SIEMENS

SIMATIC

System Software for S7-300/400 System and Standard Functions

Reference Manual

Edition 11/2001
A5E00069892-04

Safety Guidelines

This manual contains notices intended to ensure personal safety, as well as to protect the products and connected equipment against damage. These notices are highlighted by the symbols shown below and graded according to severity by the following texts:

Danger
indicates that death, severe personal injury or substantial property damage will result if proper precautions are not taken.

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

Caution
indicates that minor personal injury can result if proper precautions are not taken.

Caution
indicates that property damage can result if proper precautions are not taken.

Notice
draws your attention to particularly important information on the product, handling the product, or to a particular part of the documentation.

Qualified Personnel

Only **qualified personnel** should be allowed to install and work on this equipment. Qualified persons are defined as persons who are authorized to commission, to ground and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

Correct Usage

Note the following:

Warning
This device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

Trademarks

SIMATIC®, SIMATIC HMI® and SIMATIC NET® are registered trademarks of SIEMENS AG.

Third parties using for their own purposes any other names in this document which refer to trademarks might infringe upon the rights of the trademark owners.

Disclaimer of Liability

We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

Siemens AG
Bereich Automatisierungs- und Antriebstechnik
Geschaeftsgebiet Industrie-Automatisierungssysteme
Postfach 4848, D- 90327 Nuernberg

Preface

Purpose

This manual provides you with a comprehensive overview of the organization blocks (OB), system functions (SFC), system and standard function blocks (SFC), and IEC functions contained in the operating systems of the CPUs of the S7-300 and S7-400. The appendix describes the diagnostic data, system status lists (SZL), and events.

Note

Refer to the reference section of the "*S7-300 Programmable Controller, Hardware and Installation"* manual ***/70/*** or the "*S7-400/M7-400 Programmable Controllers* Module Specifications" reference manual ***/101/*** or the Instruction List: *S7-400 Programmable Controller* ***/102/*** (whichever version applies to your CPU) for details of which of these functions and blocks are available on which CPU. The properties of the CFBs and the S7 signaling functions for specific CPUs are described in ***/70/*** and ***/101/***.

For information about the CPU operating systems, program design, and the communications and diagnostic capabilities of the CPUs, refer to the "*Configuring Hardware and Communication Connections STEP 7 V5.1"* manual ***/234/*** How to call functions and function blocks in your program is explained in the language descriptions.

You program and assign parameters for all these functions using the STEP 7 standard software. How to use this software is described in the "*Programming with STEP 7 V5.1*" manual ***/231/*** and in the STEP 7 online help.

Audience

This manual is intended for programmers and engineers who are familiar with controlling processes and are responsible for writing programs for programmable logic controllers.

STEP 7 Documentation Packages

The print edition 06/2000 of this is part of the documentation package "STEP 7 Basic Information."

The following table displays an overview of the STEP 7 documentation:

Documentation	Purpose	Order Number
STEP 7 Basic Information with • Working with STEP 7 V5.1, Getting Started Manual • Programming with STEP 7 V5.1 • Configuring Hardware and Communication Connections, STEP 7 V5.1 • From S5 to S7, Converter Manual	Basic information for technical personnel describing the methods of implementing control tasks with STEP 7 and the S7-300/400 programmable controllers.	6ES7810-4CA05-8BA0
STEP 7 Reference with • Ladder Logic (LAD)/Function Block Diagram (FBD)/Statement List (STL) for S7-300/400 manuals • Standard and System Functions for S7-300/400	Provides reference information and describes the programming languages LAD, FBD, and STL, and standard and system functions extending the scope of the STEP 7 basic information.	6ES7810-4CA05-8BR0

Online Helps	Purpose	Order Number
Help on STEP 7	Basic information on programming and configuring hardware with STEP 7 in the form of an online help.	Part of the STEP 7 Standard software.
Reference helps on STL/LAD/FBD Reference help on SFBs/SFCs Reference help on Organization Blocks	Context-sensitive reference information.	Part of the STEP 7 Standard software.

Online Help

The manual is complemented by an online help which is integrated in the software. This online help is intended to provide you with detailed support when using the software.

The help system is integrated in the software via a number of interfaces:

- There are several menu commands which you can select in the **Help** menu: The **Contents** command opens the index for the Help on Step 7.
- **Using Help** provides detailed instructions on using the online help.
- The context-sensitive help offers information on the current context, for example, an open dialog box or an active window. You can open the context-sensitive help by clicking the "Help" button or by pressing F1.
- The status bar offers another form of context-sensitive help. It displays a short explanation for each menu command when the mouse pointer is positioned on the menu command.
- A brief explanation is also displayed for each icon in the toolbar when the mouse pointer is positioned on the icon for a short time.

If you prefer to read the information from the online help in printed format, you can print out individual help topics, books, or the entire online help.

This manual is an extract from the HTML-based Help on STEP 7. As the manual and the online help share an almost identical structure, it is easy to switch between the manual and the online help.

Feedback on Documentation

To help us to provide the best possible documentation for you and future STEP 7 users, we need your support. If you have any comments or suggestions relating to this *manual* or the *online help*, please complete the questionnaire at the end of the manual and send it to the address shown. Please include your own personal rating of the documentation.

Other Manuals

The various S7-300 and S7-400 CPUs and the S7-300 and S7-400 modules are described in the following manuals:

- For the S7-300 programmable logic controller, refer to the manuals: "*S7 300 Programmable Controller, Hardware and Installation"* ***/70/***, "*S7-300, M7-300 Programmable Controllers Module Specifications*" ***/71/*** and in the Instruction List ***/72/***.
- For the S7-400 programmable logic controller, refer to the manual: "*S7-400/M7-400 Programmable Controllers* Module Specifications" ***/101/*** and in the Instruction List ***/102/***.

How to Use this Manual

This manual covers the following topics:

- Chapter 1 explains the functions of all the organization blocks.
- Chapter 2 describes the common parameters RET_VAL, REQ and BUSY.
- Chapters 3 to 28 describe the SFCs, SFBs and IEC-FCs.
- The Chapters sections 29 to 32 contain a description of the structure of the diagnostic data, an overview of the SZL-IDs, the possible events, lists of the SFCs, SFBs and FCs described in this manual, an overview of the SDBs.
- The bibliography contains a list of further manuals.
- The Glossary explains important terminology.
- The Index helps you to locate sections of text and topics quickly.

Conventions

References to other manuals and documentation are indicated by numbers in slashes /.../. These numbers refer to the titles of manuals listed in the bibliography.

Special Note

The system functions can be interrupted. If there are any restrictions that apply to certain SFCs or situations, these are explained in the description of the particular SFC.

Documentation Reply Form

If you have comments about this manual or the online help, please fill out the questionnaire at the end of this manual and send it the address shown. Please take the time to add your own evaluation grade.

http://www.ad.siemens.de/partner

Training Centers

Siemens offers a number of training courses to familiarize you with the SIMATIC S7 automation system. Please contact your regional training center or our central training center in D 90327 Nuremberg, Germany for details:
Telephone: +49 (911) 895-3200.

http://www.sitrain.com/

SIMATIC Documentation on the Internet

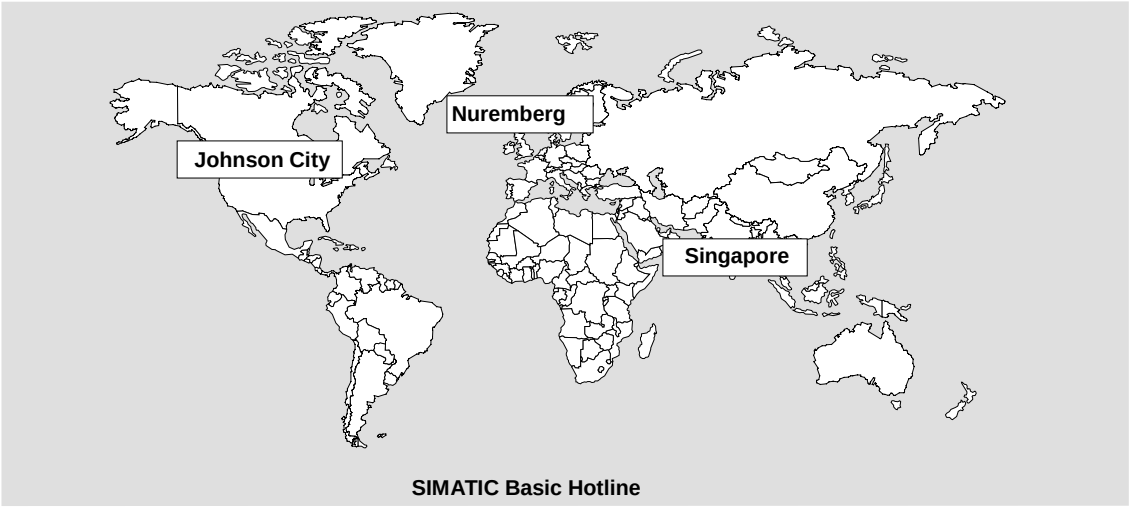
You will find the documentation on the internet at:

http://www.ad.siemens.de/support

Use the Knowledge Manager to find the documentation you need quickly. If you have any questions or suggestions concerning the documentation you can use the "Documentation" conference in the internet forum.

Automation and Drives, Service & Support

Available worldwide, around the clock:



SIMATIC Basic Hotline

Worldwide (Nuremberg) **Technical Support**	**Worldwide** (Nuremberg) **Technical Support**	
(Free Contact) Local time: Mo.-Fr. 7:00 to 17:00 Phone: +49 (180) 5050 222 Fax: +49 (180) 5050 223 E-mail: techsupport@ad.siemens.de GMT: +1:00	(charged, only with SIMATIC Card) Local time: Mo.-Fr. 0:00 to 24:00 Phone: +49 (911) 895-7777 Fax: +49 (911) 895-7001 GMT: +01:00	
Europe / Africa (Nuremberg) **Authorization**	**America** (Johnson City) **Technical Support and Authorization**	**Asia / Australia** (Singapore) **Technical Support and Authorization**
Local time: Mo.-Fr. 7:00 to 17:00 Phone: +49 (911) 895-7200 Fax: +49 (911) 895-7201 E-mail: authorization@nbgm.siemens.de GMT: +1:00	Local time: Mo.-Fr. 8:00 to 19:00 Phone: +1 423 461-2522 Fax: +1 423 461-2289 E-mail: simatic.hotline@sea.siemens.com GMT: -5:00	Local time: Mo.-Fr. 8:30 to 17:30 Phone: +65 740-7000 Fax: +65 740-7001 E-mail: simatic.hotline@sae.siemens.com.sg GMT: +8:00

German and English are spoken on all the SIMATIC hotlines, French, Italian and Spanish are also spoken on the authorization hotline.

Service & Support on the Internet

In addition to our documentation, we offer our Know-how online on the internet at:

http://www.ad.siemens.de/support

where you will find the following:

- Current Product Information leaflets, FAQs (Frequently Asked Questions), Downloads, Tips and Tricks.
- A newsletter giving you the most up-to-date information on our products.
- The Knowledge Manager helps you find the documents you need.
- Users and specialists from all over the world share information in the forum.
- Your local customer service representative for Automation & Drives in our customer service representative data bank.
- Information on field service, repairs, spare parts and more under "Services".

Contents

1 Organization Blocks

1.1 Overview of the Organization Blocks (OBs)

What Are Organization Blocks?

Organization Blocks (OBs) are the interface between the operating system of the CPU and the user program. OBs are used to execute specific program sections:

- At the startup of the CPU
- In a cyclic or clocked execution
- Whenever errors occur
- Whenever hardware interrupts occur.

Organization blocks are executed according to the priority they are allocated.

Which OBs Are Available?

Not all CPUs can process all of the OBs available in STEP 7. Consult the data sheets for your CPU to determine which OBs are included with your CPU.

Where to Find More Information?

Refer to the online help and the following manuals for more information:

- **/70/**: this manual contains the data sheets that describe the capabilities of the different S7-300 CPUs. This also includes the possible start events for each OB.
- **/101/**: this manual contains the data sheets that describe the capabilities of the different S7-400 CPUs. This also includes the possible start events for each OB.

The following table contains the start event belonging to each OB as well as the default priority class.

OB	Start Event	Default Priority Class	Explanation
OB1	End of startup or end of OB1	1	Free cycle
OB10	Time-of-day interrupt 0	2	No default time specified
OB11	Time-of-day interrupt 1	2	
OB12	Time-of-day interrupt 2	2	
OB13	Time-of-day interrupt 3	2	
OB14	Time-of-day interrupt 4	2	
OB15	Time-of-day interrupt 5	2	
OB16	Time-of-day interrupt 6	2	
OB17	Time-of-day interrupt 7	2	
OB20	Time-delay interrupt 0	3	No default time specified
OB21	Time-delay interrupt 1	4	
OB22	Time-delay interrupt 2	5	
OB23	Time-delay interrupt 3	6	
OB30	Cyclic interrupt 0 (default interval: 5 s)	7	Cyclic interrupts
OB31	Cyclic interrupt 1 (default interval: 2 s)	8	
OB32	Cyclic interrupt 2 (default interval: 1 s)	9	
OB33	Cyclic interrupt 3 (default interval: 500 ms)	10	
OB34	Cyclic interrupt 4 (default interval: 200 ms)	11	
OB35	Cyclic interrupt 5 (default interval: 100 ms)	12	
OB36	Cyclic interrupt 6 (default interval: 50 ms)	13	
OB37	Cyclic interrupt 7 (default interval: 20 ms)	14	
OB38	Cyclic interrupt 8 (default interval: 10 ms)	15	
OB40	Hardware interrupt 0	16	Hardware interrupts
OB41	Hardware interrupt 1	17	
OB42	Hardware interrupt 2	18	
OB43	Hardware interrupt 3	19	
OB44	Hardware interrupt 4	20	
OB45	Hardware interrupt 5	21	
OB46	Hardware interrupt 6	22	
OB47	Hardware interrupt 7	23	
OB55	Status interrupt	2	DPV1 interrupts
OB56	Update interrupt	2	
OB57	Manufacturer specific interrupt	2	
OB60	SFC35 "MP_ALM" call	25	Multicomputing interrupt
OB70	I/O redundancy error (only in H CPUs)	25	Redundancy error interrupts
OB72	CPU redundancy error (only in H CPUs)	28	
OB 73	Communication redundancy error OB (only in H CPUs)	25	

OB	Start Event	Default Priority Class	Explanation
OB80	Time error	26, 28 [1]	Asynchronous error interrupts
OB81	Power supply fault	25, 28 [1]	
OB82	Diagnostic interrupt	25, 28 [1]	
OB83	Insert/remove-module interrupt	25, 28 [1]	
OB84	CPU hardware fault	25, 28 [1]	
OB85	Program error	25, 28 [1]	
OB86	Failure of an expansion rack, DP master system or station for distributed I/Os	25, 28 [1]	
OB87	Communication error	25, 28 [1]	
OB90	Warm or cold restart or delete a block being executed in OB90 or load an OB90 on the CPU or terminate OB90	29 [2]	Background cycle
OB100	Warm restart	27 [1]	Startup
OB101	Hot restart	27 [1]	
OB102	Cold restart	27 [1]	
OB121	Programming error	Priority of the OB causing the error	Synchronous error interrupts
OB122	I/O access error	Priority of the OB causing the error	

[1] Priority classes 27 and 28 are valid in the priority class model of the startup.

[2] Priority class 29 corresponds to priority 0.29. This means that the background cycle has lower priority than the free cycle.

1.2 Program Cycle Organization Block (OB1)

Description

The operating system of the S7 CPU executes OB1 periodically. When OB1 has been executed, the operating system starts it again. Cyclic execution of OB1 is started after the startup has been completed. You can call other function blocks (FBs, SFBs) or functions (FCs, SFCs) in OB1.

Understanding the Operation of OB1

OB1 has the lowest priority of all of the OBs whose run-times are monitored, in other words, all of the other OBs except OB90 can interrupt the execution of OB1. The following events cause the operating system to call OB1:

- The startup is completed.
- The execution of OB1 (the previous cycle) has finished.

When OB1 has been executed, the operating system sends global data. Before restarting OB1, the operating system writes the process-image output table to the output modules, updates the process-image input table and receives any global data for the CPU.

S7 monitors the maximum scan time, ensuring a maximum response time. The value for the maximum scan time is preset to 150 ms. You can set a new value or you can restart the time monitoring anywhere within your program with SFC43 "RE_TRIGR." If your program exceeds the maximum cycle time for OB1, the operating system calls OB80 (time error OB); if OB80 is not programmed, the CPU changes to the STOP mode.

Apart from monitoring the maximum scan time, it is also possible to guarantee a minimum scan time. The operating system will delay the start of a new cycle (writing of the process image output table to the output modules) until the minimum scan time has been reached.

Refer to the manuals /**70**/ and /**101**/ for the ranges of the parameters "maximum" and "minimum" scan time. You change parameter settings using STEP 7.

Local Data for OB1

The following table describes the temporary (TEMP) variables for OB1. The variable names are the default names of OB1.

Variable	Type	Description
OB1_EV_CLASS	BYTE	Event class and identifiers: B#16#11: OB1 active
OB1_SCAN_1	BYTE	• B#16#01: completion of a warm restart • B#16#02: completion of a hot restart • B#16#03: completion of the main cycle • B#16#04: completion of a cold restart • B#16#05: first OB1 cycle of the new master CPU after master-reserve switchover and STOP of the previous master
OB1_PRIORITY	BYTE	Priority class 1
OB1_OB_NUMBR	BYTE	OB number (01)
OB1_RESERVED_1	BYTE	Reserved
OB1_RESERVED_2	BYTE	Reserved
OB1_PREV_CYCLE	INT	Run time of previous scan (ms)
OB1_MIN_CYCLE	INT	Minimum cycle time (ms) since the last startup
OB1_MAX_CYCLE	INT	Maximum cycle time (ms) since the last startup
OB1_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

1.3 Time-of-Day Interrupt Organization Blocks (OB10 to OB17)

Description

STEP 7 provides up to eight OBs (OB10 to OB17) which can be run once or periodically. You can assign parameters for CPU using SFCs or STEP 7 so that these OBs are processed at the following intervals:

- Once
- Every minute
- Hourly
- Daily
- Weekly
- Monthly
- At the end of each month

Understanding the Operation of Time-of-Day Interrupt OBs

To start a time-of-day interrupt, you must first set and then activate the interrupt. The three following start possibilities exist:

- Automatic start of the time-of-day interrupt. This occurs once you have set and then activated the time-of-day interrupt with STEP 7. The following table shows the basic possibilities for activating a time-of-day interrupt with STEP 7.
- You set the time-of-day interrupt with STEP 7 and then activate it by calling SFC30 "ACT-TINT" in your program.
- You set the time-of-day interrupt by calling SFC28 "SET_TINT" and then activate it by calling SFC30 "ACT_TINT."

