach SIL 3 and PL e with FB SF_CYCLIC_PM5XX_S_SEND or use PROFIsafe safety Output, e.g., from DX581-S to trigger safety functions (see [1.] for more details).

VAR_INPUT			
Name	*Data type*	*Initial value*	*Description, parameter values*
EN	BOOL	FALSE	Enabling of function block processing
DATA	DWORD	0	Memory address for data to be transmitted, provided via ADR operator
DATA_LEN	DWORD	0	Length of data to be transmitted (in bytes) starting at address DATA
VAR_OUTPUT			
DONE	BOOL	FALSE	Processing finished
ERR	BOOL	FALSE	Error message
ERNO	WORD	0	Error number

4.1.2. Detailed input description

EN BOOL (enable)

The input EN enables the FB processing.

EN = TRUE:

The data are sent to the PM5xx non-safety CPU (the non-safety CPU can read the data at the area with the start address %IBx.2048, see also chapter 3 "Configuration"). This is indicated at the outputs DONE = TRUE and ERR = FALSE. An error is indicated with DONE = TRUE and ERR = TRUE (see output ERNO).

EN = FALSE:

The data are not sent to the non-safety CPU. The outputs are set to DONE = FALSE, ERR = FALSE and ERNO = 0. The area of the non-safety CPU with start address %IBx.2048 (see also chapter 3 "Configuration") contains old values (no actual values from AC500-S safety CPU).

DATA DWORD (data)

Input DATA is used to specify the address of the variable the user data are to be copied from (source buffer). The address specified at DATA has to point to a variable of the type ARRAY or STRUCT.

NOTICE

Set the variable size of the source buffer to the maximum amount of data in order to avoid overlapping of memory areas (buffer size shall be equivalent to input value DATA_LEN).

NOTICE

It is highly recommended not to change the address of DATA at runtime (otherwise, the intended data may not be transmitted).

DATA_LEN DWORD (data length)

The length of data to be transmitted is specified in bytes at input DATA_LEN.

⚠ DANGER

The length value passed to input DATA_LEN of FB SF_CYCLIC_PM5XX_S_SEND must be equal to the configured value „Cyclic non-safe send data“.

For this reason it is highly recommended to use SIZEOF() with the used variable to provide the data length programmatically (see code example in chapter 7.3 for further details).

It is, however, still mandatory that the final check (verification and validation) is performed by users as described in chapter 5 to ensure the correct configuration of the data exchange and avoid unexpected system behaviour.

NOTICE

The data length set at DATA_LEN shall be constant for the active instance of the function block. Otherwise, the FB detects a difference between configured data length (“Cyclic non-safe send data”) and input DATA_LEN and generates an error with error number 20.

NOTICE

If the data length is a multiple of 4, the FB is working with DWORD access, otherwise with BYTE access.

The DWORD access is executed faster.

4.1.3. Detailed output description

DONE BOOL (done)

Output DONE indicates that the processing of the FB was finished. The output has to be always considered together with output ERR.

The following applies:

- DONE = TRUE and ERR = FALSE:

  The data was sent to the non-safety CPU.

- DONE = TRUE and ERR = TRUE: An error occurred.

  The error number is indicated at the output ERNO.

ERR BOOL (error)

Output ERR indicates whether an error occurred during FB processing. This output always has to be considered together with output DONE. The following applies if an error occurred during FB processing: DONE = TRUE and ERR = TRUE. Output ERNO indicates the error number.

ERNO WORD (error number)

Output ERNO provides the error number:

- 0: no error

- 10: DATA_LEN = 0 or DATA_LEN greater than 2048
- 20: configuration error: DATA_LEN (AC500-S safety CPU) and „Cyclic non-safe send data“ (editor “Data exchange configuration”) are different
- 30: address at input DATA not valid (DATA = 0)