Interval	Description
Not activated	The time-of-day interrupt is not executed, even when loaded in the CPU. It can be activated by calling SFC30.
Activated once only	The time-of-day OB is canceled automatically after it runs the one time specified. Your program can use SFC28 and SFC30 to reset and reactivate the OB.
Activated periodically	When the time-of-day interrupt occurs, the CPU calculates the next start time for the time-of-day interrupt based on the current time of day and the period.

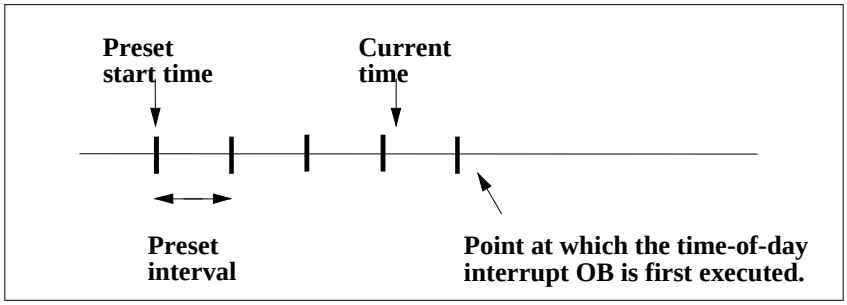
The behavior of the time-of-day interrupt when you move the clock forwards or backwards is described in /**234**/.

Note

If you configure a time-of-day interrupt in such a way that the corresponding OB is to be processed once, the DATE_AND_TIME must not be in the past (relative to the real-time clock of the CPU).

If you configure a time-of-day interrupt in such a way that the corresponding OB is to be processed periodically, the start DATE_AND_TIME, however, are in the past, then the time-of-day interrupt will be processed the next time it is due. This is illustrated in the following figure.

You can disable or delay and re-enable time-of-day interrupts using SFCs 39 to 42.



Conditions That Affect Time-of-Day Interrupt OBs

Since a time-of-day interrupt occurs only at specified intervals, certain conditions can affect the operation of the OB during the execution of your program. The following table shows some of these conditions and describes the effect on the execution of the time-of-day interrupt OB.

Condition	Result
Your program calls SFC29 (CAN_TINT) and cancels a time-of-day interrupt.	The operating system clears the start event (DATE_AND_TIME) for the time-of-day interrupt. You must set the start event again and activate it before the OB can be called again.
Your program attempted to activate a time-of-day interrupt OB, but the OB was not loaded on the CPU.	The operating system calls OB85. If OB85 has not been programmed (loaded on the CPU), the CPU changes to the STOP mode.
When synchronizing or correcting the system clock of the CPU, you set the time ahead and skipped the start event date or time for the time-of-day OB.	The operating system calls OB80 and encodes the number of the time-of-day OB and the start event information in OB80. The operating system then runs the time-of-day OB once, regardless of the number of times that this OB should have been executed. The start event information of OB80 shows the DATE_AND_TIME that the time-of-day OB was first skipped.
When synchronizing or correcting the system clock of the CPU, the time was set back so that the start event, date, or time for the OB is repeated.	If the time-of-day OB had already been activated before the clock was set back, it is not called again.

Condition	Result
The CPU runs through a warm or cold restart.	Any time-of-day OB that was configured by an SFC is changed back to the configuration that was specified in STEP 7. If you have configured a time-of-day interrupt for a one-time start of the corresponding OB, set it with STEP 7, and activated it, the OB is called once after a warm or cold restart of the operating system, if the configured start time is in the past (relative to the real-time clock of the CPU).
A time-of-day OB is still being executed when the start event for the next interval occurs.	The operating system calls OB80. If OB80 is not programmed, the CPU changes to the STOP mode. If OB80 is loaded, both OB80 and the time-of-day interrupt OB are first executed and then second the requested interrupt is executed.

Local Data for Time-of-Day Interrupt OBs

The following table describes the temporary (TEMP) variables for a time-of-day interrupt OB. The variable names are the default names of OB10.

Variable	Type	Description
OB10_EV_CLASS	BYTE	Event class and identifiers: B#16#11 = interrupt is active
OB10_STRT_INFO	BYTE	B#16#11: start request for OB10 (B#16#12: start request for OB11) : : (B#16#18: start request for OB17)
OB10_PRIORITY	BYTE	Assigned priority class; default 2
OB10_OB_NUMBR	BYTE	OB number (10 to 17)
OB10_RESERVED_1	BYTE	Reserved
OB10_RESERVED_2	BYTE	Reserved
OB10_PERIOD_EXE	WORD	The OB is executed at the specified intervals: W#16#0000: once W#16#0201: once every minute W#16#0401: once hourly W#16#1001: once daily W#16#1201: once weekly W#16#1401: once monthly W#16#1801: once yearly W#16#2001: end of month
OB10_RESERVED_3	INT	Reserved
OB10_RESERVED_4	INT	Reserved
OB10_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

1.4 Time-Delay Interrupt Organization Blocks (OB20 to OB23)

Description

S7 provides up to four OBs (OB20 to OB23) which are executed after a specified delay. Every time-delay OB is started by calling SFC32 (SRT_DINT). The delay time is an input parameter of the SFC.

When your program calls SFC32 (SRT_DINT), you provide the OB number, the delay time, and a user-specific identifier. After the specified delay, the OB starts. You can also cancel the execution of a time-delay interrupt that has not yet started.

Understanding the Operation of Time-Delay Interrupt OBs

After the delay time has expired (value in milliseconds transferred to SFC32 together with an OB number), the operating system starts the corresponding OB.

To use the time-delay interrupts, you must perform the following tasks:

- You must call SFC32 (SRT_DINT).
- You must download the time-delay interrupt OB to the CPU as part of your program.

Time-delay OBs are executed only when the CPU is in the RUN mode. A warm or a cold restart clears any start events for the time-delay OBs. If a time-delay interrupt has not started, you can use SFC33 (CAN_DINT) to cancel its execution.

The delay time has a resolution of 1 ms. A delay time that has expired can be started again immediately. You can query the status of a delay-time interrupt using SFC34 (QRY_DINT).

The operating system calls an asynchronous error OB if one of the following events occur:

- If the operating system attempts to start an OB that is not loaded and you specified its number when calling SFC32 "SRT_DINT."
- If the next start event for a time-delay interrupt occurs before the time-delay OB has been completely executed.

You can disable or delay and re-enable delay interrupts using SFCs 39 to 42.

Local Data for Time-Delay Interrupt OBs

The following table describes the temporary (TEMP) variables for a time-delay interrupt OB. The variable names are the default names of OB20.

Variable	Type	Description
OB20_EV_CLASS	BYTE	Event class and identifiers: B#16#11: interrupt is active
OB20_STRT_INF	BYTE	B#16#21: start request for OB20 (B#16#22: start request for OB21) (B#16#23: start request for OB22) (B#16#24: start request for OB23)
OB20_PRIORITY	BYTE	Assigned priority class: default values 3 (OB20) to 6 (OB23)
OB20_OB_NUMBR	BYTE	OB number (20 to 23)
OB20_RESERVED_1	BYTE	Reserved
OB20_RESERVED_2	BYTE	Reserved
OB20_SIGN	WORD	User ID: input parameter SIGN from the call for SFC32 (SRT_DINT)
OB20_DTIME	TIME	Elapsed delay time in ms
OB20_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

1.5 Cyclic Interrupt Organization Blocks (OB30 to OB38)

Description

S7 provides up to nine cyclic interrupt OBs (OB30 to OB38) which interrupt your program at fixed intervals. The following table shows the default intervals and priority classes for the cyclic interrupt OBs.

OB Number	Default Interval	Default Priority Class
OB30	5 s	7
OB31	2 s	8
OB32	1 s	9
OB33	500 ms	10
OB34	200 ms	11
OB35	100 ms	12
OB36	50 ms	13
OB37	20 ms	14
OB38	10 ms	15

Understanding the Operation of Cyclic Interrupt OBs

The equidistant start times of the cyclic interrupt OBs are determined by the interval and the phase offset. Refer to /**234**/ for the relationship between the start time, time cycle, and phase offset of an OB.

Note

You must make sure that the run time of each cyclic interrupt OB is significantly shorter than its interval. If a cyclic interrupt OB has not been completely executed before it is due for execution again because the interval has expired, the time error OB (OB80) is started. The cyclic interrupt that caused the error is executed later.

You can disable or delay and re-enable cyclic interrupts using SFCs 39 to 42

Refer to the specifications of your specific CPU for the range of the parameters interval, priority class, and phase offset. You can change the parameter settings using STEP 7.

Local Data for Cyclic Interrupt OBs

The following table describes the temporary (TEMP) variables for a cyclic interrupt OB. The variable names are the default names of OB35.

Variable	Type	Description
OB35_EV_CLASS	BYTE	Event class and identifiers B#16#11: interrupt is active
OB35_STRT_INF	BYTE	B#16#30: Start request for cyclic interrupt OB with special criteria (only for H-CPUs and there only if explicitly configured for them) B#16#31 : start request for OB30 : B#16#36 : start request for OB35 : B#16#39 : start request for OB38
OB35_PRIORITY	BYTE	Assigned priority class: defaults 7 (OB30) to 15 (OB38)
OB35_OB_NUMBR	BYTE	OB number (30 to 38)
OB35_RESERVED_1	BYTE	Reserved
OB35_RESERVED_2	BYTE	Reserved
OB35_PHASE_OFFSET	WORD	Phase offset [ms]
OB35_RESERVED_3	INT	Reserved
OB35_EXC_FREQ	INT	Interval in milliseconds
OB35_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

1.6 Hardware Interrupt Organization Blocks (OB40 to OB47)

Description

S7 provides up to eight independent hardware interrupts each with its own OB.

By assigning parameters with STEP 7, you specify the following for each signal module that will trigger hardware interrupts:

- Which channels trigger a hardware interrupt under what conditions.
- Which hardware interrupt OB is assigned to the individual groups of channels (as default, all hardware interrupts are processed by OB40).

With CPs and FMs, you assign these parameters using their own software.

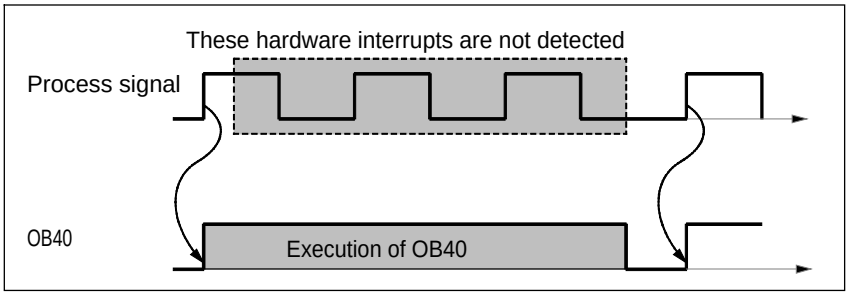
You select the priority classes for the individual hardware interrupt OBs using STEP 7.

Understanding the Operation of Hardware Interrupt OBs

After a hardware interrupt has been triggered by the module, the operating system identifies the slot and the corresponding hardware interrupt OB. If this OB has a higher priority than the currently active priority class, it will be started. The channel-specific acknowledgement is sent after this hardware interrupt OB has been executed.

If another event that triggers a hardware interrupt occurs on the same module during the time between identification and acknowledgement of a hardware interrupt, the following applies:

- If the event occurs on the channel that previously triggered the hardware interrupt, then the new interrupt is lost. This is illustrated in the following figure based on the example of a channel of a digital input module. The triggering event is the rising edge. The hardware interrupt OB is OB40.



- If the event occurs on another channel of the same module, then no hardware interrupt can currently be triggered. This interrupt, however, is not lost, but is triggered after the acknowledgement of the currently active hardware interrupt.

If a hardware interrupt is triggered and its OB is currently active due to a hardware interrupt from another module, the new request is recorded and the OB processed when it is free.

You can disable or delay and re-enable hardware interrupts using SFCs 39 to 42.

You can assign parameters for the hardware interrupts of a module not only with STEP 7 but also with SFCs 55 to 57.

Local Data for Hardware Interrupt OBs

The following table describes the temporary (TEMP) variables for a hardware interrupt OB. The variable names are the default names of OB40.

Variable	Type	Description
OB40_EV_CLASS	BYTE	Event class and identifiers: B#16#11: interrupt is active
OB40_STRT_INF	BYTE	• B#16#41: interrupt via interrupt line 1 • B#16#42: interrupt via interrupt line 2 (only with an S7-400) • B#16#43: interrupt via interrupt line 3 (only with an S7-400) • B#16#44: interrupt via interrupt line 4 (only with an S7-400) • B#16#45: WinAC: interrupt triggered via PC
OB40_PRIORITY	BYTE	Assigned priority class: defaults 16 (OB40) to 23 (OB47)
OB40_OB_NUMBR	BYTE	OB number (40 to 47)
OB40_RESERVED_1	BYTE	Reserved
OB40_IO_FLAG	BYTE	Input module: B#16#54 Output module: B#16#55
OB40_MDL_ADDR	WORD	Logical base address of the module that triggers the interrupt
OB40_POINT_ADDR	DWORD	• For digital modules: bit field with the statuses of the inputs on the module(Bit 0 corresponds to the first input) • For analog modules: Bit field, informing which channel has exceeded which limit (for detailed info on the structure refer to /71/ or /101/). • For CPs or IMs: Module interrupt status (not user relevant)
OB40_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

Note

If you are using a DPV1 capable CPU you can obtain additional information on the interrupt with the help of SFB54 "RALRM" which exceeds the start information of the OB. This also applies when you operate the DP Master in S7 compatible mode.

1.7 Status Interrupt OB (OB 55)

Note

A status interrupt OB (OB 55) is only available for DPV1 capable CPUs.

Description

The CPU operating system calls OB55 if a status interrupt was triggered via the slot of a DPV1 slave. This might be the case if a component (module or rack) of a DPV1 slaves changes its operating mode, for example from RUN to STOP. For precise information on events that trigger a status interrupt, refer to the documentation of the DPV1 slave‘s manufacturer.

Local data of the status interrupt OB

The table below contains the temporary (TEMP) variables of the status interrupt OB. Selected variable names are the default names of OB55.

Variable	Data type	Description
OB55_EV_CLASS	BYTE	Event class and identifiers: B#16#11 (upcoming event)
OB55_STRT_INF	BYTE	B#16#55 (Start request for OB55)
OB55_PRIORITY	BYTE	Configured priority class, default values 2
OB55_OB_NUMBR	BYTE	OB number (55)
OB55_RESERVED_1	BYTE	Reserved
OB55_IO_FLAG	BYTE	Input module: B#16#54 Output module: B#16#55
OB55_MDL_ADDR	WORD	logical base address of the interrupt triggering component (module)
OB55_LEN	BYTE	Data block length supplied by the interrupt
OB55_TYPE	BYTE	ID for the interrupt type “Status interrupt”
OB55_SLOT	BYTE	slot number of the interrupt triggering component (module)
OB55_SPEC	BYTE	Specifier • Bit 0 to 1: Interrupt specifier • bit 2: Add_Ack • Bit 3 to 7: Seq. no.
OB55_DATE_TIME	DATE_AND_TIME	Date and time at which the OB was called

Note

You can obtain the full auxiliary information on the interrupt the DP message frame contains by calling SFB54 "RALRM" with OB55.

1.8 Update Interrupt OB (OB 56)

Note

A update interrupt OB (OB 56) is only available for DPV1 capable CPUs.

Description

The CPU operating system calls OB56 if an update interrupt was triggered via the slot of a DPV1 slave. This can be the case if you have changed the parameters for the slot of a DPV1 slave (via local or remote access). For precise information on events that trigger an update interrupt, refer to the documentation of the DPV1 slave‘s manufacturer.

Local data of the update interrupt OB

The table below contains the temporary (TEMP) variables of the update interrupt OB. Selected variable names are the default names of OB56.

Variable	Data type	Description
OB56_EV_CLASS	BYTE	Event class and identifiers: B#16#11 (upcoming event)
OB56_STRT_INF	BYTE	B#16#56 (Start request for OB56)
OB56_PRIORITY	BYTE	Configured priority class, default values 2
OB56_OB_NUMBR	BYTE	OB number (56)
OB56_RESERVED_1	BYTE	Reserved
OB56_IO_FLAG	BYTE	Input module: B#16#54 Output module: B#16#55
OB56_MDL_ADDR	WORD	logical base address of the interrupt triggering component (module)
OB56_LEN	BYTE	Data block length supplied by the interrupt
OB56_TYPE	BYTE	ID for the interrupt type “Update interrupt”
OB56_SLOT	BYTE	slot number of the interrupt triggering component (module)
OB56_SPEC	BYTE	Specifier • Bit 0 to 1: Interrupt specifier • bit 2: Add_Ack • Bit 3 to 7: Seq. no.
OB56_DATE_TIME	DATE_AND_TIME	Date and time at which the OB was called

Note

You can obtain the full auxiliary information on the interrupt the DP message frame contains by calling SFB54 "RALRM" with OB55.

1.9 Manufacturer Specific Interrupt OB (OB57)

Note

An OB for manufacturer specific interrupts(OB57) is only available for DPV1 capable CPUs.

Description

The CPU operating system calls OB57 if an manufacturer specific interrupt was triggered via the slot of a DPV1 slave.

Local data of the OB for manufacturer specific interrupts

The table below contains the temporary (TEMP) variables of the OB for manufacturer specific interrupt interrupts. Selected variable names are the default names of OB57.

Variable	Data type	Description
OB57_EV_CLASS	BYTE	Event class and identifiers: B#16#11 (upcoming event)
OB57_STRT_INF	BYTE	B#16#57 (Start request for OB57)
OB57_PRIORITY	BYTE	Configured priority class, default values 2
OB57_OB_NUMBR	BYTE	OB number (57)
OB57_RESERVED_1	BYTE	Reserved
OB57_IO_FLAG	BYTE	Input module: B#16#54 Output module: B#16#55
OB57_MDL_ADDR	WORD	logical base address of the interrupt triggering component (module)
OB57_LEN	BYTE	Data block length supplied by the interrupt
OB57_TYPE	BYTE	ID for the interrupt type “Manufacturer specific interrupt”
OB57_SLOT	BYTE	slot number of the interrupt triggering component (module)
OB57_SPEC	BYTE	Specifier • Bit 0 to 1: Interrupt specifier • bit 2: Add_Ack • Bit 3 to 7: Seq. no.
OB57_DATE_TIME	DATE_AND_TIME	Date and time at which the OB was called

Note

You can obtain the full auxiliary information on the interrupt the DP message frame contains by calling SFB54 "RALRM" with OB57.

1.10 Multicomputing Interrupt Organization Block (OB60)

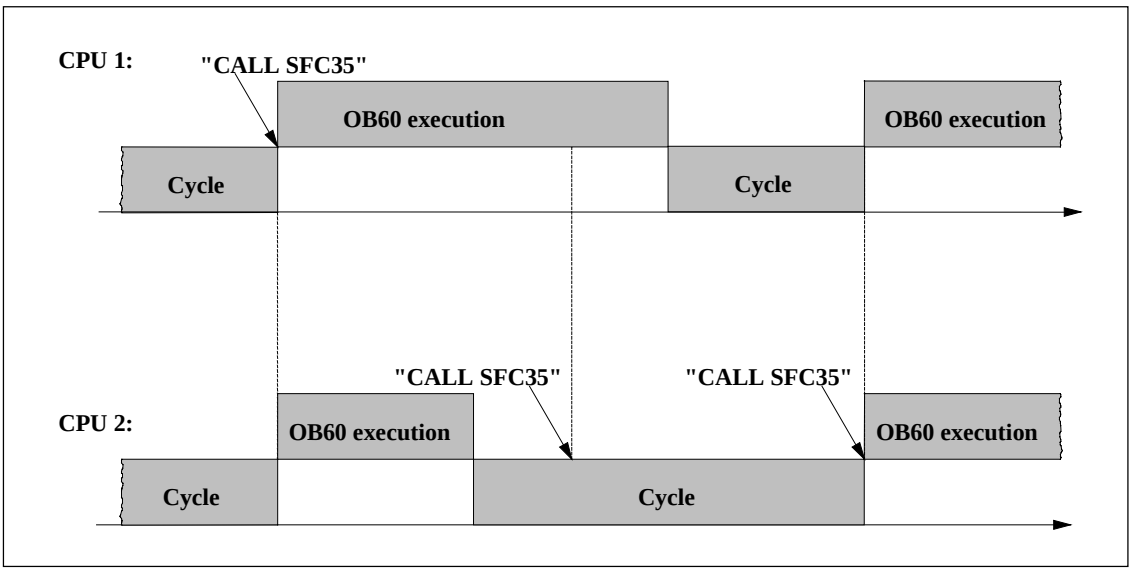
Description

Using the multicomputing interrupt, you can make sure that the reaction of the CPUs is synchronized to an event during multicomputing. In contrast to hardware interrupts triggered by signal modules, the multicomputing interrupt can only be output by CPUs.

Understanding the Operation of Multicomputing Interrupt OBs

A multicomputing interrupt is triggered by calling SFC35 "MP_ALM." During multicomputing, this brings about a synchronized OB60 start on all CPUs of the bus segment unless you have disabled OB60 (with SFC39 "DIS_IRT") or delayed it (with SFC41 "DIS_AIRT"). If you have not loaded OB60 on a CPU, the CPU returns to the last priority class before the interrupt and continues program execution there. In single processor operation and when using segmented racks, OB60 is only started on the CPU on which you called SFC35 "MP_ALM."

When your program calls SFC35 "MP_ALM," you supply a job ID. This ID is transferred to all CPUs. This allows you to react to a specific event. If you program OB60 differently on the various CPUs, this may result in different execution times for the OB. In this case, the CPUs return to the interrupted priority class at different times. If the next multicomputing interrupt is output by a CPU while another CPU is still busy executing the OB60 of the previous multicomputing interrupt, then OB60 is not started either on the requesting or on any other CPU belonging to the bus segment. This is illustrated in the following figure taking the example of two CPUs. You are informed of the outcome by the function value of the called SFC35.



Local Data for Multicomputing Interrupt OBs

The following table describes the temporary (TEMP) variables of the multicomputing interrupt OB. The variable names are the default names of OB60.

Variable	Data Type	Description
OB60_EV_CLASS	BYTE	Event class and IDs: B#16#11: Interrupt is active
OB60_STRT_INF	BYTE	B#16#61: Multicomputing interrupt triggered by own CPU B#16#62: Multicomputing interrupt triggered by another CPU
OB60_PRIORITY	BYTE	Assigned Priority class: default 25
OB60_OB_NUMBR	BYTE	OB number: 60
OB60_RESERVED_1	BYTE	Reserved
OB60_RESERVED_2	BYTE	Reserved
OB60_JOB	INT	Job ID: input variable JOB of SFC35 "MP_ALM"
OB60_RESERVED_3	INT	Reserved
OB60_RESERVED_4	INT	Reserved
OB60_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day at which the OB was called.

1.11 Synchronous Cycle Interrupt OB (OB61)

Description

Synchronous cycle interrupts give you the option of starting programs in synchronous cycle with the DP cycle. OB 61 serves as an interface OB to the synchronous cycle interrupt TSAL1. You can set the priority for OB 61 between 0 (OB deselected) and from 2 to 26.

Local Data for the Synchronous Cycle Interrupt OB

The following table describes the temporary (TEMP) variables of the synchronous cycle interrupt OB. The variable names are the default names of OB 61.

Variable	Data Type	Description
OB61_EV_CLASS	BYTE	Event class and IDs: B#16#11: Interrupt is active
OB61_STRT_INF	BYTE	B#16#64: Start request for the synchronous cycle interrupt OB
OB61_PRIORITY	BYTE	Assigned Priority class; default: 25
OB61_OB_NUMBR	BYTE	OB number: 61
OB61_RESERVED_1	BYTE	Reserved
OB61_RESERVED_2	BYTE	Reserved
OB61_GC_VIOL	BOOL	GC violation
OB61_FIRST	BOOL	First use after startup or stop status
OB61_MISSED_EXEC	BYTE	Number of failed starts of OB 61 since last execution of OB 61
OB61_DP_ID	BYTE	DP master system ID of the synchronous cycle DP string
OB61_RESERVED_3	BYTE	Reserved
OB61_RESERVED_4	BYTE	Reserved
OB61_RESERVED_5	WORD	Reserved
OB61_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day at which the OB was called.

1.12 I/O Redundancy Error OB (OB70)

Note

The I/O redundancy error OB (OB70) can only be used with H CPUs.

Description

The operating system of the H CPU calls OB70 when there is a loss of redundancy on PROFIBUS DP (for example, a bus failure for the active DP master or when an error occurs in the interface module of the DP slave) or when the active DP master of DP slaves with connected I/Os changes.

The CPU does not change to the STOP mode if a start event occurs and OB70 is not programmed. If OB70 is loaded and if the H system is in the redundant mode, OB70 is executed on both CPUs. The H system remains in the redundant mode.

Local Data of the I/O Redundancy OB

The following table contains the temporary (TEMP) variables of the I/O redundancy error OB. The variable names selected are the default names of OB70.

Variable	Type	Description
OB70_EV_CLASS	BYTE	Event class and IDs: • B#16#72: outgoing event • B#16#73: incoming event
OB70_FLT_ID	BYTE	Error code (possible values: B#16#A2, B#16#A3
OB70_PRIORITY	BYTE	Priority class; can be assigned via STEP 7 (hardware configuration)
OB70_OB_NUMBR	BYTE	OB number (70)
OB70_RESERVED_1	WORD	Reserved
OB70_INFO_1	WORD	Dependent on error code
OB70_INFO_2	WORD	Dependent on error code
OB70_INFO_3	WORD	Dependent on error code
OB70_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME at which the OB was called

The following table shows which event resulted in OB70 being started.

OB70_FLT_ID	Start Event of OB70
B#16#A2	Failure of a DP master or a DP master system respectively.
B#16#A3	Loss of redundancy / Return of redundancy of a DP slave.

The variables that depend on the error code have the following significance:

Error code	Bit	Meaning
• B#16#A2		
OB70_INFO_1:		Logical base address of the affected DP master
OB70_INFO_2:		Reserved
OB70_INFO_3:	0 to 7:	Reserved
	8 to 15:	DP master system ID of the affected DP master
• B#16#A3		
OB70_INFO_1:		Logical base address of the DP master
OB70_INFO_2:		Affected DP Slave:
	0 to 14:	Logical base address, if an S7 slave is used, or diagnostic address if a DP norm slave is used.
OB70_INFO_3	15:	I/O identification
		Affected DP slave:
	0 to 7:	Number of the DP station
	8 to 15:	DP master system ID

Note

If you are using a DPV1 capable CPU you can obtain additional information on the interrupt with the help of SFB54 "RALRM" which exceeds the start information of the OB. This also applies when you operate the DP Master in S7 compatible mode.

1.13 CPU Redundancy Error OB (OB72)

Note

The CPU redundancy error OB (OB72) exists only with H CPUs.

Description

The operating system of the H CPU calls OB72 when one of the following events occurs:

- Loss of CPU redundancy
- Reserve-master switchover
- Synchronization error
- Error in a SYNC module
- Updating aborted
- Comparison error (for example, RAM, PIQ)

OB72 is executed by all CPUs that are in the RUN or STARTUP mode following a suitable start event.

Local Data of the CPU Redundancy Error OB

The following table contains the temporary (TEMP) variables of the CPU redundancy error OB. The default names of OB72 have been used as the variable names.

Variable	Type	Description
OB72_EV_CLASS	BYTE	Event class and IDs: B#16#73, B#16#75, B#16#79, B#16#78
OB72_FLT_ID	BYTE	Error code (possible values: B#16#01, B#16#02, B#16#03, B#16#20, B#16#21, B#16#22, B#16#23, B#16#31, B#16#33, B#16#34, B#16#35, B#16#40, B#16#41, B#16#42, B#16#43, B#16#44, B#16#50, B#16#51, B#16#52, B#16#53, B#16#54, B#16#55, B#16#56, B#16#C1, B#16#C2)
OB72_PRIORITY	BYTE	Priority class; can be assigned via STEP 7 (hardware configuration)
OB72_OB_NUMBR	BYTE	OB number (72)

Variable	Type	Description
OB72_RESERVED_1	WORD	Only for error code B#16#03: • high byte: ID for the content of OB72_INFO_2 and OB72_INFO_3 - 0: OB72_INFO-2 and OB72_INFO_3 are of no significance - B#16#C4: Transition to redundant mode after troubleshooting mode was carried out with standby-master switch-over (if OB72_INFO_3=W#16#0001) or without standby-master switch-over (if OB72_INFO_3=W#16#0002). OB72_INFO_2 is reserved. - B#16#CD: OB72_INFO_2 and OB72_INFO_3 contain the actual lock time for priority classes > 15 - Low byte: reserved
OB72_INFO_1	WORD	Only for error code B#16#C2: - High byte: ID for exceeded monitoring time: - 1: Scan cycle time increase - 2: I/O dead time - 3: Communication time delay - Low byte: current update attempt
OB72_INFO_2	WORD	Only for error code B#16#03 and OB72_RESERVED_1=B#16#CD: high word of the actual lock time for priority classes > 15 in ms
OB72_INFO_3	WORD	Only for error code B#16#03: • OB72_RESERVED_1=B#16#C4: - W#16#0001: Transition to redundant operation after troubleshooting mode was carried out **with** standby-master switch-over - W#16#0002: Transition to redundant operation after troubleshooting mode was carried out **without** standby-master switch-over OB72_RESERVED_1=B#16#CD: high word of the actual lock time for priority classes > 15 in ms
OB82_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME at which the OB was called

The following table shows which event caused OB72 to be started.

OB72_FLT_ID	Start Event of OB72
B#16#01	Loss of redundancy (1 of 2) due to a CPU failure
B#16#02	Loss of redundancy (1 of 2) due to STOP on the reserve triggered by user
B#16#03	H system (1 of 2) changed to redundant mode
B#16#20	Error in RAM comparison
B#16#21	Error comparing process image output value

OB72_FLT_ID	Start Event of OB72
B#16#22	Error comparing memory bits, timers, or counters
B#16#23	Different operating system data recognized
B#16#31	Standby-master switchover due to master failure
B#16#33	Standby-master switchover due to operator intervention
B#16#34	Standby-master switchover due to sync module connection problem
B#16#35	Standby-master switchover triggered by 90 "H_CTRL"
B#16#40	Synchronization error in user program due to elapsed wait time
B#16#41	Synchronization error in user program due to waiting at different synchronization points
B#16#42	Synchronization error in operating system due to waiting at different synchronization points
B#16#43	Synchronization error in operating system due to elapsed wait time
B#16#44	Synchronization error in operating system due to wrong data
B#16#50	No SYNC module
B#16#51	Modification at SYNC module without Power On
B#16#52	SYNC module removed/inserted
B#16#53	Modification at SYNC module without reset
B#16#54	SYNC module: rack number assigned twice
B#16#55	SYNC module error/eliminated
B#16#56	Illegal rack number set on the SYNC module
B#16#C1	Updating aborted
B#16#C2	Abort of update attempt because a monitoring time was exceeded during the n-th attempt (1 <= n <= maximum possible number of update attempts after an abort due to the monitoring time being exceeded.)

1.14 Communication Redundancy Error OB (OB73)

Note

The communications redundancy error OB (OB73is only available in firmware version V2.0.x for the CPU 417-4H.

Description

The operating system of the H CPU calls OB73 when the first loss of redundancy occurs in a fault-tolerant S7 connection (Fault-tolerant S7 connections only exist for S7 communication. For more information, see "S7-400 H Programmable Controller, Fault-Tolerant Systems."). If a loss of redundancy occurs for additional fault-tolerant S7 connections, there are no more OB73 starts.

Another OB73 start will not occur until you have restored redundancy for all S7 connections that were fault tolerant.

The CPU does not change to the STOP mode if a start event occurs and the OB73 is not programmed.

Local Data of the CPU Redundancy Error OB

The following table contains the temporary (TEMP) variables of the communication redundancy error OB. The default names of OB73 have been used as the variable names.

Variable	Type	Description
OB73_EV_CLASS	BYTE	Event class and IDs: B#16#73, B#16#72
OB73_FLT_ID	BYTE	Error code (possible values: B#16#E0)
OB73_PRIORITY	BYTE	Assigned priority class: default 25
OB73_OB_NUMBR	BYTE	OB number (73)
OB73_RESERVED_1	WORD	Reserved
OB73_INFO_1	WORD	(irrelevant to the user)
OB73_INFO_2	WORD	(irrelevant to the user)
OB73_INFO_3	WORD	(irrelevant to the user)
OB73_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME at which the OB was called

The following table shows which event caused OB73 to be started.

OB73_FLT_ID	Start Event of OB 73
B#16#E0	Loss of redundancy in communication/problem eliminated

1.15 Time Error Organization Block (OB80)

Description

The operating system of the S7-300 CPU calls OB80 whenever an error occurs while executing an OB. Such errors include: exceeding the cycle time, an acknowledgement error when executing an OB, moving the time forward so that the start time for the OB is skipped. If, for example, a start event for a cyclic OB occurs while the same OB is still being executed following a previous call, the operating system calls OB80.

If OB80 has not been programmed, the CPU changes to the STOP mode.

You can disable or delay and re-enable the time error OB using SFCs 39 to 42.

Note

If OB80 is called twice during the same scan cycle due to the scan time being exceeded, the CPU changes to the STOP mode. You can prevent this by calling SFC43 "RE_TRIGR" at a suitable point in the program.

Local Data for the Time Error OB

The following table describes the temporary (TEMP) variables for the time error OB. The variable names are the default names of OB80.

Variable	Type	Description
OB80_EV_CLASS	BYTE	Event class and identifiers: B#16#35
OB80_FLT_ID	BYTE	Error code: (possible values: B#16#01, B#16#02, B#16#05, B#16#06 or B#16#07 B#16#08)
OB80_PRIORITY	BYTE	Priority class; can be assigned via STEP 7 (hardware configuration)
OB80_OB_NUMBR	BYTE	OB number (80)
OB80_RESERVED_1	BYTE	Reserved
OB80_RESERVED_2	BYTE	Reserved
OB80_ERROR_INFO	WORD	Error information: depending on error code
OB80_ERR_EV_CLASS	BYTE	Event class for the start event that caused the error
OB80_ERR_EV_NUM	BYTE	Event number for the start event that caused the error
OB80_OB_PRIORITY	BYTE	Error information: depending on error code
OB80_OB_NUM	BYTE	Error information: depending on error code
OB80_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

The variables dependent on the error code have the following meaning:

Error code	Bit	Meaning
• B#16#01		Cycle time exceeded.
OB80_ERROR_INFO:		Run time of last scan cycle (ms).
OB80_ERR_EV_CLASS:		Class of the event that triggered the interrupt.
OB80_ERR_EV_NUM:		Number of the event that triggered the interrupt.
OB80_OB_PRIORITY:		Priority class of the OB which was being executed when the error occurred.
OB80_OB_NUM		Number of the OB which was being executed when the error occurred.
• B#16#02		The called OB is still being executed.
OB80_ERROR_INFO:		The respective temporary variable of the called block which is determined by • OB80_ERR_EV_CLASS and • OB80_ERR_EV_NUM.
OB80_ERR_EV_CLASS:		• Class of the event that triggered the interrupt.
OB80_ERR_EV_NUM:		• Number of the event that triggered the interrupt.
OB80_OB_PRIORITY		• Priority class of the OB causing the error (for example: "7“ for OB30/Priority class 7 which should have been started, but could not be started).
OB80_OB_NUM:		• Number of the OB causing the error (for example: "30“ for OB30 which should have been started, but could not be started).
• B#16#05 and		Elapsed time-of-day interrupt due to moving the clock forward.
• B#16#06		Elapsed time-of-day interrupt on return to RUN after HOLD.
OB80_ERROR_INFO:	Bit 0 set:	The start time for time-of-day interrupt 0 is in the past.
	Bit 7 set:	The start time for time-of-day interrupt 7 is in the past.
	Bit 8 to 15:	Not used
OB80_ERR_EV_CLASS:		Not used
OB80_ERR_EV_NUM:		Not used
OB80_OB_PRIORITY:		Not used
OB80_OB_NUM:		Not used
• B#16#07 Meaning of the parameters see error code B#16#02.		Overflow of OB request buffer for the current priority class (Each OB start request for a priority class will be entered in the corresponding OB request buffer; after completion of the OB the entry will be deleted. If there are more OB start requests for a priority class than the maximum permitted number of entries in the corresponding OB request buffer, OB80 will be called with error code B#16#07).
• B#16#08 Meaning of the parameters see error code B#16#02.		Synchronous-cycle interrupt time error

1.16 Power Supply Error Organization Block (OB81)

Description

The operating system of the S7-300 CPU calls OB81 whenever an event occurs that is triggered by an error or fault related to the power supply (only on an S7-400) or the back-up battery (when entering and when outgoing event).

In S7-400, OB81 is only called in the event of a battery fault if the battery test function has been activated with the BATT.INDIC switch.

The CPU does not change to the STOP mode if OB81 is not programmed.

You can disable or delay and re-enable the power supply error OB using SFCs 39 to 42.

Local Data for the Power Supply Error OB

The following table describes the temporary (TEMP) variables for the power supply error OB. The variable names are the default names of OB81.