4.2. SF_CYCLIC_PM5XX_S_REC

4.2.1. Functionality

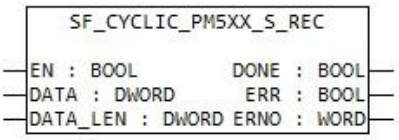


Figure 3: SF_CYCLIC_PM5XX_S_REC

FB Name	**SF_CYCLIC_PM5XX_S_REC**

The FB SF_CYCLIC_PM5XX_S_REC is used to receive non-safety data cyclically from the PM5xx non-safety CPU. These data are stored in the reserved memory area (DATA: memory address for received data, provided via ADR operator). The FB is enabled by a TRUE signal at the input EN. The FB remains active until input EN is set to FALSE. The length of the data is specified in bytes at the input DATA_LEN. DONE = TRUE and ERR = FALSE indicate that receiving was successful. If an error was detected during FB processing, the error is indicated at the outputs ERR and ERNO.

NOTICE

Function block SF_CYCLIC_PM5XX_S_REC is need only if “Cyclic non-safe receive data” > 0.

VAR_INPUT

Name	*Data type*	*Initial value*	*Description, parameter values*
EN	BOOL	FALSE	Enabling of function block processing
DATA	DWORD	0	Memory address for data to be received, provided via ADR operator
DATA_LEN	DWORD	0	Length of data to be received (in bytes) starting at address DATA

VAR_OUTPUT

DONE	BOOL	FALSE	Processing finished
ERR	BOOL	FALSE	Error message
ERNO	WORD	0	Error number

4.2.2. Detailed input description

EN BOOL (enable)

The input EN enables the FB processing.

EN = TRUE:

The data are received from the PM5xx non-safety CPU. The non-safety CPU stores the data at the area with the start address %QBx.2048 (see also chapter 3 "Configuration"). The receiving is indicated with DONE = TRUE and ERR = FALSE. An error is indicated with DONE = TRUE and ERR = TRUE (see output ERNO).

EN = FALSE:

The data are not received from the non-safety CPU (the outputs are set to DONE = FALSE, ERR = FALSE und ERNO = 0). The area with the start address DATA contains old values (no actual data from the non-safety CPU).

DATA DWORD (data)

Input DATA (destination buffer) is used to specify the address of the variable the user data is to be copied to. The address specified at DATA has to point to a variable of the type ARRAY or STRUCT.

NOTICE

Set the variable size of the destination buffer to the maximum amount of data in order to avoid overlapping of memory areas (buffer size shall be equivalent to input value DATA_LEN).

NOTICE

It is highly recommended not to change the address of DATA on runtime (otherwise, the received data may not be copied into the intended destination buffer).

DATA_LEN DWORD (data length)

The length of data to be received is specified in bytes at input DATA_LEN.

⚠ DANGER

The length value passed to input DATA_LEN of FB SF_CYCLIC_PM5XX_S_REC must be equal to the configured value „Cyclic non-safe receive data“.

For this reason it is highly recommended to use SIZEOF() with the used variable to provide the data length programmatically (see code example in chapter 7.3 for further details).

It is, however, still mandatory that the final check (verification and validation) is performed by users as described in chapter 5 to ensure the correct configuration of the data exchange and avoid unexpected system behaviour.

NOTICE

The data length set at DATA_LEN shall be constant for the active instance of the function block.

Otherwise, the FB detects a difference between configured data length (“Cyclic non-safe receive data”) and input DATA_LEN and generates an error with error number 20.

NOTICE

If the data length is a multiple of 4, the FB is working with DWORD access, otherwise with BYTE access.

The DWORD access is executed faster.

4.2.3. Detailed output description

DONE BOOL (done)

Output DONE indicates that the processing of the FB was finished. The output has to be always considered together with the output ERR.

The following applies:

- DONE = TRUE and ERR = FALSE:

  The data were received from the non-safety CPU.

- DONE = TRUE and ERR = TRUE:

  An error occurred. The error number is indicated at output ERNO.

ERR BOOL (error)

Output ERR indicates whether an error occurred during FB processing. This output has to be always considered together with the output DONE. The following applies if an error occurred during FB processing:

- DONE = TRUE and ERR = TRUE; Output ERNO indicates the error number.