Variable	Type	Description
OB81_EV_CLASS	BYTE	Event class and identifiers: B#16#38: outgoing event B#16#39: incoming event
OB81_FLT_ID	BYTE	Error code: (possible values) B#16#21, B#16#22, B#16#23, B#16#25, B#16#26, B#16#27, B#16#31, B#16#32, B#16#33)
OB81_PRIORITY	BYTE	Priority class; can be assigned via STEP 7 (hardware configuration) For example, possible values for the RUN mode: 2-26
OB81_OB_NUMBR	BYTE	OB number (81)
OB81_RESERVED_1	BYTE	Reserved
OB81_RESERVED_2	BYTE	Reserved
OB81_MDL_ADDR	INT	• Bits 0 to 2: Rack no. • Bit 3: 0=standby CPU, 1=master CPU • Bits 4 to 7: 1111
OB81_RESERVED_3	BYTE	Relevant only for error codes B#16#31, B#16#32 and B#16#33
OB81_RESERVED_4	BYTE	
OB81_RESERVED_5	BYTE	
OB81_RESERVED_6	BYTE	
OB81_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

The variables OB81_RESERVED_i, $3 \leq i \leq 6$ indicate the expansion racks on which the battery backup (error code B#16#31), the back-up voltage (error code B#16#32) or the 24-V power supply (error code B#16#33) has failed or returned.

The following table shows what bit is assigned to which expansion rack in the variables OB81_RESERVED_i, $3 \leq i \leq 6$.

	OB81_RESERVED_6	**OB81_RESERVED_5**	**OB81_RESERVED_4**	**OB81_RESERVED_3**
Bit 0	Reserved	8th expansion rack	16th expansion rack	Reserved
Bit 1	1st expansion rack	9th expansion rack	17th expansion rack	Reserved
Bit 2	2nd expansion rack	10th expansion rack	18th expansion rack	Reserved
Bit 3	3rd expansion rack	11th expansion rack	19th expansion rack	Reserved
Bit 4	4th expansion rack	12th expansion rack	20th expansion rack	Reserved
Bit 5	5th expansion rack	13th expansion rack	21st expansion rack	Reserved
Bit 6	6th expansion rack	14th expansion rack	Reserved	Reserved
Bit 7	7th expansion rack	15th expansion rack	Reserved	Reserved

The bits in the variables OB81_RESERVED_i have the following meaning (for the expansion rack concerned):

When the event occurs, the expansion racks are marked (the corresponding bits are set) on which at least one battery or back-up voltage or the 24 V power supply has failed. Expansion racks on which at least one battery or back-up voltage or the 24 V power supply failed earlier are no longer indicated.
When the event is eliminated and the backup is restored on at least one expansion rack, this is signaled (the corresponding bits are set).

The variable OB81_FLT_ID has the following meaning:

OB81_FLT_ID	**Meaning**
B#16#21:	At least one back-up battery of the central rack is exhausted/problem eliminated (BATTF) Note: This event occurs only if one of the two batteries fails (if there are redundant back-up batteries). If the second battery should also happen to fail, the event will not occur again.
B#16#22:	Back-up voltage in the central rack failed/problem eliminated (BAF)
B#16#23:	Failure of the 24 V power supply in the central rack/problem eliminated.
B#16#25:	At least one back-up battery in at least one redundant central rack is exhausted/problem eliminated (BATTF)
B#16#26:	Back-up voltage in at least one redundant central rack failed/problem eliminated (BAF)
B#16#27:	Failure of the 24 V supply in at least one redundant central rack
B#16#31:	At least one back-up battery of at least one expansion rack is exhausted/problem eliminated (BATTF).
B#16#32:	Back-up voltage in at least one expansion rack failed/problem eliminated (BAF)
B#16#33:	Failure of the 24 V power supply in at least one expansion rack/problem eliminated.

1.17 Diagnostic Interrupt Organization Block (OB82)

Description

If a module with diagnostic capability for which you have enabled the diagnostic interrupt detects an error, it outputs a request for a diagnostic interrupt to the CPU (when entering and outgoing event). The operating system then calls OB82.

The local variables of OB82 contain the logical base address as well as four bytes of diagnostic data of the defective module (see the following table).

If OB82 has not been programmed, the CPU changes to the STOP mode.

You can disable or delay and re-enable the diagnostic interrupt OB using SFCs 39 to 42.

Local Data for Diagnostic Interrupt OB

The following table describes the temporary (TEMP) variables for the diagnostic interrupt OB. The variable names are the default names of OB82.

Variable	Type	Description
OB82_EV_CLASS	BYTE	Event class and identifiers: • B#16#38: outgoing event • B#16#39: incoming event
OB82_FLT_ID	BYTE	Error code (B#16#42)
OB82_PRIORITY	BYTE	• Priority class; can be assigned via STEP 7 (hardware configuration)
OB82_OB_NUMBR	BYTE	OB number (82)
OB82_RESERVED_1	BYTE	Reserved
OB82_IO_FLAG	BYTE	• Input module: B#16#54 • Output module: B#16#55
OB82_MDL_ADDR	WORD	Logical base address of the module where the fault occurred
OB82_MDL_DEFECT	BOOL	Module is defective
OB82_INT_FAULT	BOOL	Internal fault
OB82_EXT_FAULT	BOOL	External fault
OB82_PNT_INFO	BOOL	Channel fault
OB82_EXT_VOLTAGE	BOOL	External voltage failed
OB82_FLD_CONNCTR	BOOL	Front panel connector not plugged in
OB82_NO_CONFIG	BOOL	Module is not configured
OB82_CONFIG_ERR	BOOL	Incorrect parameters on module
OB82_MDL_TYPE	BYTE	• Bit 0 to 3: Module class • Bit 4: Channel information exists • Bit 5: User information exists • Bit 6: Diagnostic interrupt from substitute • Bit 7: Reserve

Variable	Type	Description
OB82_SUB_MDL_ERR	BOOL	Submodule is missing or has an error
OB82_COMM_FAULT	BOOL	Communication problem
OB82_MDL_STOP	BOOL	Operating mode (0: RUN, 1: STOP)
OB82_WTCH_DOG_FLT	BOOL	Watchdog timer responded
OB82_INT_PS_FLT	BOOL	Internal power supply failed
OB82_PRIM_BATT_FLT	BOOL	Battery exhausted
OB82_BCKUP_BATT_FLT	BOOL	Entire backup failed
OB82_RESERVED_2	BOOL	Reserved
OB82_RACK_FLT	BOOL	Expansion rack failure
OB82_PROC_FLT	BOOL	Processor failure
OB82_EPROM_FLT	BOOL	EPROM fault
OB82_RAM_FLT	BOOL	RAM fault
OB82_ADU_FLT	BOOL	ADC/DAC error
OB82_FUSE_FLT	BOOL	Fuse tripped
OB82_HW_INTR_FLT	BOOL	Hardware interrupt lost
OB82_RESERVED_3	BOOL	Reserved
OB82_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

Note

If you are using a DPV1 capable CPU you can obtain additional information on the interrupt with the help of SFB54 "RALRM" which exceeds the start information of the OB. This also applies when you operate the DP Master in S7 compatible mode.

1.18 Insert / Remove Module Interrupt Organization Block (OB83)

Description

The insertion and removal of modules is monitored within the system at intervals of one second. For the CPU to recognize that a module has been removed and inserted you must wait a minimum of two seconds between removing and inserting.

Each time a configured module is removed or inserted during the RUN, STOP, and STARTUP modes, an insert/remove interrupt is generated (power supply modules, CPUs, adapter modules and IMs must not be removed in these modes). This interrupt causes an entry in the diagnostic buffer and in the system status list for the CPU involved. The insert/remove OB is also started if the CPU is in the RUN mode. If this OB has not been programmed, the CPU changes to the STOP mode.

You can disable or delay and re-enable the insert/remove OB using SFCs 39 to 42.

Understanding the Operation of OB83

If you remove a configured module in the RUN mode, OB83 is started. Since the existence of modules is only monitored at intervals of one second, an access error may be detected first if the module is accessed directly or when the process image is updated.

If you insert a module in a configured slot in the RUN mode, the operating system checks whether the type of the module inserted corresponds to the recorded configuration. OB83 is then started and parameters are assigned if the module types match.

Local Data for OB83

The following table describes the temporary (TEMP) variables for the insert/remove module interrupt OB. The variable names are the default names of OB83.

Variable	Type	Description
OB83_EV_CLASS	BYTE	Event class and identifiers: • B#16#38: module inserted • B#16#39: module removed or not responding
OB83_FLT_ID	BYTE	Error code: (possible values B#16#61, B#16#63 or B#16#64)
OB83_PRIORITY	BYTE	• Priority class; can be assigned via STEP 7 (hardware configuration)
OB83_OB_NUMBR	BYTE	OB number (83)
OB83_RESERVED_1	BYTE	Identification of block module or interface module
OB83_MDL_TD	BYTE	Range: • B#16#54: Peripheral input (PI) • B#16#55: Peripheral output (PQ)
OB83_MDL_ADDR	WORD	Logical base address of the module affected

Variable	Type	Description
OB83_RACK_NUM	WORD	• If OB83_RESERVED_1 =B#16#A0: number of interface module • If OB83_RESERVED_1 = B#16#C4: rack number or number of DP station (low byte) and DP master system ID (high byte)
OB83_MDL_TYPE	WORD	Module type of the module affected: • W#16#X5XX: analog module • W#16#X8XX: function module • W#16#XCXX: CP • W#16#XFXX: digital module X : Value irrelevant to the user
OB83_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

The variable OB83_MDL_TYPE dependent on the error code OB83_FLT_ID has the following meaning:

Error code	Meaning
• B#16#61:	• Module inserted. Module type OK (for event class B#16#38) • Module removed or not responding (for event class B#16#39)
OB83_MDL_TYPE:	Actual module type
• B#16#63	Module inserted but incorrect module type
• OB83_MDL_TYPE:	Actual module type
• B#16#64:	Module inserted but problem (module ID cannot be read)
OB83_MDL_TYPE:	Configured module type
• B#16#65:	Module inserted but error in module parameter assignment
OB83_MDL_TYPE:	Actual module type
• W#16#3866	Module responds again, loaded voltage error corrected
• W#16#3966	Module not responding, loaded voltage error
• W#16#3884	Interface module inserted
• W#16#3984	Interface module removed

Note

If you are using a DPV1 capable CPU you can obtain additional information on the interrupt with the help of SFB54 "RALRM" which exceeds the start information of the OB. This also applies when you operate the DP Master in S7 compatible mode.

1.19 CPU Hardware Fault Organization Block (OB84)

Description

The CPU's operating system calls OB84 whenever an error is detected on the interface to the MPI network, to the internal communication bus (K bus), or to the interface module for the distributed I/Os.

If OB84 has not been programmed, the CPU changes to the STOP mode when this type of error is detected.

You can disable or delay and re-enable the CPU hardware error OB using SFCs 39 to 42.

Behavior with newer S7-400 CPUs

Differing to previous behavior, OB84 is not called anymore by S7-400-CPUs with the following MLFBs : 6ES7 412-1XF03-0AB0, 6ES7 412-2XG00-0AB0, 6ES7 414-2XG03-0AB0, 6ES7 414-3XJ00-0AB0, 6ES7 414-4HJ00-0AB0, 6ES7 416-2XK02-0AB0, 6ES7 416-3XL00-0AB0, 6ES7 417-4XL00-0AB0, 6ES7 417-4HL01-0AB0. Only an entry in the diagnostic buffer is generated.

You must observe this fact if you did not program OB 84 in order to put the CPU in STOP with one of the above mentioned errors.

Local Data for the Hardware Fault OB

The following table includes the temporary (TEMP) variables of the CPU hardware fault. The variable names are the default names of OB84.

Variable	Type	Description
OB84_EV_CLASS	BYTE	Event class and identifiers: • B#16#38: outgoing event • B#16#39: incoming event
OB84_FLT_ID	BYTE	Error code (B#16#81)
OB84_PRIORITY	BYTE	Priority class; can be assigned via STEP 7 (hardware configuration)
OB84_OB_NUMBR	BYTE	OB number (84)
OB84_RESERVED_1	BYTE	Reserved
OB84_RESERVED_2	BYTE	Reserved
OB84_RESERVED_3	WORD	Reserved
OB84_RESERVED_4	DWORD	Reserved
OB84_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

Note

Before loading OB 84 in the CPU check the contents of the diagnostic buffer. With the entries W#16#6881 or W#16#6981 (interface error) the load process is stopped. You can only start a new load process after resetting the CPU.

1.20 Priority Class Error Organization Block (OB85)

Description

The operating system of the CPU calls OB85 whenever one of the following events occurs:

- Start event for an OB that has not been loaded (except OB81).
- Error when the operating system accesses a module.
- I/O access error during update of the process image by the system (if the OB85 call was not suppressed due to the configuration).

Note

If OB85 has not been programmed, the CPU changes to the STOP mode when one of these errors is detected.

You can disable or delay and re-enable the priority class error OB using SFCs 39 to 42.

Local Data for the Priority Class Error OB

The following table describes the temporary (TEMP) variables for the priority class error OB. The variable names are the default names of OB85.

Variable	Type	Description
OB85_EV_CLASS	BYTE	Event class and identifiers: B#16#35 B#16#38 (only with error codes B#16#B3 and B#16#B4) B#16#39 (only with error codes B#16#B1, B#16#B2, B#16#B3 and B#16#B4)
OB85_FLT_ID	BYTE	Error code (possible values: B#16#A1, B#16#A2, B#16#A3, B#16#B1, B#16#B2, B#16#B3, B#16#B4)
OB85_PRIORITY	BYTE	Priority class; can be assigned via STEP 7 (hardware configuration)
OB85_OB_NUMBR	BYTE	OB number (85)
OB85_RESERVED_1	BYTE	Reserved
OB85_RESERVED_2	BYTE	Reserved
OB85_RESERVED_3	INT	Reserved
OB85_ERR_EV_CLASS	BYTE	Class of the event that caused the error
OB85_ERR_EV_NUM	BYTE	Number of the event that caused the error
OB85_OB_PRIOR	BYTE	Priority class of the OB that was active when the error occurred
OB85_OB_NUM	BYTE	Number of the OB that was active when the error occurred
OB85_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

If you want to program OB85 dependent on the possible error codes, we recommend that you organize the local variables as follows:

Variable	Type
OB85_EV_CLASS	BYTE
OB85_FLT_ID	BYTE
OB85_PRIORITY	BYTE
OB85_OB_NUMBR	BYTE
OB85_DKZ23	BYTE
OB85_RESERVED_2	BYTE
OB85_Z1	WORD
OB85_Z23	DWORD
OB85_DATE_TIME	DATE_AND_TIME

The variables modified compared with the default have the following meaning, dependent on the error code:

Error code	Byte/Word	Meaning
B#16#A1 and		As a result of your configuration created with STEP 7, your program or the operating system creates a start event for an OB that is not loaded on the CPU.
B#16#A2		As a result of your configuration created with STEP 7, your program or the operating system creates a start event for an OB that is not loaded on the CPU.
OB85_Z1:		The respective local variable of the called OB which is determined by OB85_Z23.
OB85_Z23:	high word:	Class and number of the event causing the OB call
	low word:	Program level and OB active at the time of error
B#16#A3		Error when the operating system accesses a module
		• Error ID of the operating system
OB85_Z1:	high byte:	• 1: integrated function
		• 2: IEC timer
	low byte:	• 0: no error resolution
		• 1: block not loaded
		• 2: area length error
		• 3: write-protect error
OB85_Z23	high word	• Block number
	low word:	• Relative address of the MC7 command causing the error. The block type must be taken from OB85_DKZ23 (B#16#88: OB, B#16#8C: FC, B#16#8E: FB, B#16#8A: DB).

Error code	Byte/Word	Meaning
B#16#B1 and B#16#B2:		I/O access error when updating the process image of the inputs I/O access error when transferring the process image of the outputs to the output modules
You obtain the error codes B#16#B1 and B#16#B2 if you have configured the repeated OB85 call of I/O access errors for the system process image table update.		
B#16#B3: B#16#B4:		I/O access error when updating the process image of the inputs, incoming/outgoing event I/O access error when transferring the output process image to the output module, incoming/outgoing event
You obtain the error codes B#16#B3 and B#16#B4 if you configured the OB85 call of I/O access errors entering and outgoing event for process image table updating by the system. After a cold or warm restart, all access to non-existing inputs and outputs will be reported as I/O access errors during the next process image table updating.		
OB85_DKZ23:		ID of the type of process image transfer during which the I/O access error has occurred • B#16#10: Byte access • B#16#20: Word access • B#16#30: DWord access • B#16#57: Transmitting a configured consistency range
OB85_Z1:		Reserved for internal use by the CPU: logical base address of the module If OB85_RESERVED_2 has the value B#16#76, OB85_Z1 receives the return value of the affected SFC (SFC 14, 15, 26 or 27).
OB85_Z23:	Byte 0:	Part process image no.
	Byte 1:	• Irrelevant, if OB85_DKZ23=B#16#10, 20 or 30 • Length of the consistency range in bytes, if OB85_DKZ23=B#16#57
	Bytes 2 and 3	• The I/O address causing the PAE, if OB85_DKZ23=B#16#10, 20 or 30 • Logical start address of the consistency range, if OB85_DKZ23=B#16#57

1.21 Rack Failure Organization Block (OB86)

Description

The operating system of the CPU calls OB86 whenever the failure of an expansion rack, a DP master system, or a station is detected in the distributed I/Os (both when entering and outgoing event).

If OB86 has not been programmed, the CPU changes to the STOP mode when this type of error is detected.

You can disable or delay and re-enable OB86 using SFCs 39 to 42.

Local Data for the Rack Failure OB

The following table describes the temporary (TEMP) variables for the rack failure OB. The variable names are the default names of OB86.

Variable	Type	Description
OB86_EV_CLASS	BYTE	Event class and identifiers: • B#16#38: outgoing event • B#16#39: incoming event
OB86_FLT_ID	BYTE	Error code: (possible values B#16#C1, B#16#C2, B#16#C3, B#16#C4, B#16#C5, B#16#C6, B#16#C7, B#16#C8)
OB86_PRIORITY	BYTE	• Priority class; can be assigned via STEP 7 (hardware configuration)
OB86_OB_NUMBR	BYTE	OB number (86)
OB86_RESERVED_1	BYTE	Reserved
OB86_RESERVED_2	BYTE	Reserved
OB86_MDL_ADDR	WORD	Depends on the error code
OB86_RACKS_FLTD	Array [0 ..31] of BOOL	Depends on the error code
OB86_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

If you want to program OB86 dependent on the possible error codes, we recommend that you organize the local variables as follows:

Variable	Type
OB86_EV_CLASS	BYTE
OB86_FLT_ID	BYTE
OB86_PRIORITY	BYTE
OB86_OB_NUMBR	BYTE
OB86_RESERVED_1	BYTE
OB86_RESERVED_2	BYTE
OB86_MDL_ADDR	WORD
OB86_Z23	DWORD

Variable	Type
OB86_DATE_TIME	DATE_AND_TIME

The variables whose contents are dependent on the error code have the following meaning:

Error code	Meaning
B#16#39C1:	Expansion rack failure
OB86_MDL_ADDR:	Logical base address of the IM
OB86_Z23:	Contains one bit for each possible expansion rack: • Bit 0: always 0 • Bit 1: 1st expansion rack : • Bit 21: 21st expansion rack • Bit 22 to 29: always 0 • Bit 30: Failure of at least one expansion rack in the SIMATIC S5 area • Bit 31: always 0
B#16#38C1:	Expansion rack operational again

Meaning: when the event occurs, the expansion racks that caused OB86 to be called are reported as having failed (the bits assigned to them are set). Expansion racks that had already failed earlier are no longer indicated. When the failure is eliminated, the expansion racks that are active again are reported in the error code (the bits assigned to them are set).

Error code	Meaning
B#16#C2:	Expansion rack returned with discrepancy between expected and actual configuration.
OB86_MDL_ADDR:	Logical base address of the IM
OB86_Z23:	• Contains one bit for every possible expansion rack, see error code B#16#C1. • Meaning of a bit when set (for the expansion rack concerned): - Modules with an incorrect type ID exist - Configured modules missing - At least one module is defective.
B#16#C3:	Distributed I/Os: Failure of the master system. (Only incoming event causes the start of OB86 with error code B#16#C3. Leaving the state starts OB86 with the error code B#16#C4 and event class B#16#38. The return of every DP slave station starts OB86.)
OB86_MDL_ADDR:	Logical base address of the DP master.
OB86_Z23:	DP master system ID • Bit 0 to 7: reserved • Bit 8 to 15: DP master system ID • Bit 16 to 31: reserved
B#16#C4:	Failure of a DP station.
B#16#C5:	DP station disturbed

Error code	Meaning
OB86_MDL_ADDR: OB86_Z23:	Logical base address of the DP master. Address of DP slave affected: • Bits 0 to 7: number of DP station • Bits 8 to 15: DP master system ID • Bits 16 to 30: logical base address of an S7 slave or diagnostic address of a standard DP slave • Bit 31: I/O identifier
B#16#C6: OB86_MDL_ADDR: OB86_Z23:	Expansion rack operational again but error in module parameter assignment Logical base address of IM Contains a bit for every possible expansion rack: • Bit 0: always 0 • Bit 1: 1st expansion rack : • Bit 21: 21st expansion rack • Bit 22 to 30: reserved • Bit 31: always 0 Meaning when bit set (in expansion rack concerned): • Modules with incorrect type identifiers exist • Modules with missing or incorrect parameters exist
B#16#C7: OB86_MDL_ADDR: OB86_Z23:	Return of a DP station, but error in module parameter assignment Logical base address of the DP master Address of the DP slave affected: • Bits 0 to 7: No. of the DP station • Bits 8 to 15: DP master system ID • Bits 16 to 30: Logical base address of the DP slave • Bit 31: I/O identifier
B#16#C8: OB86_MDL_ADDR: OB86_Z23:	Return of a DP station, however discrepancy in configured and actual configuration Logical base address of the DP master Address of the DP slave affected: • Bits 0 to 7: No. of the DP station • Bits 8 to 15: DP master system ID • Bits 16 to 30: Logical base address of the DP slave Bit 31: I/O identifier

Note

If you are using a DPV1 capable CPU you can obtain additional information on the interrupt with the help of SFB54 "RALRM" which exceeds the start information of the OB. This also applies when you operate the DP Master in S7 compatible mode.

1.22 Communication Error Organization Block (OB87)

Description

The operating system of the CPU calls OB87 whenever an event occurs that was caused by a communication error.

The CPU does not change to the STOP mode if OB87 has not been programmed.

You can disable or delay and re-enable the communication error OB using SFCs 39 to 42.

Local Data for OB87

The following table describes the temporary (TEMP) variables for the communication error OB. The variable names are the default names of OB87.

Variable	Type	Description
OB87_EV_CLASS	BYTE	Event class and identifiers: B#16#35
OB87_FLT_ID	BYTE	Error code: (possible values: B#16#D2, B#16#D3, B#16#D4, B#16#D5, B#16#E1, B#16#E2, B#16#E3, B#16#E4, B#16#E5, B#16#E6)
OB87_PRIORITY	BYTE	Priority class; can be assigned via STEP 7 (hardware configuration)
OB87_OB_NUMBR	BYTE	OB number (87)
OB87_RESERVED_1	BYTE	Reserved
OB87_RESERVED_2	BYTE	Reserved
OB87_RESERVED_3	WORD	Depends on the error code
OB87_RESERVED_4	DWORD	Depends on the error code
OB87_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

The variables dependent on the error code have the following meaning:

Error code	Byte/Word	Meaning
B#16#D2:		Transmission of diagnostic entries currently not possible.
B#16#D3		Synchronization messages cannot be transmitted (master).
B#16#D4:		Illegal time-of-day jump due to clock synchronization.
B#16#D5		Error when receiving synchronization time (slave).
OB87_RESERVED_3:		Contains no further information.
OB87_RESERVED_4:		Contains no further information.
B#16#E1:		Incorrect frame ID during global data communication.
B#16#E3:		Frame length error during global data communication.

Error code	Byte/Word	Meaning
B#16#E4:		Illegal GD packet number received.
OB87_RESERVED_3:		interface ID (0: K bus, 1: MPI)
OB87_RESERVED_4:	high byte:	GD circuit number
	low byte:	Contains no further information.
B#16#E2:		GD packet status cannot be entered in the DB
OB87_RESERVED_3:		DB number
OB87_RESERVED_4:	high word:	Contains no further information.
	low word:	• GD circle number (high byte), • GD packet number (low byte)
B#16#E5:		Access error to DB during data exchange via communication function blocks
OB87_RESERVED_3:		Reserved for internal use by CPU.
OB87_RESERVED_4:	high word:	Number of block containing the MC7 command that caused the error.
	low word:	Relative address of the MC7 command that has caused the error.

The block type may be read from OB_87_RESERVED_1 (B#16#88: OB, B#16#8A: DB, B#16#8C: FC, B#16#8E: FB).

Error code	Meaning
B#16#E6:	GD group status cannot be entered in DB.
OB87_RESERVED_3:	DB number.
OB87_RESERVED_4:	Contains no further information.

1.23 Background Organization Block (OB90)

Description

With STEP 7, you can monitor a maximum scan cycle time and can guarantee a minimum scan cycle time. If the execution time of OB1 including all the nested interrupts and system activities is less than the minimum scan cycle time that you have specified, the operating system reacts as follows:

- It calls the background OB (providing it exists on the CPU).
- It delays the next OB1 start (if OB90 does not exist on the CPU).

Understanding the Operation of OB90

OB90 has the lowest priority of all OBs. It is interrupted by any system activity and any interrupt (even by OB1 after the minimum cycle time has elapsed) and is only resumed if the selected minimum scan cycle time has not yet been reached. The one exception to this is the execution of SFCs and SFBs that are started in OB90. These are executed with the priority of OB1 and are therefore not interrupted by OB1. There is no time monitoring of OB90.

The user program in OB90 is processed starting with the first instruction in the following situations:

- Following a warm, cold, or hot restart
- After deleting a block being executed in OB90 (with STEP 7)
- After loading OB90 on the CPU in the RUN mode
- After terminating the background cycle

Note

With configurations in which there is no great difference between the minimum scan cycle time and the cycle monitoring time, SFC and SFB calls in the background OB can lead to the cycle time being exceeded unexpectedly.

Local Data for OB90

The following table describes the temporary (TEMP) variables of OB90. The variable names are the default names of OB90.

Variable	Data Type	Description
OB90_EV_CLASS	BYTE	Event class and identifiers: B#16#11: active
OB90_STRT_INF	BYTE	• B#16#91: warm restart/cold restart/hot restart • B#16#92: block deleted • B#16#93: downloading OB90 to the CPU in the RUN mode • B#16#95: termination of the background cycle
OB90_PRIORITY	BYTE	Priority class: 29 (corresponds to priority 0.29)
OB90_OB_NUMBR	BYTE	OB number (90)
OB90_RESERVED_1	BYTE	Reserved
OB90_RESERVED_2	BYTE	Reserved
OB90_RESERVED_3	INT	Reserved
OB90_RESERVED_4	INT	Reserved
OB90_RESERVED_5	INT	Reserved
OB90_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day at which the OB was called

1.24 Startup Organization Blocks (OB100, OB101 and OB102)

Types of Startup

A distinction is made between the following types of startup

- Hot restart (not with the S7-300 and the S7-400H)
- Warm restart
- Cold restart

In the following table, you can see which OB is called by the operating system during startup.

Type of Startup	Corresponding OB
Hot restart	OB101
Warm restart	OB100
Cold restart	OB102

For more detailed information on the types of startup, refer to the manuals **"Programming and Hardware Configuration with STEP 7"** and **"S7-400H Programmable Controller."**

Start Events for Startup

The CPU executes a startup as follows:

- After POWER ON
- Whenever you switch the mode selector from STOP to RUN-P
- After a request using a communication function (menu command from the programming device or by calling the communication function blocks 19 "START" or 21 "RESUME" on a different CPU).
- Synchronization in multicomputing
- In an H system after link-up (only on the standby CPU)

Depending on the start event, the particular CPU, and its parameters, the appropriate startup OB (OB100, OB101, or OB102) is called. With suitable programming, you can make certain settings for your cyclic program (exception: in an H system, when the standby CPU is linked-up, there is a startup on the standby CPU but no startup OB is called).

Local Data for Startup OBs

The following table describes the temporary (TEMP) variables for a startup OB. The variable names are the default names of OB100.

Variable	Type	Description
OB10x_EV_CLASS	BYTE	Event class and identifiers: B#16#13: active
OB10x_STRTUP	BYTE	Startup request: • B#16#81: Manual warm restart • B#16#82: Automatic warm restart • B#16#83: Request for manual hot restart • B#16#84: Request for automatic hot restart • B#16#85: Request for manual cold restart • B#16#86: Request for automatic cold restart • B#16#87: Master: Request for manual cold restart • B#16#88: Master: Request for automatic cold restart • B#16#8A: Master: Request for manual warm restart • B#16#8B: Master: Request for automatic warm restart • B#16#8C: Standby: Request for manual restart • B#16#8D: Standby: Request for automatic restart
OB10x_PRIORITY	BYTE	Priority class: 27
OB10x_OB_NUMBR	BYTE	OB number (100, 101, or 102)
OB10x_RESERVED_1	BYTE	Reserved
OB10x_RESERVED_2	BYTE	Reserved
OB10x_STOP	WORD	Number of the event that caused the CPU to stop
OB10x_STRT_INFO	DWORD	Supplementary information about the current startup
OB10x_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

The following table shows the variables OB100_STR_INFO and OB101_STR_INFO.

Bit No.	Meaning	Possible Binary Values	Explanation
31 - 24	Startup information	0000 xxxx	Rack number 0 (H CPUs only)
		0100 xxxx	Rack number 1 (H CPUs only)
		1000 xxxx	Rack number 2 (H CPUs only)
		0001 xxxx	Multicomputing (S7-400 only)
		0010 xxxx	Operation of more than one CPU in the segmented rack (S7-400 only)
		xxxx xxx0	No difference between expected and actual configuration (S7-300 only)
		xxxx xxx1	Difference between expected and actual configuration (S7-300 only)
		xxxx xx0x	No difference between expected and actual configuration
		xxxx xx1x	Difference between expected and actual configuration
		xxxx x0xx	Not an H CPU
		xxxx x1xx	H CPU
		xxxx 0xxx	Clock for time stamp not battery-backed at last POWER ON

Bit No.	Meaning	Possible Binary Values	Explanation
		xxxx 1xxx	Clock for time stamp battery-backed at last POWER ON
23 - 16	Startup just completed	0000 0001	Warm restart in multicomputing without changing setting on the CPU according to parameter assignment (S7-400 only)
		0000 0011	Restart (warm) triggered with mode selector
		0000 0100	Restart (warm) triggered by command via MPI
		0000 0101	Cold restart in multicomputing without changing setting on the CPU according to parameter assignment (S7-400 only)
		0000 0011	Cold restart triggered with mode selector
		0000 1000	Cold restart triggered by command via MPI
		0000 1010	Hot restart in multicomputing without changing setting on the CPU according to parameter assignment (S7-400 only)
		0000 1011	Hot restart triggered with mode selector (S7-400 only)
		0000 1100	Hot restart triggered by command via MPI (S7-400 only)
		0001 0000	Automatic restart (warm) after battery-backed POWER ON
		0001 0001	Cold restart after battery-backed POWER ON according to parameter assignment
		0001 0011	Restart (warm) triggered with mode selector; last POWER ON battery-backed
		0001 0100	Restart (warm) triggered by command via MPI; last POWER ON battery-backed
		0010 0000	Automatic restart (warm) after battery-backed POWER ON (with memory reset by system)
		0010 0001	Cold restart after battery-backed POWER ON (with memory reset by system)
		0010 0011	Restart (warm) triggered with mode selector; last POWER ON not battery-backed
		0010 0100	Restart (warm) triggered by command via MPI; last POWER ON not battery-backed
		1010 0000	Automatic hot restart after battery-backed POWER ON according to parameter assignment (S7-400 only)
15 - 12	Permissibility of automatic startup	0000	Automatic startup illegal, memory reset requested
		0001	Automatic startup illegal, parameter modifications, etc. necessary
		0111	Automatic restart (warm) permitted
		1111	Automatic restart (warm/hot) permitted (S7-400 only)

Bit No.	Meaning	Possible Binary Values	Explanation
11 - 8	Permissibility of manual startup	0000	Startup illegal, memory reset requested
		0001	Startup illegal, parameter modifications, etc. necessary
		0111	Restart (warm) permitted
		1111	Restart (warm/hot) permitted (S7-400 only)
7 - 0	Last valid intervention or setting of the automatic startup at POWER ON	0000 0000	No startup
		0000 0001	Warm restart in multicomputing without changing setting on the CPU according to parameter assignment (S7-400 only)
		0000 0011	Restart (warm) triggered by mode selector
		0000 0100	Restart (warm) triggered by command via MPI
		0000 0101	Hot restart in multicomputing without changing setting on the CPU according to parameter assignment (S7-400 only)
		0000 0111	Cold restart triggered with mode selector
		0000 1000	Cold restart triggered by command via MPI
		0000 1010	Hot restart in multicomputing without changing setting on the CPU according to parameter assignment (S7-400 only)
		0000 1011	Hot restart triggered with mode selector (S7-400 only)
		0000 1100	Hot restart triggered by command via MPI (S7-400 only)
		0001 0000	Automatic Restart (warm) after battery-backed POWER ON
		0001 0001	Cold restart after battery-backed POWER ON according to parameter assignment
		0001 0011	Restart (warm) triggered with mode selector; last POWER ON battery-backed
		0001 0100	Restart (warm) triggered by command via MPI; last POWER ON battery-backed
		0010 0000	Automatic Restart (warm) after battery-backed POWER ON (with memory reset by system)
		0010 0001	Cold restart after battery-backed POWER ON according to parameter assignment
		0010 0011	Restart (warm) triggered with mode selector; last POWER ON not battery-backed
		0010 0100	Restart (warm) triggered by command via MPI; last POWER ON not battery-backed
		1010 0000	Automatic hot restart after battery-backed POWER ON according to parameter assignment (S7-400 only)

1.25 Programming Error Organization Block (OB121)

Description

The operating system of the CPU calls OB121 whenever an event occurs that is caused by an error related to the processing of the program. For example, if your program calls a block that has not been loaded on the CPU, OB121 is called.

Understanding the Operation of the Programming Error OB

OB121 is executed in the same priority class as the interrupted block.

If OB121 is not programmed, the CPU changes from the RUN mode to the STOP mode.

S7 provides the following SFCs for masking and unmasking start events for OB121 during the execution of your program:

- SFC36 (MSK_FLT): masks specific error codes
- SFC37 (DMSK_FLT): unmasks the error codes that were masked by SFC36
- SFC38 (READ_ERR): reads the error register

Local Data for the Programming Error OB

The following table describes the temporary (TEMP) variables for programming error OB. The variable names are the default names of OB121.

Variable	Type	Description
OB121_EV_CLASS	BYTE	Event class and identifiers: B#16#25
OB121_SW_FLT	BYTE	Error code : (possible values: B#16#21, B#16#22, B#16#23, B#16#24, B#16#25, B#16#26, B#16#27, B#16#28, B#16#29, B#16#30, B#16#31, B#16#32, B#16#33, B#16#34, B#16#35, B#16#3A, B#16#3C, B#16#3D, B#16#3E, B#16#3F)
OB121_PRIORITY	BYTE	Priority class = priority class of the OB in which the error occurred
OB121_OB_NUMBR	BYTE	OB number (121)
OB121_BLK_TYPE	BYTE	Type of block where the error occurred (no valid value is entered here in case of S7-300): B#16#88: OB, B#16#8A: DB, B#16#8C: FC, B#16#8E: FB
OB121_RESERVED_1	BYTE	Reserved
OB121_FLT_REG	WORD	Source of the error (depends on error code). For example: • Register where the conversion error occurred • Incorrect address (read/write error) • Incorrect timer/counter/block number • Incorrect memory area

Variable	Type	Description
OB121_BLK_NUM	WORD	Number of the block with the MC7 command that caused the error (no valid number is entered here for an S7-300)
OB121_PRG_ADDR	WORD	Relative address of the MC7 command that caused the error (no valid value is entered here for an S7-300)
OB121_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

The variables dependent on the error code have the following meaning:

Error code	Meaning
B#16#21:	BCD conversion error
OB121_FLT_REG:	ID for the register concerned (W#16#0000: accumulator 1)
B#16#22:	Area length error when reading
B#16#23:	Area length error when writing
B#16#28:	Read access to a byte, word, or double word with a pointer whose bit address is not 0.
B#16#29:	Write access to a byte, word, or double word with a pointer whose bit address is not 0. Incorrect byte address. The data area and access type can be read from OB121_RESERVED_1.
OB121_FLT_REG:	• Bits 7 to 4 access type. - 0: bit access, - 1: byte access, - 2: word access, - 3: double word access
OB121_RESERVED_1:	• Bits 3 to 0 memory area: - 0: I/O area - 1: process-image input table - 2: Process-image output table - 3: bit memory - 4: global DB - 5: instance DB - 6: own local data - 7: local data of caller
B#16#24:	Range error when reading
B#16#25:	Range error when writing
OB121_FLT_REG:	Contains the ID of the illegal area in the low byte (B#16#86 of own local data area)
B#16#26:	Error for timer number
B#16#27:	Error for counter number
OB121_FLT_REG:	Illegal number
B#16#30:	Write access to a write-protected global DB
B#16#31:	Write access to a write-protected instance DB
B#16#32:	DB number error accessing a global DB
B#16#33:	DB number error accessing an instance DB

Error code	Meaning
OB121_FLT_REG:	Illegal DB number
B#16#34:	FC number error in FC call
B#16#35:	FB number error in FB call
B#16#3A:	Access to a DB that has not been loaded; the DB number is in the permitted range
B#16#3C:	Access to an FC that has not been loaded; the FC number is in the permitted range
B#16#3D:	Access to an SFC that has not been loaded; the SFC number is in the permitted range
B#16#3E:	Access to an FB that has not been loaded; the FB number is in the permitted range
B#16#3F:	Access to an SFB that has not been loaded; the SFB number is in the permitted range
OB121_FLT_REG:	Illegal number

1.26 I/O Access Error Organization Block (OB122)

Description

The operating system of the CPU calls OB122 whenever an error occurs while accessing data on a module. For example, if the CPU detects a read error when accessing data on an I/O module, the operating system calls OB122.