ERNO WORD (error number)

Output ERNO provides the error number:

- 0: no error
- 10: DATA_LEN = 0 or DATA_LEN greater than 2048
- 20: configuration error: DATA_LEN (AC500-S safety CPU) and „Cyclic non-safe receive data“ (editor “Data exchange configuration”) are different
- 30: address at input DATA not valid (DATA = 0)

5. Verification and validation of cyclic non-safe data exchange

The following scenarios must be considered for verification and validation purposes. Possible safety application impact has to be tracked with additional care.

Step no.	Configuration step	Potential configuration error	Impact on safety application	Highly recommended verification measure (see also Chapter 6.2)	Highly recommended validation measure
1	Definition of length of cyclic non-safe receive data in “Data exchange configuration”	Value is too big and more than 2048 bytes in total	No impact because this value is not accepted from Automation Builder configuration.	Checklist item 1.1	Not needed
		Value is bigger than used value at input DATA_LEN of FB SF_CYCLIC_PM5XX_S_REC	No impact, because the FB SF_CYCLIC_PM5XX_S_REC checks the configured size and the DATA_LEN value which must be equal. If not equal, nothing is copied in the receive array and the FB signals an error.	Checklist item 3.1	Validation test (chapter 7). The error will be detected on the safety CPU: • Test pattern not received. • FB SF_CYCLIC_PM5XX_S_REC signals error at output ERNO.

		Value is smaller than used value at input DATA_LEN of FB SF_CYCLIC_ PM5XX_S_REC	No impact, because the FB SF_CYCLIC_ PM5XX_S_REC checks the configured size and the DATA_LEN value which must be equal. If not equal, nothing is copied in the receive array and the FB signals an error.	Checklist item 3.1	Validation test (chapter 7). The error will be detected on the safety CPU: • Test pattern not received. • FB SF_CYCLIC_ PM5XX_S_ REC signals error at output ERNO.
2	Definition of length of cyclic non-safe send data in “Data exchange configuration”	Value is too big and more than 2048 bytes in total	No impact because this value is not accepted from Automation Builder configuration.	Checklist item 1.1	Not needed
		Value is bigger than used value at input DATA_LEN of FB SF_CYCLIC_ PM5XX_S_SEND	No impact, because the FB SF_CYCLIC_ PM5XX_S_SEND checks the configured size and the DATA_LEN value which must be equal. If not equal, nothing is copied to the send array and the FB signals an error.	Checklist item 2.1	Validation test (chapter 7). The error will be detected on PM5xx: • Test pattern not received. • FB SF_CYCLIC_ PM5XX_S_ SEND signals error at output ERNO.
		Value is smaller than used value at input DATA_LEN of FB SF_CYCLIC_ PM5XX_S_SEND	No impact, because the FB SF_CYCLIC_ PM5XX_S_SEND checks the configured size and the DATA_LEN value which must be equal. If not equal, nothing is copied to the transmit array and the FB signals an error.	Checklist item 2.1	Validation test (chapter 7). The error will be detected on PM5xx: • Test pattern not received. • FB SF_CYCLIC_ PM5XX_S_ SEND signals error at output ERNO

3	Size of array (or struct) for DATA input of SF_ CYCLIC_ PM5XX_S _REC FB	Size of array (struct) is too small for received data (FB input DATA_LEN > array size)	There is an impact which may result in overwritten safety data which are located behind the array.	Checklist items 3.3 and 3.7	Validation test (chapter 7). The error will be detected on the safety CPU: • Test pattern not at expected position (last byte in the receive array).
		Size of array (struct) is too big for received data (FB input DATA_LEN < array size)	No impact. The array will be only partially written.	Checklist items 3.3 and 3.7	Validation test (chapter 7). The error will be detected on the safety CPU: • Test pattern not at expected position (last byte in the receive array).
4	Size of array (or struct) for DATA input of SF_ CYCLIC_ PM5XX_S _SEND FB	Size of array (struct) is too small for sent data (FB input DATA_LEN > array size)	Potential impact (it depends on the application type), because the ap-plication data (not intended to be transferred) which are located be-hind the array is transferred on top of the planned send data.	Checklist items 2.3 and 2.8	Validation test (chapter 7). The error will be detected on PM5xx: • Test pattern not at expected position (last byte in the receive array).
		Size of array (struct) is too big for sent data (FB input DATA_LEN < array size)	Potential impact (it depends on the application type), because the array is not transferred completely.	Checklist items 2.3 and 2.8	Validation test (chapter 7). The error will be detected on PM5xx: • Test pattern not at expected position (last byte in the receive array).