Understanding the Operation of the I/O Access Error OB

OB122 is executed in the same priority class as the interrupted OB. If OB122 is not programmed, the CPU changes from the RUN mode to the STOP mode.

S7 provides the following SFCs for masking and unmasking start events for OB122 during the execution of your program:

- SFC36 (MSK_FLT): masks specific error codes
- SFC37 (DMSK_FLT): unmasks the error codes that were masked by SFC36
- SFC38 (READ_ERR): reads the error register

Local Data for the I/O Access Error OB

The following table describes the temporary (TEMP) variables for the I/O access error OB. The variable names are the default names of OB122.

Variable	Type	Description
OB122_EV_CLASS	BYTE	Event class and identifiers: B#16#29
OB122_SW_FLT	BYTE	Error code: B#16#42 - For S7-300 and CPU 417: I/O access error, reading For all other S7-400 CPUs: error during the first read access after an error occurred B#16#43 - For S7-300 and CPU 417: I/O access error, writing For all other S7-400 CPUs: error during the first write access after an error occurred B#16#44 - (Only for S7-400, excluding CPU 417) error during the n-th (n > 1) read access after an error has occurred B#16#45 - (Only for S7-400, excluding CPU 417) error during the n-th (n > 1) write access after an error has occurred
OB122_PRIORITY	BYTE	Priority class: • Priority class of the OB where the error occurred
OB122_OB_NUMBR	BYTE	OB number (122)
OB122_BLK_TYPE	BYTE	Type of block where the error occurred (B#16#88: OB, B#16#8A: DB, B#16#8C: FC, B#16#8E: FB) (no valid number is entered here for an S7-300)
OB122_MEM_AREA	BYTE	Memory area and access type: • Bit 7 to 4: Access type - 0: Bit access - 1: Byte access - 2: Word access - 3: DWord access • Bit 3 to 0: memory area - 0: I/O area - 1: Process image of the inputs 2: Process image of the outputs
OB122_MEM_ADDR	WORD	Memory address where the error occurred
OB122_BLK_NUM	WORD	Number of the block with the MC7 command that caused the error (no valid number is entered here for an S7-300)
OB122_PRG_ADDR	WORD	Relative address of the MC7 command that caused the error (no valid number is entered here for an S7-300)
OB122_DATE_TIME	DATE_AND_TIME	DATE_AND_TIME of day when the OB was called

2 Common Parameters for SFCs

2.1 Evaluating Errors with the Output Parameter RET_VAL

Types of Error Information

A system function (SFC) executed in your user program indicates whether or not the CPU was able to execute the function of the SFC successfully.
You can obtain information about any errors that occurred in two ways:

- In the BR bit of the status word
- In the output parameter RET_VAL (return value)

Note

Before evaluating the output parameters specific to an SFC, you should always follow the steps below:

- First, evaluate the BR bit of the status word.
- Then check the output parameter RET_VAL.
- If the BR bit indicates that an error has occurred or if RET_VAL contains a general error code, you must not evaluate the SFC output parameter!

Error Information in the Return Value

A system function (SFC) indicates that an error occurred during its execution by entering the value "0" in the binary result bit (BR) of the status word. Some system functions provide an additional error code at an output known as the return value (RET_VAL) output. If a general error is entered in the output parameter RET_VAL (see below for explanation), this is only indicated by the value "0" in the BR bit of the status word.

The return value is of the data type integer (INT). The relationship of the return value to the value "0" indicates whether or not an error occurred during execution of the function.

CPU Execution of the SFC	BR	Return Value	Sign of the Integer
With error(s)	0	less than "0"	negative (sign bit is "1")
Without error	1	greater than or equal to "0"	positive (sign bit is "0")

Reactions to Error Information

There are two different types of error code in RET_VAL as follows:

- A general error code, that all system functions can output and
- A specific error code, that the system function can output and which relates to its specific function.

You can write your program so that it reacts to the errors that occur during execution of a system function. This way you prevent further errors occurring as a result of the first error.

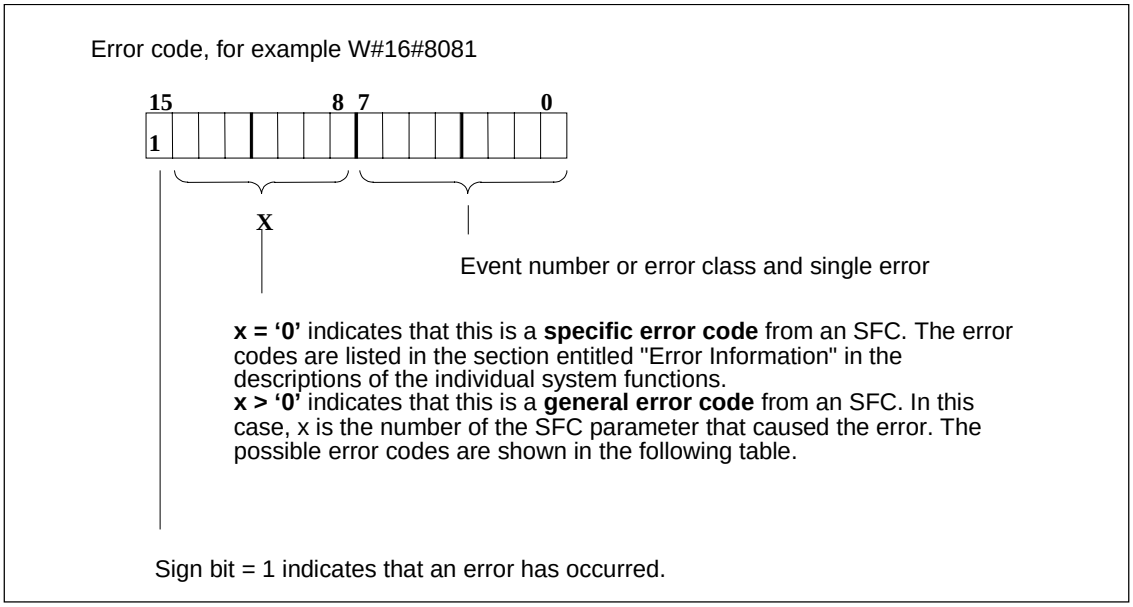
General and Specific Error Information

The return value (RET_VAL) of a system function provides one of the two following types of error codes:

- A general error code, that relates to errors that can occur in any system function.
- A specific error code, that relates only to the particular system function.

Although the data type of the output parameter RET_VAL is integer (INT), the error codes for system functions are grouped according to hexadecimal values. If you want to examine a return value and compare the value with the error codes listed in this manual, then display the error code in hexadecimal format.

The figure below shows the structure of a system function error code in hexadecimal format.

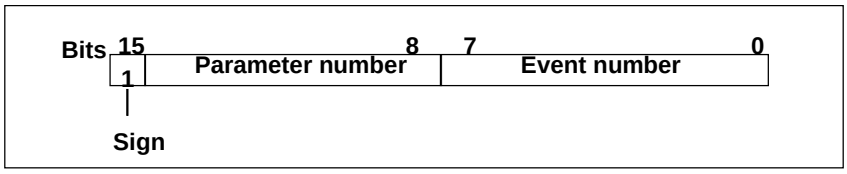


General Error Information

The general error code indicates errors that can occur in any system function. A general error code consists of the following two numbers:

- A parameter number from 1 to 111, where 1 indicates the first parameter, 2 indicates the second parameter of the SFC, etc.
- An event number from 0 to 127. The event number indicates that a synchronous error occurred.

The following table lists the codes for general errors and an explanation of each error.



Note

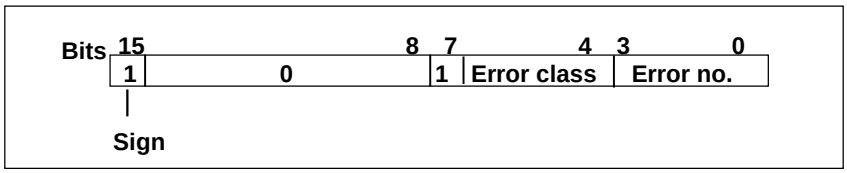
If a general error code was entered in RET_VAL, the following situations are possible:

- The action associated with the SFC may have been started or already completed.
- A specific SFC error may have occurred when the action was performed. As a result of a general error that occurred later, the specific error could, however, no longer be indicated.

Specific Error Information

Some system functions (SFCs) have a return value that provides a specific error code. This error code indicates that an error pertaining to a particular system function occurred during the execution of the function (see figure). A specific error code consists of the following two numbers:

- An error class from 0 to 7.
- An error number from 0 to 15.



General Error Codes

The following table explains the general error codes of a return value. The error code is shown in hexadecimal format. The letter x in each code number is simply a place holder and represents the number of the system function parameter that caused the error.

Error Code (W#16#...)	Explanation
8x7F	Internal error This error code indicates an internal error at parameter x. This error was not caused by the user and cannot be eliminated by the user.
8x22 8x23	Range length error when reading a parameter. Range length error when writing a parameter. This error code indicates that the parameter x is located either entirely or partly outside the range of an address or that the length of a bit range is not a multiple of 8 with an ANY parameter.
8x24 8x25	Range error when reading a parameter. Range error when writing a parameter. This error code indicates that the parameter x is located in a range that is illegal for the system function. Refer to the descriptions of the individual functions for information about the illegal ranges.
8x24 8x25	Range error when reading a parameter. Range error when writing a parameter. This error code indicates that the parameter x is located in a range that is illegal for the system function. Refer to the descriptions of the individual functions for information about the illegal ranges.
8x26	The parameter contains a timer number that is too high. This error code indicates that the timer specified in parameter x does not exist.
8x27	The parameter contains a counter number that is too high (counter number error). This error code indicates that the counter specified in parameter x does not exist.
8x28 8x29	Alignment error when reading a parameter. Alignment error when writing a parameter. This error code indicates that the reference to parameter x is a bit address that is not equal to 0.
8x30 8x31	The parameter is located in a read-only global DB. The parameter is located in a read-only instance DB. This error code indicates that parameter x is located in a read-only data block. If the data block was opened by the system function itself, the system function always returns the value W#16#8x30.
8x32 8x34 8x35	The parameter contains a DB number that is too high (DB number error). The parameter contains an FC number that is too high (FC number error). The parameter contains an FB number that is too high (FB number error). This error code indicates that parameter x contains a block number higher than the highest permitted number.
8x3A 8x3C 8x3E	The parameter contains the number of a DB that is not loaded. The parameter contains the number of an FC that is not loaded. The parameter contains the number of an FB that is not loaded.
8x42 8x43	An access error occurred while the system was attempting to read a parameter from the peripheral input area. An access error occurred while the system was attempting to write a parameter to the peripheral output area.
8x44 8x45	Error in the nth (n > 1) read access after an error occurred. Error in the nth (n > 1) write access after an error occurred. This error code indicates that access to the required parameter is denied.

2.2 Meaning of the Parameters REQ, RET_VAL and BUSY with Asynchronous SFCs

Asynchronous SFCs

SFCs that operate asynchronously are SFCs that are called more than once before they complete their functions. The following SFCs are either always executed asynchronously or in certain situations:

- SFC 7 "DP_PRAL"
- SFC 11 "DPSYC_FR"
- SFC 12 "D_ACT_DP"
- SFC 13 "DPNRM_DG"
- SFC 51 "RDSYSST"
- SFC 55 "WR_PARM"
- SFC 56 "WR_DPARM"
- SFC 57 "PARM_MOD"
- SFC 58 "WR_REC"
- SFC 59 "RD_REC"
- SFC 65 "X_SEND"
- SFC 67 "X_GET"
- SFC 68 "X_PUT"
- SFC 69 "X_ABORT"
- SFC 72 "I_GET"
- SFC 73 "I_PUT"
- SFC 74 "I_ABORT"
- SFC82 "CREA_DBL"
- SFC83 "READ_DBL"
- SFC84 "WRIT_DBL"
- SFC90 "H_CTRL"
- SFC102 "RD_DPARA"

Identifying the Job

If you trigger a hardware interrupt, output control commands to DP slaves, start a data transfer, or abort a non-configured connection with one of the SFCs listed above and then call the same SFC again before the current job is completed, the reaction of the SFC will depend on whether or not the second call involves the same job.

The following table explains which input parameters specify the job for each of these SFCs. If these parameters match those of a job that is not yet completed, the SFC call counts as a follow-on call.

SFC	Job is Identified by ...
7 "DP_PRAL"	IOID, LADDR
11"DPSYC_FR"	LADDR, GROUP, MODE
13 "DPNRM_DG"	LADDR
51 "RDSYSST"	SSL_ID, INDEX
55 "WR_PARM"	IOID, LADDR, RECNUM
56 "WR_DPARM"	IOID, LADDR, RECNUM
57 "PARM_MOD"	IOID, LADDR
58 "WR_REC"	IOID, LADDR, RECNUM
59 "RD_REC"	IOID, LADDR, RECNUM
65 "X_SEND"	DEST_ID, REQ_ID
67 "X_GET"	DEST_ID, VAR_ADDR
68 "X_PUT"	DEST_ID, VAR_ADDR
69 "X_ABORT"	DEST_ID
72 "I_GET"	IOID, LADDR, VAR_ADDR
73 "I_PUT"	IOID, LADDR, VAR_ADDR
74 "I_ABORT"	IOID, LADDR
82 "CREA_DBL"	LOW_LIMIT, UP_LIMIT, COUNT, ATTRIB, SRCBLK
83 "READ_DBL"	SRCBLK, DSTBLK
84 "WRIT_DBL"	SRCBLK, DSTBLK
90 "H_CTRL"	MODE, SUBMODE
102 "RD_DPARA"	LADDR, RECNUM

Input Parameter REQ

The REQ (request) input parameter is used solely to start the job:

- If you call the SFC for a job that is not currently active, the job is started by REQ = 1 (situation 1).
- If a particular job has been started and not yet completed and you call the SFC again to perform the same job (for example, in a cyclic interrupt OB), then REQ is not evaluated by the SFC (situation 2).

Output Parameters RET_VAL and BUSY

The status of the job execution is indicated by the output parameters RET_VAL and BUSY.

Refer also to the note in Evaluating Errors with the Output Parameter RET_VAL.

- In Case 1 (first call with REQ=1), W#16#7001 is entered in RET_VAL if system resources are free and the input parameters are correct. BUSY is then set.

  If the required system resources are currently being used or the input parameters have errors, the corresponding error code is entered in RET_VAL and BUSY has the value 0.

- In Case 2 (call while the same job is active), W#16#7002 is entered in RET_VAL (this is a warning that the job is still being processed), and BUSY is set.

- The following applies to the last call for a job:

  - With SFC 13 "DPNRM_DG," SFC 67 "X_GET" and SFC 72 "I_GET" the number of supplied data is entered in RET_VAL as a positive number of bytes if no error occurred. BUSY then has the value 0.
    If an error occurs, RET_VAL contains the error information and BUSY has the value 0.

  - With SFC 59 "RD_REC" the size of the data record in bytes is entered in RET_VAL or the value 0 if no error occurred (refer to Reading a Data Record with the SFC 59 "RD_REC" !). In this case, BUSY has the value 0. If an error occurs, the error code is entered in RET_VAL and BUSY has the value 0.)

  - With all other SFCs, if the job was executed error-free, 0 is entered in RET_VAL, and BUSY has the value 0. If an error occurs, the error code is entered in RET_VAL and BUSY has the value 0.

Note

If the first and last call come together, the reaction is the same for RET_VAL and BUSY as described for the last call.

Overview

The following table provides you with an overview of the relationships explained above. In particular, it shows the possible values of the output parameters if the execution of the job is not completed after an SFC has been called.

Note

Following every call, you must evaluate the relevant output parameters in your program.

Relationship between Call, REQ, RET, RET_VAL, and BUSY during the execution of a job.

Number of the Call	Type of Call	REQ	RET_VAL	BUSY
1	First call	1	W#16#7001	1
			Error code	0
2 to (n - 1)	Intermediate call	Irrelevant	W#16#7002	1
N	Last call	Irrelevant	W#16#0000 (exceptions: SFC 59 "RD_REC" if the destination area is larger than the data record transferred and SFC 13 "DPNRM_DG," SFC 67 "X_GET" and SFC 72 "I_GET"), if no error has occurred	0
			Error code if errors occurred	0

3 Copy and Block Functions

3.1 Copying Variables with SFC 20 "BLKMOV"

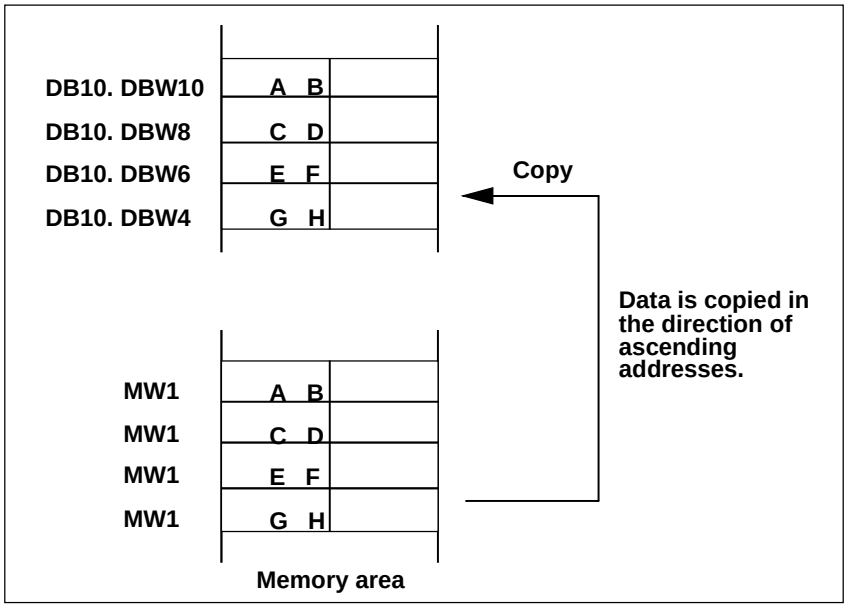
Description

You use SFC 20 "BLKMOV" (block move) to copy the contents of a memory area (= source area) to another memory area (= destination area).