5	Wrong array (or struct) for DATA input of SF_CYCLIC_PM5XX_S_REC FB	Size of wrong referenced array (struct) is too big respectively too small, or even is identical for received data	There is a safety impact which results in the over-written safety data inside the wrong referenced array respectively of safety data which are located behind the array.	Checklist item 3.7	Validation test (chapter 7). The error will be detected on the safety CPU: • Test pattern not at expected position (last byte in the receive array).
6	Wrong array (or struct) for DATA input of SF_CYCLIC_PM5XX_S_SEND FB	Size of wrong referenced array (struct) is too big respectively too small, or even is identical for sent data.	Potential safety impact, because completely wrong data is transferred.	Checklist item 2.8	Validation test (chapter 7). The error will be detected on PM5xx: • Test pattern not at expected position (last byte in the receive array).

NOTICE

Exceedance (> 2048 bytes in total) of configured values in “Data exchange configuration” tab is prevented by the Automation Builder.

Inconsistencies between configured data length values in the “Data exchange configuration” tab and the used values at input DATA_LEN of SF_CYCLIC_PM5XX_S_REC and SF_CYCLIC_ PM5XX_S_SEND FBs are detected by FBs, which leads to the result of no copying of data and stating an error at FB output ERNO.

⚠ DANGER

If the user does not fulfil the checklist items defined in chapter 6.2, the following specific situations are not guaranteed to be detected:

- **Setting the DATA_LEN of the FB SF_CYCLIC_PM5XX_S_REC bigger than the actual array referenced at input DATA;**
- **Wrong buffer address is given on FB input DATA.**

A potential impact in both scenarios is that safety data of the user safety program can be overwritten by the FB SF_CYCLIC_PM5XX_S_REC.

Usage of SIZEOF() function with the used variable to provide the data length programmatically will prevent any errors related to the wrong DATA_LEN usage. Thus, usage of SIZEOF() function for DATA_LEN inputs of SF_CYCLIC_PM5XX_S_REC and SF_CYCLIC_PM5XX_S_SEND FBs is highly recommended.

⚠ DANGER

One important fact is the testing of the proper function of the cyclic non-safe data exchange as part of the safety application validation.

It is, therefore, strongly recommended to the user to perform a kind of pattern test, as described in chapter 7, to ensure correct cyclic non-safe data exchange.

6. Checklist

6.1. Overview

All users of cyclic non-safe data exchange between AC500-S safety CPU and PM5xx non-safety CPU shall fill out the checklist presented below and document it in their final reports.

NOTICE

The items presented in the checklist include only those related to the cyclic non-safe data exchange. It means that further checklists according AC500-S safety User Manual [1.] will not become obsolete and still have to be fulfilled.

6.2. Checklist for cyclic non-safe data exchange

Nr.	Item to check	Fulfilled (yes / no)?	Comment
1. Configuration			
1.1	Check configuration tab settings: • Sum of cyclic non-safe receive data and safety input data ≤ 2048 bytes. • Sum of cyclic non-safe send data and safety output data ≤ 2048 bytes.		
1.2	Make sure that the PM5xx task, which writes to data area starting from %QBx.2048, has the highest priority out of all PM5xx tasks. Otherwise, inconsistent data may be written to AC500-S safety CPU.		
1.3	Make sure that the PM5xx task, which reads from data area starting at %IBx.2048, has the highest priority out of all PM5xx tasks. Otherwise, inconsistent data may be read from AC500-S safety CPU.		
2. Implementation of FB SF_CYCLIC_PM5XX_S_SEND			
2.1	The length of the „Cyclic non-safe send data“ is equal to the input value passed to input DATA_LEN of function block SF_CY-CLIC_PM5XX_S_SEND.		
2.2	Only one instance of the function block SF_CY-CLIC_PM5XX_S_SEND is used.		

Nr.	Item to check	Fulfilled (yes / no)?	Comment
2.3	FB SF_CYCLIC_PM5XX_S_SEND: • The variable size of the source buffer at ADR(DATA) is set to the maximum amount of data.		
2.4	FB SF_CYCLIC_PM5XX_S_SEND: • Input DATA is not changed at runtime.		
2.5	FB SF_CYCLIC_PM5XX_S_SEND: • The source memory area is fixed for the active instance of the function block.		
2.6	FB SF_CYCLIC_PM5XX_S_SEND: • The data length set at DATA_LEN is constant for the active instance of the function block.		
2.7	Make sure that only safety functions with up to SIL2 (IEC 61508 and IEC 62061) and PL d (ISO 13849-1) will be triggered using FB SF_CYCLIC_PM5XX_S_SEND. Note: • If FB SF_CYCLIC_PM5XX_S_SEND is used to send safety-critical data, then SIL3 (IEC 61508 and IEC 62061) and PL e (ISO 13849-1) safety requirements will not be fulfilled for sent data (independently on application safety communication profile used), because only one microprocessor (no 1oo2 safety architecture in the background) on AC500-S safety CPU handles FB SF_CY-CLIC_PM5XX_S_SEND. • Contact ABB technical support on how to reach SIL 3 and PL e with FB SF_CY-CLIC_PM5XX_S_SEND or use PROFIsafe safety Output to trigger safety functions (see [1.] for more details).		
2.8	FB SF_CYCLIC_PM5XX_S_SEND: • The data length at DATA_LEN is set with SIZEOF() of the used variable.		
3. Implementation of FB SF_CYCLIC_PM5XX_S_REC			
3.1	The length of the „Cyclic non-safe receive data“ is equal to the input value passed to in-put DATA_LEN of function block SF_CY-CLIC_PM5XX_S_REC.		

Nr.	Item to check	Fulfilled (yes / no)?	Comment
3.2	Only one instance of the function block SF_CYCLIC_PM5XX_S_REC is used.		
3.3	FB SF_CYCLIC_PM5XX_S_REC: • The variable size of the source buffer at ADR(DATA) is set to the maximum amount of data.		
3.4	FB SF_CYCLIC_PM5XX_S_REC: • Input DATA is not changed at runtime.		
3.5	FB SF_CYCLIC_PM5XX_S_REC: • The destination memory area is fixed for the active instance of the function block.		
3.6	FB SF_CYCLIC_PM5XX_S_REC: • The data length set at DATA_LEN is constant for the active instance of the function block.		
3.7	FB SF_CYCLIC_PM5XX_S_REC: • The data length set at DATA_LEN is set with SIZEOF() of the used variable.		
4. Testing			
4.1	Testing procedure according to chapter 7 has been performed.		
Reviewer(s): Machine/Application <ID>: Signature: Date:			

7. Test example

The following test example can be used as a basis for validation purpose.

⚠ DANGER

To cover all scenarios described in chapter 5, it is mandatory to install the pattern test described below. Only if the pattern is located at the last array entry on both sides (safety CPU and PM5xx), and is supervised vice versa, this pattern test ensures that data exchange from safety CPU to PM5xx (respectively vice versa) works correctly and does not overwrite safety application data in the safety CPU.