Using SFC 20 "BLKMOV" you can copy all memory areas except:

- The following block types: FB, SFB, FC, SFC, OB, SDB,
- Counters,
- Timers,
- Memory areas of the peripheral I/O areas.

The source parameter can be a part of the data block in the load memory which is not relevant to program execution (DB compiled with the keyword UNLINKED).



Interruptability

As long as the source area is not part of a data block that only exists in the load memory, there is no limit to the nesting depth.

If, however, SFC 20 is interrupted while copying from a DB that is not relevant to program execution, the execution of SFC 20 can no longer be nested.

Parameter	Declaration	Data Type	Memory Area	Description
SRCBLK	INPUT	ANY	I, Q, M, D, L	Specifies the memory area to be copied (source area). Arrays of the data type STRING are not permitted.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs when the function is being executed, the return value contains an error code.
DSTBLK	OUTPUT	ANY	I, Q, M, D, L	Specifies the memory area to which the data will be copied (destination area). Arrays of the data type STRING are not permitted.

Note

The source and destination areas must not overlap. If the specified destination area is larger than the source area, the function only copies as much data to the destination area as is contained in the source area.

If the specified destination area is smaller than the source area, the function only copies as much data as can be written to the destination area.

If the ANY pointer (source or destination) is of the type BOOL, the length specified must be divisible by 8; otherwise the SFC will not be executed.

Source or destination parameters (or both) should also be STRING data types. If the source is a string, only the current characters in the string will be copied. If the destination is a string, the current length of the number of copied characters will be written. No ARRAY OF STRING can be copied. This means that only "STRING 1" is permissible.

Special feature: If an unlinked data block is copied to the RAM with SFC 20 BLKMOV and loaded at the same time, for example, through the programming device, the SFC can be delayed up to several milliseconds. This results in a longer OB cycle and may alert the cycle monitoring. Avoid loading the block during the time in which the CPU is copying this block with SFC 20.

Error Information

Error Code (W#16#...)	Explanation
0000	No error
8091	Nesting depth exceeded. The source area does not lie in the execution-relevant data block.

3.2 Uninterruptible Copying of Variables with SFC 81 " "

Description

With SFC 81 "UBLKMOV" (uninterruptible block move), you can copy the contents of a memory area (= source area) consistently to a different memory area (= destination area). The copy operation cannot be interrupted by other operating system activities.

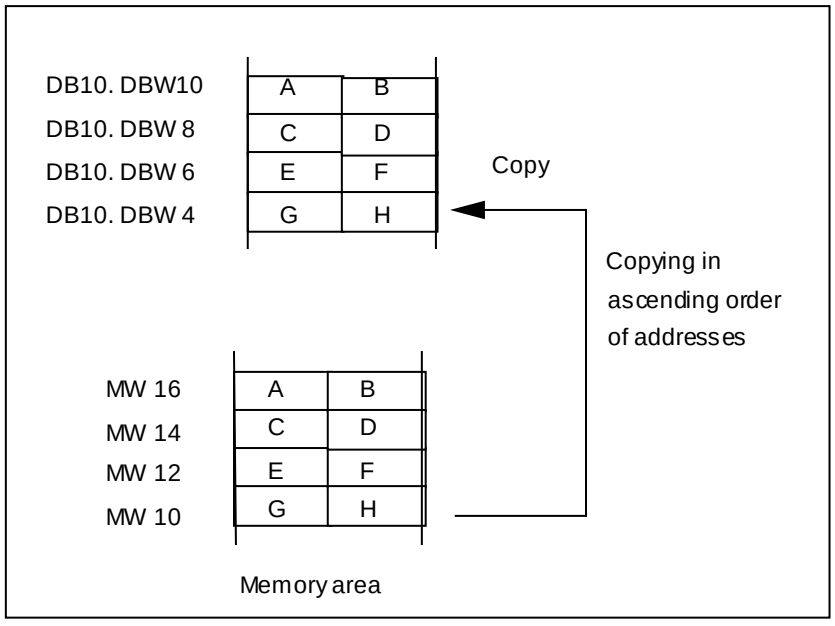
Using SFC 81 "UBLKMOV," you can copy all memory areas except:

- The following block types: FB, SFB, FC, SFC, OB, SDB
- Counters
- Timers
- Memory areas of the peripheral I/O areas

The source parameters can also be included in a the load memory of an unlinked data block (DB, compiled with the keyword UNLINKED)!

Note

If your CPU is equipped with the SFC83 you must use this SFC to read data blocks in the load memory which are not relevant for the execution. If you are using SFC20 the error message W#16#8092 is output.



Interruptability, Interrupt Reaction Times

Copying cannot be interrupted. Remember that if you use SFC 81 "UBLKMOV," this can increase the interrupt reaction times of your CPU.

Parameter	Declaration	Type	Memory Area	Description
SRCBLK	INPUT	ANY	I, Q, M, D, L	Specifies the memory area to be copied (source area). Arrays of the data type STRING are not permitted.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs when the function is being executed, the return value contains an error code.
DSTBLK	OUTPUT	ANY	I, Q, M, D, L	Specifies the memory area to which the data will be copied (destination area). Arrays of the data type STRING are not permitted.

Note

The source and destination areas must not overlap. If the specified destination area is larger than the source area, the function only copies as much data to the destination area as is contained in the source area.

If the specified destination area is smaller than the source area, the function only copies as much data as can be written to the destination area.

If the ANY pointer (source or destination) is of the type BOOL, the length specified must be divisible by 8; otherwise the SFC will not be executed.

If the ANY pointer is of the type STRING, the length specified must be 1.

Error Information

Error Code (W#16#...)	Description
0000	No error
8091	The source area is in an unlinked data block.
8092	The "Copy Variable " operation cannot be carried out because of access to a data block that is not executable. Here, use the SFC83.

3.3 Initializing a Memory Area with SFC 21 "FILL"

Description

With SFC 21 "FILL," you can initialize a memory area (destination area) with the contents of another memory area (source area). The SFC copies the contents of the specified destination area until the memory area is completely full.

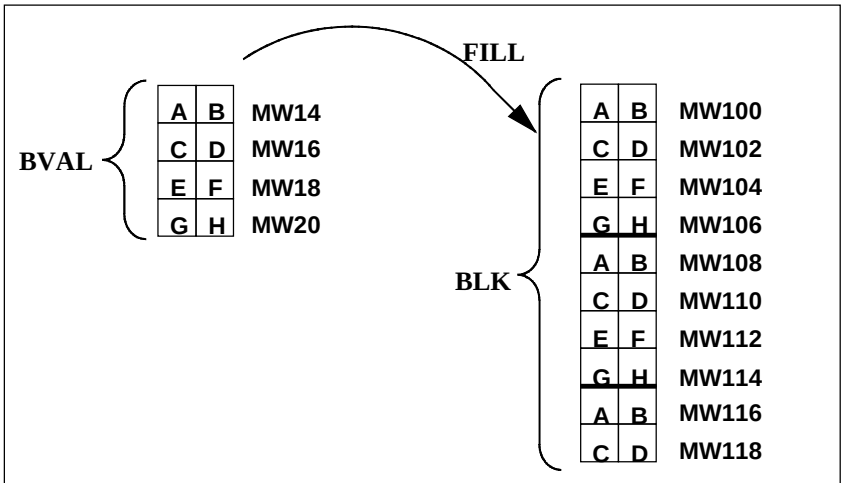
Note

The source and destination field must not overlap.

If the destination area to be initialized is not a whole multiple of the length of the input parameter BVAL, the destination area is nevertheless written up to the last byte.

If the destination area to be initialized is smaller than the source area, the function only copies as much data as can be written to the destination area.

If the ANY pointer (source or destination) is of the type BOOL, the length specified must be divisible by 8; otherwise the SFC will not be executed.



Example: The content of the area MW100 to MW118 will be written with the contents of memory words MW14 to MW20.

Exceptions

You cannot write values to the following using SFC 21:

- The following block types: FB, SFB, FC, SFC, SDB,
- Counters,
- Timers,
- Memory areas of the peripheral I/O area.

Parameter	Declaration	Data Type	Memory Area	Description
BVAL	INPUT	ANY	I, Q, M, D, L	The parameter BVAL contains the value or description of the area whose contents will be used to initialize the destination area (source area). Arrays of the data type STRING are not permitted.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being processed, the return value contains an error code.
BLK	OUTPUT	ANY	I, Q, M, D, L	The parameter BLK contains the description of the area to be initialized (destination area). Arrays of the data type STRING are not permitted.

The Input Parameter is a Structure

If you transfer a structure as the input parameter, remember the following point:

STEP 7 always defines the length of a structure as an even number of bytes. As a result, the structure will need one byte of additional memory space if you declare a structure with an odd number of bytes.

Example

The structure was declared as:

```
TYP_5_BYTE_STRUCTURE : STRUCT
   BYTE_1_2 : WORD
   BYTE_3_4 : WORD
   BYTE_5 : BYTE
END_STRUCT
```

The declared structure "TYP_5_BYTE_STRUCTURE" requires 6 bytes of memory.

3.4 Creating a Data Block with SFC 22 "CREAT_DB"

Description

With SFC 22 "CREAT_DB" (create data block), you create a data block that does not contain initialized values. The SFC creates a data block of a selectable length with a block number taken from a specified range. The SFC assigns the lowest possible number to the DB from the specified range. If you want to create a DB with a particular number, simply select the range specifying the same value as the upper and lower limit. You cannot assign a number if a DB with the same number already exists in the user program. The length of the DB must be an even number of bytes.

Interruptability

SFC 22 "CREAT_DB" can be interrupted by higher priority OBs. If SFC 22 "CREAT_DB" is called again in a higher priority OB, the call is rejected with error code W#16#8091.

Parameter	Declaration	Data Type	Memory Area	Description
LOW_LIMIT	INPUT	WORD	I, Q, M, D, L, constant	The lower limit value is the smallest number in the range of numbers that you can assign to your data block.
UP_LIMIT	INPUT	WORD	I, Q, M, D, L, constant	The upper limit value is the highest number in the range of numbers you can assign to your data block.
COUNT	INPUT	WORD	I, Q, M, D, L, constant	The count value specifies the number of data bytes you want to reserve for your data block. Here you must specify an even number of bytes (maximum 65534).
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains an error code.
DB_NUMBER	OUTPUT	WORD	I, Q, M, D, L	The data block number is the number of the created data block. If an error occurs, (bit 15 of RET_VAL was set) the value 0 is entered in DB_NUMBER.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8091	You have called SFC 22 nested.
8092	The "create DB" function cannot be executed currently because • The "compress user memory" function is currently active

Error Code (W#16#...)	Explanation
	• The "compress user program" function is currently active. • The H CPU runs coupling or update functions. • The WinAC Software CPU has detected an error in the operating system of the computer where WinAC is installed.
80A1	Error in the number of the DB: • The number is 0. • The number exceeds the number of DBs for the specific CPU. • Parameter lower limit > upper limit.
80A2	Error in the length of the DB: • The length is 0. • The length was specified as an odd number. • The length is greater than permitted by the CPU.
80B1	There is no DB number free.
80B2	There is not enough free memory available.
80B3	There is not enough continuous memory space available (remedy: compress memory!)

3.5 Deleting a Data Block with SFC 23 "DEL_DB"

Description

With SFC 23 "DEL_DB" (delete data block) you delete a data block located in the work memory and, if present, in the load memory of the CPU. The DB to be deleted must not be open in the current or in any lower priority class, in other words, it must not be entered in either of the two DB registers or in the B stack. Otherwise the CPU starts OB°121 when SFC 23 is called. If OB°121 is not present the CPU switches to the STOP mode.

The following table explains when a DB can be deleted with SFC 23 "DEL-DB."

If ...	Then ...
The DB was created by calling SFC 22 "CREAT_DB,"	SFC 23 can delete it.
The DB was transferred to the CPU by STEP 7 and was not created with the keyword UNLINKED,	SFC 23 can delete it.
The DB is located on a flash card,	SFC 23 cannot delete it.

Interruptability

SFC 23 "DEL_DB" can be interrupted by priority classes of a higher priority. If the SFC is again called there, then this second call is aborted and W#16#8091 is entered in RET_VAL.

Parameter	Declaration	Data Type	Memory Area	Description
DB_NUMBER	INPUT	WORD	I, Q, M, D, L, constant	Number of the DB to be deleted
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8091	SFC 23 calls were nested and the maximum nesting level of the CPU used was exceeded.
8092	The "delete DB" function cannot be executed currently because • The "compress user memory" function is currently active. • The "save user program" function is currently active. • The "upload" function is currently active on the DB to be deleted. • The H CPU runs coupling or update functions. • WinAC Software CPU has detected an error in the operating system of the computer where WinAC is installed
80A1	Error in the input parameter DB_NUMBER: the actual parameter selected • Is 0. • Is greater than the maximum permitted DB number for the CPU used.
80B1	The DB with the specified number does not exist on the CPU.
80B2	The DB with the specified number created using the keyword UNLINKED.
80B3	The DB is on a flash card.
80B4	The DB cannot be deleted because it is a block in the F-Library
80C1	The "Delete DB" function cannot be executed at this time due to a temporary resource bottleneck.

3.6 Testing a Data Block with SFC 24 "TEST_DB"

Description

With SFC 24 "TEST_DB" (test data block), you obtain information about a data block located in the work memory of the CPU. The SFC queries the number of data bytes in the selected DB and checks whether or not the DB is read only.

Parameter	Declaration	Data Type	Memory Area	Description
DB_NUMBER	INPUT	WORD	I, Q, M, D, L, constant	Number of the DB to be tested
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
DB_LENGTH	OUTPUT	WORD	I, Q, M, D, L	Number of data bytes the selected DB contains.
WRITE_PROT	OUTPUT	BOOL	I, Q, M, D, L	Information about the write-protect identifier of the DB (1 means read only).

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
80A1	Error in the input parameter DB_NUMBER: the actual parameter selected • Is 0 • Is greater than the max. permissible DB number for the CPU used.
80B1	The DB with the specified number does not exist on the CPU.
80B2	The DB was created using the keyword UNLINKED.

3.7 Compressing the User Memory with SFC 25 "COMPRESS"

Gaps in Memory

Gaps can occur in the load memory and in the work memory if data blocks are deleted and reloaded several times. These gaps reduce the effective memory area.

Description

With SFC 25 "COMPRESS," you start compression of the RAM section of both the load memory and the work memory. The compression function is the same as when started externally in the RUN-P mode (mode selector setting).

If compression was started externally and is still active, the SFC 25 call will result in an error message.

Note

Data blocks with a length greater than 1000 bytes are not shifted with SFC 25 "COMPRESS." This means that gaps may still remain in the work memory after compression.

Parameter	Declaration	Data Type	Memory Area	Description
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
BUSY	OUTPUT	BOOL	I, Q, M, D, L	Indicates whether the compression function started by an SFC 25 call is still active. (1 means active.)
DONE	OUTPUT	BOOL	I, Q, M, D, L	Indicates whether the compression function started by SFC 25 was completed successfully. (1 means completed successfully.)

Checking the Compression Function

If SFC 25 "COMPRESS" is called once, the compression function is started. You cannot, however, check whether the memory was successfully compressed.

If you want to check the compression function, follow the steps outlined below:

Call SFC 25 cyclically. First evaluate the parameter RET_VAL after every call. Provided that its value is 0, the parameters BUSY and DONE can be evaluated. If BUSY = 1 and DONE = 0, this indicates that the compression function is still active. When BUSY changes to value 0 and DONE to the value 1, this indicates that the compression function was completed successfully. If SFC 25 is called again afterwards, the compression function is started again.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred. The compression function was started by SFC 25. Evaluation of the output parameters BUSY and DONE by the user program (see above) only provides useful information when this is the case.
8091	The compression function was started externally and is still active.
8092	The "compress user memory" function cannot currently be executed because • The "delete data block" function was started externally and is still active • A test and startup function currently requires a particular block (for example, status) • The "copy blocks" function was triggered externally and is still active. • The H- CPU runs coupling or update functions.

3.8 Transferring a Substitute Value to Accumulator 1 with SFC 44 "REPL_VAL"

Description

With SFC 44 "REPL_VAL" (replace value), you transfer a value to accumulator 1 of the priority class that caused the error.

Restriction: Only in Synchronous Error OBs

You can only call SFC 44 "REPL_VAL" in a synchronous error OB (OB121, OB122).

Example of an Application

If an input module is damaged to such an extent that no more values can be read from it, then each time the module is accessed, OB122 is started. Using SFC 44 "REPL_VAL," a suitable value in OB122 can be transferred to accumulator 1 of the interrupted priority class so that the program can continue with this substitute value. The information for selecting the substitute value (for example, the block in which the error occurred or the address affected) is located in the local variables of OB122.

Parameter	Declaration	Data Type	Memory Area	Description
VAL	INPUT	DWORD	I, Q, M, D, L, constant	Substitute value
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value ontains an error code.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred. A substitute value was entered.
8080	SFC 44 was not called by a synchronous error OB (OB121, OB122).

3.9 Generating Data Blocks in Load Memory with SFC 82 "CREA_DBL"

Description

With SFC 82 "CREA_DBL" (create data block in load memory) you can create a new data block in the load memory (Memory Card). The SFC 82 generates a default size data block using a number from a specified range and. The SFC 82 assigns the smallest possible number to the DB. You can generate a DB with a certain number by assigning the same number to the upper and to the lower limit of the rage to be specified. You cannot assign numbers that are already assigned to DBs that exist in the user program. If a DB already exists with the same number in work memory and/or load memory or if the DB exists as copied version the SFC is terminated and an error message is generated.

Note

With the SFC 24 "TEST_DB" you can determine whether a DB with the same number already exists.

The content of the data area to which the parameter SRCBLK points are written to the DB. This data area must be a DB with the data type BLOCK_DB or an area from a DB. To maintain consistency, you must not change this data area while the SFC 82 is being processed.

A DB with READ_ONLY attribute can only be created and initialized by SFC 82.

The SFC 82 does not change the checksum of the user program.

Operating principle

The SFC 82 "CREA_DBL" operates asynchronously, that is, processing covers multiple SFC calls. Start the job by calling SFC 82 with REQ = 1.

Th job status is displayed via the output parameters RET_VAL and BUSY.

Refer also to Meaning of REQ, RET_VAL and BUSY for asynchronously operating SFCs

Parameters	Declaration	Data type	Memory area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	REQ = 1: Request to generate the DB
LOW_LIMIT	INPUT	WORD	I, Q, M, D, L	Lower limit of the range used by the SFC to assign a number to our DB
UP_LIMIT	INPUT	WORD	I, Q, M, D, L	Upper limit of the range used by the SFC to assign a number to our DB
COUNT	INPUT	WORD	I, Q, M, D, L	The count value specifies the quantity of data bytes you want to reserve for your DB. Here you must specify an even number of bytes.