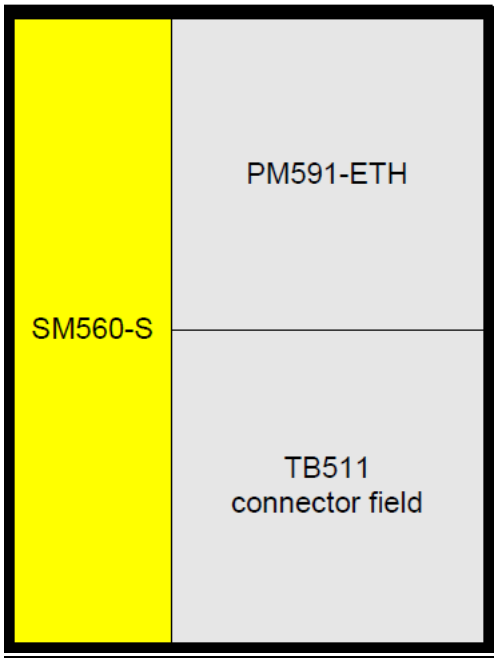
Furthermore and independently of the validation test, fulfilment of the checklist items defined in chapter 6.2 remains mandatory.

7.1. Overview

The following example describes a test implementation which performs a simple “pattern” test.

The main purpose of this test is to check that the data to be transferred using cyclic non-safe data exchange between the safety CPU and PM5xx works as expected and that the complete data buffers are transferred between them as configured (no loss of cyclic non-safe data).

Exemplary test setup:



Test functionality:

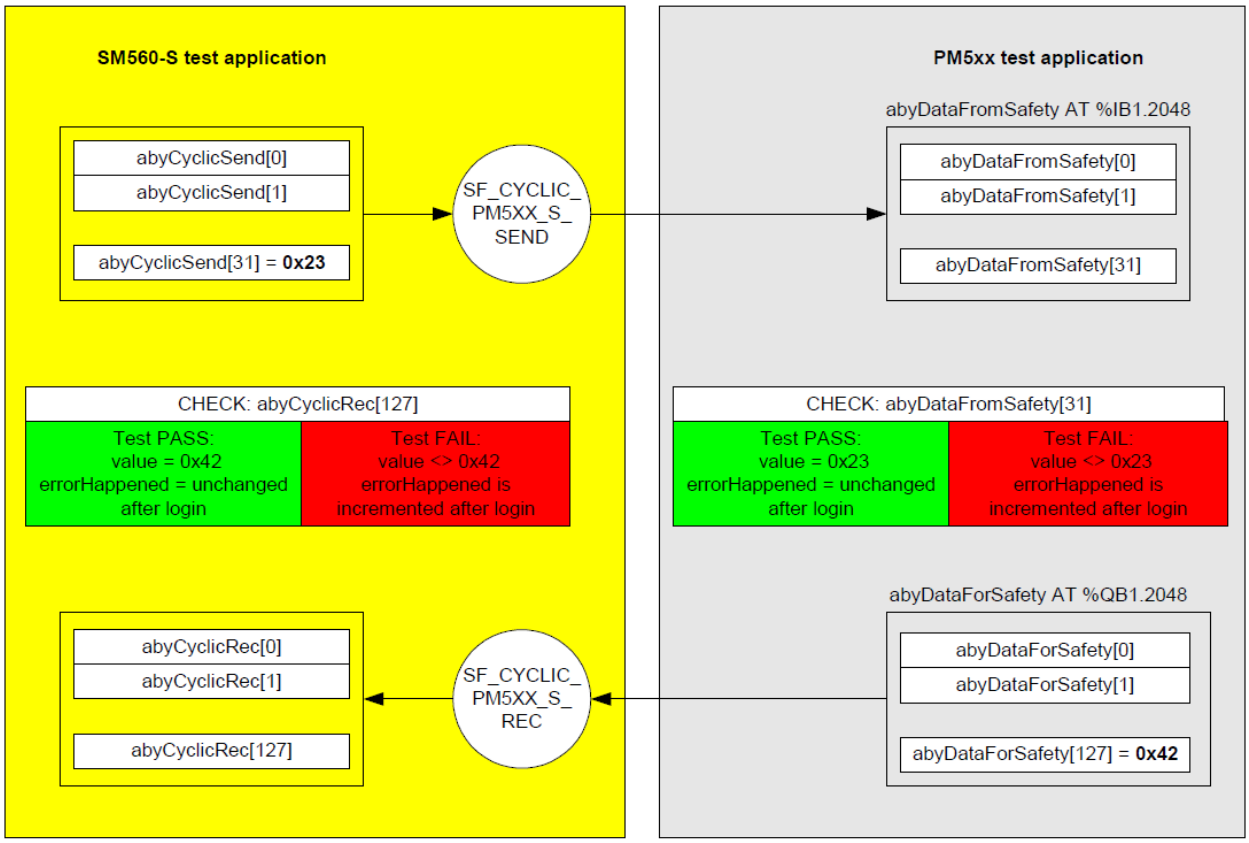
As an example, the safety CPU transmits 32 bytes to the PM5xx and receives 128 bytes from the PM5xx.

A special pre-defined number is placed in the last byte of each buffer. This value is checked for correct reception on both the safety CPU and PM5xx:

- PM5xx transmits value 0x42 at buffer offset 127.
- The safety CPU transmits value 0x23 at buffer offset 31.

The safety CPU cyclically checks the value at buffer offset 127 against the expected value (0x42). If unequal, an error counter “errorHappened” is incremented (Note that the check shall be done on the PM5xx side as well).

The figure below provides an overview of exchanged data between PM5xx and the safety CPU:



Due to asynchronous startup timing behaviour in the safety CPU and PM5xx and further project-specific settings (e.g. “Minimal update time” settings for communication modules), the error counters may be still unintentionally incremented until both CPUs are in the run mode.

Pass criteria:

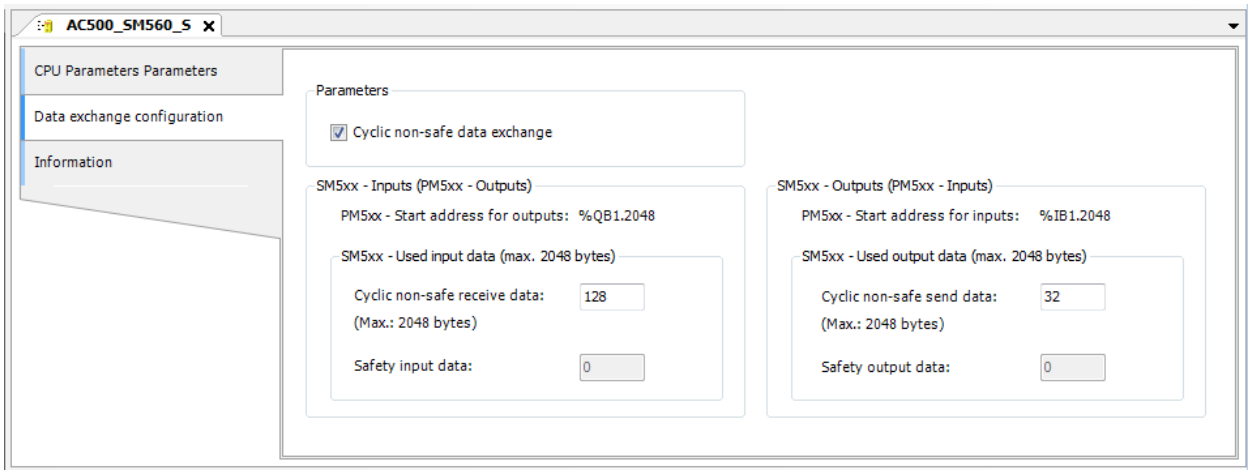
- After login, the error counters on both CPUs remain unchanged (error counter values before the run mode shall be ignored in the analysis).

Fail criteria:

- After login in the run mode, at least one error counter does not remain unchanged and is incremented in each processing cycle.

7.2. Automation Builder configuration

The following exemplary configuration of “Data exchange configuration” was used in the test:



Note: If not mentioned explicitly, default settings for configuration of the safety CPU and PM591-ETH) are used.

Device topology

- PM591-ETH non-safety CPU
- AC500-S safety CPU (in coupler slot 1)

Safety CPU configuration „CPU parameters configuration“ (Automation Builder)

- Debug Mode = “ON”

Safety CPU configuration „Data exchange configuration“ (Automation Builder)

- Cyclic non-safe data exchange enabled
- 128 bytes cyclic non-safe receive data
- 32 bytes cyclic non-safe send data

7.3. Safety test application (overview)

PLC_PRG (PRG-ST)

```
0001 PROGRAM PLC_PRG
0002 VAR
0003     sf_cyclic_rec        : SF_CYCLIC_PM5XX_S_REC;
0004     sf_cyclic_send       : SF_CYCLIC_PM5XX_S_SEND;
0005     abyCyclicSend        : ARRAY[0..31] OF BYTE;
0006     abyCyclicRec         : ARRAY[0..127] OF BYTE;
0007     errorHappened        : INT:= 0;
0008
0009     fbWD                 : SF_WDOG_TIME_SET;
0010 END_VAR
```

```
0001 fbWD(EN:=TRUE , WDOG:=100 , RESET:= , DONE=> , ACT_TIME=> , MAX_TIME=> );
0002
0003 sf_cyclic_rec(EN:=TRUE , DATA:=ADR(abyCyclicRec) , DATA_LEN:= SIZEOF(abyCyclicRec) , DONE=> , ERR=> , ERNO=> );
0004 IF (abyCyclicRec[SIZEOF(abyCyclicRec)-1] <> 16#42) OR sf_cyclic_rec.ERR THEN
0005     errorHappened := errorHappened + 1;
0006 END_IF
0007
0008 abyCyclicSend[SIZEOF(abyCyclicSend)-1]:= 16#23;
0009
0010 sf_cyclic_send(EN:=TRUE , DATA:=ADR(abyCyclicSend) , DATA_LEN:= SIZEOF(abyCyclicSend), DONE=> , ERR=> , ERNO=> );
0011 IF sf_cyclic_send.ERR THEN
0012     errorHappened := errorHappened + 1;
0013 END_IF
0014
0015
0016
```

7.4. Non-safety test application (overview)

PLC_PRG (PRG-ST)

```
0001 PROGRAM PLC_PRG
0002 VAR
0003     abyDataFromSafety AT %IB1.2048     : ARRAY[0..31] OF BYTE;
0004     abyDataForSafety AT %QB1.2048      : ARRAY[0..127] OF BYTE;
0005     errorHappened                      : UINT;
0006 END_VAR
0007
```

```
0001 IF (abyDataFromSafety[SIZEOF(abyDataFromSafety)-1] <> 16#23) THEN
0002     errorHappened := errorHappened + 1;
0003 END_IF
0004
0005 abyDataForSafety[SIZEOF(abyDataForSafety)-1] := 16#42;
0006
0007
0008
0009
```

8. Troubleshooting

The following table describes a list of known issues and solutions to help the user to fix potential problems which can occur using the cyclic non-safe data exchange.

If some of problems persist, please contact ABB technical support.

NOTICE

Before any troubleshooting, make sure that the checklist (see chapter 6.2) has been correctly filled out.

ID	Behaviour	Potential cause	Remedy
1.	No communication	FBs are not configured or configured wrongly	Check if FBs are configured correctly
		FB input “EN” is not “TRUE”	Check if FB input signal “EN” is “TRUE”
		Configuration files have not been updated	Create configuration data
		Error messages on FB output signal “ERR” respectively “ERNO”	Check error messages
2.	Communication ok, but data does not change	FBs are not called cyclically	Ensure that the FBs are called cyclically
3.	Update of exchanged cyclic non-safe data seems to be delayed or old values occur	Processing sequence not OK	Check if SF_CYCLIC_PM5XX_S_REC FB is called at the be-ginning of the safety CPU cycle, SF_CYCLIC_PM5XX_S_SEND FB at the end of the safety CPU cycle.
4.	AC500-S safety CPU cycle time too high for the given application	Amount of cyclic non-safe data is too big.	Check if configured size is really necessary for the particular use case. Reduce safety I/O size to increase performance.

3ADR025195M0202, REV D, en_US, 2023/04/26

ABB AG
Eppelheimer Straße 82
69123 Heidelberg, Germany
Phone: +49 62 21 701 1444
Fax: +49 62 21 701 1382
Mail: plc.support@de.abb.com
www.abb.com/plc