Parameters	Declaration	Data type	Memory area	Description	
ATTRIB	INPUT	BYTE	I, Q, M, D, L	DB properties:	
				Bit 0 = 1:	UNLINKED: The DB exists only in load memory.
				Bit 1 = 1:	READ_ONLY: The DB is write protected.
				Bit 2 = 1:	NON_RETAIN: The DB is not retentive.
				Bit 3 to 7:	reserved
SRCBLK	INPUT	ANY	D	Pointer to the data area with which the DB is initialized	
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error code	
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: The process is not terminated.	
DB_NUM	OUTPUT	WORD	I, Q, M, D, L	Number of the generated DB	

Error Information

Error code (W#16#...):	Description
0000	No error
0081	The target range is larger than the source range. The source area is written completely to the target area. The remaining bytes of the target area are filled with 0.
7000	First call with REQ=0: no data transfer active; BUSY has the value 0.
7001	First call with REQ=1: no data transfer active; BUSY has the value 1.
7002	Intermediate call (REQ irrelevant): data transfer already active; BUSY has the value 1.
8081	The source range is larger than the target range. The target range is fully written. The remaining bytes of the source area are ignored.
8091	You have called the SFC 82 nested.
8092	The operation "Generate a DB" is currently not executable because • the "Compress Application Memory" operation is currently active • the H CPU being connected or updated • Windows NT operating system has crashed (blue screen)
8093	No block or no execution-relevant block is indicated for the SRCBLK (initializing block) parameter.
8094	A not yet supported attribute was specified for the ATTRIB parameter
80A1	DB number error: • the number is 0 • Lower limit > Upper Limit
80A2	DB length error: • the length is 0 • the length is an odd number • the length is higher than permitted by the CPU

Error code (W#16#...):	Description
80B1	No free DB number
80B2	Insufficient memory
80B3	Insufficient continuous memory (compress it)
80BB	Insufficient load memory
80C0	The target is currently being processed by another SFC or communication operation.
80C3	The required operating resources are currently occupied.
8xyy	General error codes, for example: • Source DB does not exist or it is only available as copied version • Source area in DB does not exist

3.10 Reading from a Data Block In Load Memory with SFC 83 "READ_DBL"

Description

Use SFC 83 "READ_DBL" (read data block in load memory) to fetch a DB of the data type BLOCK_DB or an area from a DB in load memory (Memory Card) to the data area of a target DB. The target DB must be relevant for execution (keyword UNLINKED = 0). The source area to be read does not need to be relevant to execution (keyword UNLINKED = 1). The content of the load memory is not changed during the read process.

To maintain consistency, you must not change the target area while the SFC83 is being processed.

The following restrictions apply to the parameters SRCBLK and DSTBLK:

- For an ANY pointer of the type BOOLEAN the length must be divisible by 8.
- For an ANY pointer of the type BOOLEAN the length must be divisible by 1.

If required, you can determine the length of the source area with SFC24 "TEST_DB".

Note

The SFC 83 is processed asynchronously. Therefore, it is not suitable for frequent reading of variables from load memory.

Note

Once started, a job is always completed, even if the same resource is requested by a job with higher priority. If error code 80C3 is displayed at the high-priority run level, it does not make sense to restart the high-priority job right away. Instead, you should wait until the blocking job is completed.

Working method

The SFC83 "READ_DBL" is an asynchronous operating SFC, that is, processing covers multiple SFC calls. Start the job by calling SFC83 with REQ = 1.

The job status is displayed via the output parameters RET_VAL and BUSY.

See also Meaning of REQ, RET_VAL and BUSY for Asynchronously Operating SFCs

Parameter	Declaration	Data type	Memory area	Description
REQ	INPUT	BOOL	E, A, M, D, L	REQ = 1: Read request
SRCBLK	INPUT	ANY	D	Pointer to the data area of the DB in the load memory from which is to be read

Parameter	Declaration	Data type	Memory area	Description
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error code
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: The read process is not yet terminated.
DSTBLK	OUTPUT	ANY	D	Pointer to the data area of the target DB

Error Information

Error code (W#16#...):	Description
0000	No error
8081	The source range is larger than the target range. The target range is fully written. The remaining bytes of the source area are ignored.
7000	First call with REQ=0: no data transfer active; BUSY has the value 0.
7001	First call with REQ=1: no data transfer active; BUSY has the value 1.
7002	Intermediate call (REQ irrelevant): data transfer already active; BUSY has the value 1.
0081	The target range is larger than the source range. The source area is written completely to the target area. The remaining bytes of the target area are not changed.
80C0	The target is currently being processed by another SFC or communication operation.
8093	No block or no execution-relevant block is indicated for the parameter DSTBLK (initializing block).
80B1	No block or no execution-relevant block is indicated for the parameter SRCBLK (initializing block).
80B4	DB with an F-attribute must not be changed
80C3	The required operating resources are currently occupied.
8xyy	General error codes

3.11 Writing a Data Block in Load Memory with SFC 84 "WRITE_DBL"

Description

With the SFC84 "WRIT_DBL" (write data block in load memory) you can write a source DB in a DB of the data type BLOCK DB or into an area of a DB in load memory (Memory Card). The DB in load memory to which the parameter DSTBLK refers can be relevant or not relevant for processing. The source area to which the parameter SRCBLK refers can be of data type BLOCK_DB or a DB(-content) in the work memory. The source DB to which the parameter SRCBLK refers must therefore be process relevant (keyword UNLINKED = 0). The source DB can also be generate with SFC22 "CREAT_DB".

To maintain consistency, you must not change the source area while the SFC 83 is being processed.

The following restrictions apply to the parameters SRCBLK and DSTBLK:

- For an ANY pointer of the type BOOLEAN the length must be divisible by 8.
- For an ANY pointer of the type STRING the length must be equal to 1.

If required, you can determine the length of the target DB with SFC24 "TEST_DB".

SFC82 does not change the checksum of the user program if you describe a DB that was generated via SFC. When writing a loaded DB the first entry in this DB changes the checksum of the user program.

Note

SFC84 is processed asynchronously. Therefore, it is not suitable for frequent writing of variables in the load memory. Frequent writing processes will, above that, shorten the useful life of the load memory.

Operating principle

The SFC 84 "CREA_DBL" operates asynchronously, that is, processing covers multiple SFC calls. Start the job by calling SFC84 with REQ = 1.

The job status is displayed via the output parameters RET_VAL and BUSY.

Refer also to Meaning of REQ, RET_VAL and BUSY with Asynchronously Operating SFCs

Parameters	Declaration	Data type	Memory area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	REQ = 1: Write request
SRCBLK	INPUT	ANY	D	Pointer to the data area of the source DB
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error code
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: The write process is not yet terminated.
DSTBLK	OUTPUT	ANY	D	Pointer to the data area of the DB in the load memory which is to be written

Error Information

Error code (W#16#...):	Description
0000	no error
0081	The target range is larger than the source range. The source area is written completely to the target area. The remaining bytes of the target area are not changed.
7000	First call with REQ=0: no data transfer active; BUSY has the value 0.
7001	First call with REQ=1: no data transfer active; BUSY has the value 1.
7002	Intermediate call (REQ irrelevant): data transfer already active; BUSY has the value 1.
8081	The source range is larger than the target range. The target range is fully written. The remaining bytes of the source area are ignored.
8092	Windows NT operating system has crashed (blue screen)
8093	No block or no execution-relevant block is indicated for the SRCBLK (initializing block) parameter.
80B1	No block or no execution-relevant block is indicated for the DSTBLK (initializing block) parameter.
80B4	A DB with F attribute must not be changed
80C0	The target is currently being processed by another SFC or communication operation.
80C3	The required operating resources are currently occupied.
8xyy	General error codes

4 SFCs for Controlling Program Execution

4.1 Re-triggering Cycle Time Monitoring with SFC 43 "RE_TRIGR"

Description

With SFC 43 "RE_TRIGR" (re-trigger watchdog), you can re-trigger the cycle time monitoring.

Parameters

SFC 43 "RE_TRIGR" has no parameters.

Error Information

SFC 43 "RE_TRIGR" does not provide any error information.

4.2 Changing the CPU to STOP with SFC 46 "STP"

Description

With SFC 46 "STP" (stop), you change the CPU to the STOP mode.

Parameters

SFC 46 "STP" does not have any parameters.

Error Information

SFC 46 "STP" does not provide any error information.

4.3 Delaying Execution of the User Program with SFC 47 "WAIT"

Description

With SFC 47 "WAIT," you program delays or waiting times in your user program. You can program waiting times up to 32767 µs. The smallest possible waiting time depends on the particular CPU and is the same as the execution time of SFC 47.

Interruptability

SFC 47 "WAIT" can be interrupted by higher priority OBs.

Note

(for S7-300 only, but not for CPU 318)

The waiting time programmed with SFC 47 is a minimum time. It is extended by the execution time of the nested priority classes and by load on the system.

Parameter	Declaration	Data Type	Memory Area	Description
WT	INPUT	INT	I, Q, M, D, L, constant	The parameter WT contains the waiting time in µs.

Error information

SFC 47 "WAIT" does not provide any error information.

4.4 Triggering a Multicomputing Interrupt with SFC 35 "MP_ALM"

Description

Calling SFC 35 "MP_ALM" during multicomputing triggers the multicomputing interrupt. This leads to a synchronized start of OB60 on all CPUs involved. In the single processor mode and when operating with a segmented rack, OB60 is only started on the CPU that called SFC 35.

You can indicate the cause of the multicomputing interrupt using the JOB input parameter. This job identifier is transferred to all the CPUs involved and you can evaluate it in the multicomputing interrupt (OB60) (refer to the **online documentation "Programming with STEP 7"**).

You can call SFC 35 "MP_ALM" at any point in your program. Since the call would be pointless in any mode other than RUN, if it is called in the STARTUP mode, the multicomputing interrupt is suppressed. The function value informs you of this.

Parameter	Declaration	Data Type	Memory Area	Description
JOB	INPUT	BYTE	I, Q, M, D, L, const.	Job identifier: Possible values: 1 to 15
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs during execution of the function, the return value contains an error code.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	The JOB input parameter contains an illegal value.
80A0	Execution of OB60 following the last multicomputing interrupt is not completed either on the local or on another CPU.
80A1	Incorrect mode (STARTUP instead of RUN).

5 SFCs for Handling the System Clock

5.1 Setting the Time with SFC 0 "SET_CLK"

Note

SFC 100 "SET_CLKS" now replaces SFC 0 "SET_CLK". In new programs, you should only use SFC 100.

Description

With SFC 0 "SET_CLK" (set system clock), you set the time and the date of the CPU clock. The SFC 0 call starts the clock. The clock then runs starting from the set time and set date.

If the clock is a master clock, the CPU also starts to synchronize the time when SFC 0 is called. You set the synchronization intervals using STEP 7.

Parameter	Declaration	Data Type	Memory Area	Description
PDT	INPUT	DT	D,L	At the PDT input, you enter the date and time you want to set.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs during the execution of the function, the return value contains an error code.

Date and Time

You enter the date and time as data type DT. As an example: for January 15th, 1995, 10:30 a.m. and 30 seconds you would enter: DT#1995-01-15-10:30:30. The time can only be entered with a precision of seconds. The day of the week is calculated by SFC 0 "SET_CLK" from the date.

Remember that you must first create the data type DT with the FC "D_TOD_DT" before you can transfer input parameters to it (see time functions; FC 3, FC 6, FC 7, FC 8, FC 33, FC 40, FC 1, FC 35, FC, FC 34).

Error Information

Error Code (W#16#...)	Explanation
0000	No error
8080	Error in date
8081	Error in time

5.2 Reading the Time with SFC 1 "READ_CLK"

Description

With SFC 1 "READ_CLK" (read system clock), you read the current date or current time of the system clock of the CPU.

Parameter	Declaration	Data Type	Memory Area	Description
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs during the execution of the function, the return value contains an error code.
CDT	OUTPUT	DT	D,L	The current date and current time are output at the CDT output.

Error Information

See Chapter Evaluating Errors with the Output Parameter RET_VAL

5.3 Synchronizing Slave Clocks with SFC 48 "SNC_RTCB"

Definition: Synchronizing Slave Clocks

Synchronizing slave clocks refers to the transmission of the date and time from the master clock of a bus segment (for example, the S7-400 K-bus, MPI, or S7 backplane bus) to all clock slaves of the bus segment.

Description

With SFC 48 "SNC_RTCB" (synchronize real time clocks) you synchronize all the slave clocks on a bus segment. Successful synchronization is only possible when SFC 48 is called on a CPU whose real-time clock was assigned the master clock function for at least one bus segment. You assign the relevant parameters with STEP 7.

The system synchronization of the slave clocks (cyclic after the selected synchronization interval has elapsed) is independent of SFC 48 calls.

Parameter	Declaration	Data Type	Memory Area	Description
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs during the execution of the function, the return value contains an error code.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred during synchronization.
0001	The existing clock was not assigned the master clock function for any of the bus segments.

5.4 Setting the Time-of-Day and the TOD Status with SFC 100 "SET_CLKS"

Description

Set the TOD and the TOD status for your CPU with SFC 100 "SET_CLKS".

Caution

Only use SFC100 if the TOD for your CPU is not going to synchronized. Otherwise, with every synchronization the master's TOD status would be applied. This would overwrite the value specified per SFC.

Via the MODE parameter you can specify whether to change only the TOD, only the TOD status or both. This is explained in the table below:

MODE (B#16#...)	Meaning
01	Setting the TOD The SFC call corresponds with the call of SFC 0 "SET_CLK". The input parameters CORR, SUMMER and ANN_1 are not evaluated.
02	Setting the TOD status The input parameter PDT is not evaluated. The remaining input parameters form the following TOD status elements: • Correction value including the sign • Announcement hour • Summer/Winter Time indicator The TOD resolution is matched to that of your CPU. The bit synchronization failure of the TOD status is indicated with FALSE. The TOD remains unchanged.
03	Setting the TOD and the TOD status

Note

You can determine the current TOD status of your CPU by retrieving SZL-ID W#16#0132 Index W#16#0008 with SFC 51 "RDSYSST".

Parameters	Declaration	Data type	Memory Area	Description	
MODE	INPUT	BYTE	I, Q, M, D, L, Const.	Operating mode Possible values:	
				B#16#01:	Setting the TOD
				B#16#02:	Setting the TOD status
				B#16#03:	Setting the TOD and the TOD status
PDT	INPUT	DT	D, L	Default TOD	
CORR	INPUT	INT	I, Q, M, D, L, Const.	Correction value (in 0.5 h pattern) Possible values: -24 to +26	
SUMMER	INPUT	BOOL	I, Q, M, D, L, Const.	Summer/Winter Time ID: • 0 = Winter Time • 1 = Summer Time	
ANN_1	INPUT	BOOL	I, Q, M, D, L, Const.	Announcement hour 1: At the next hourly change summer time is switched over to winter time or vice versa.	
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error code	

Error Information

Error code (W#16#...):	Explanation
0000	no error
8080	MODE out of the permitted value range
8081	CORR out of the permitted value range (only for MODE = B#16#02 or MODE = B#16#03)
8082	PDT out of the permitted value range: Illegal date and or TOD

6 SFCs for Handling Run-Time Meters

6.1 Runtime Meters

Introduction

The CPUs have a number of run-time meters (refer to the data sheets of your CPU). Using SFCs 2, 3, and 4, you can set, stop, and read run-time meters.

Application

You can use a run-time meter for a variety of applications:

- For measuring the run-time of the CPU
- For measuring the run-time of controlled apparatus or connected devices.

Characteristics of the Run-time Meter

When it is started, the run-time meter begins to count starting at the last recorded value. If you want it to start at a different initial value, you must set this value with SFC 2. If the CPU changes to the STOP mode, or you stop the run-time meter, the CPU records the current value of the run-time meter. When a warm restart or a cold restart of the CPU is executed, the run-time meter must be started again with SFC 3.

Range of Values

Each run-time meter has a range of values from 0 to 32 767 hours.

6.2 Setting the Runtime Meter with SFC 2 "SET_RTM"

Description

With SFC 2 "SET_RTM" (set run-time meter), you set a run-time meter of the CPU to a selected value. The number of run-time meters you can set depends on the particular CPU you are using.

Parameter	Declaration	Data Type	Memory Area	Description
NR	INPUT	BYTE	I, Q, M, D, L, constant	Input NR contains the number of the run-time meter you want to set (possible values: 0 to 7).
PV	INPUT	INT	I, Q, M, D, L, constant	Input PV contains the setting for the run-time meter (default).
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains an error code.

Error Information

Error Code (W#16#...)	Explanation
0000	No error
8080	Wrong number for the run-time meter
8081	A negative value was transferred to the PV parameter.

6.3 Starting and Stopping a Run-time Meter with SFC 3 "CTRL_RTM"

Description

With SFC 3 "CTRL_RTM" (control run-time meter), you can start or stop a run-time meter of the CPU.

Parameter	Declaration	Data Type	Memory Area	Description
NR	INPUT	BYTE	I, Q, M, D, L, constant	Input NR contains the number of the run-time meter you want to start or stop (possible values: 0 to 7).
S	INPUT	BOOL	I, Q, M, D, L, constant	Input S starts or stops the run-time meter. Set the signal state to "0" when you want to stop the counter. Set the signal state to "1" when you want to start the counter.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains an error code.

Error Information

Error Code (W#16#...)	Explanation
0000	No error
8080	Wrong number for the run-time meter

6.4 Reading a Runtime Meter with SFC 4 "READ_RTM"

Description

With SFC 4 "READ_RTM" (read run-time meter), you read a run-time meter. SFC 4 provides the current run time as output data and the status of the counter, for example, "stopped" or "counting."

If the run-time meter runs for longer than 32767 hours, it stops at the count 32767 and outputs the error message "overflow."

Parameter	Declaration	Data Type	Memory Area	Description
NR	INPUT	BYTE	I, Q, M, D, L, constant	Input NR contains the number of the run-time meter you want to read (possible values: 0 to 7).
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while processing the function, the return value contains an error code.
CQ	OUTPUT	BOOL	I, Q, M, D, L	Output CQ indicates whether the run-time meter is running or stopped. The signal state "0" shows that the run-time meter is stopped. Signal state "1" shows that the run-time meter is running.
CV	OUTPUT	INT	I, Q, M, D, L	Output CV indicates the current value of the run-time meter.

Error Information

Error Code (W#16#...)	Explanation
0000	No error
8080	Wrong number for the run-time meter
8081	Overflow of the run-time meter

6.5 Reading the System Time with SFC 64 "TIME_TCK"

Description

With SFC 64 "TIME_TCK" (time tick), you can read the system time of the CPU. The system time is a "time counter" counting cyclically from 0 to a maximum of 2147483647 ms. In case of an overflow the system time is counted again starting with 0. The resolution and the accuracy of the system time are 1 ms for the S7-400 and CPU 318 and 10 ms for all other S7-300 CPUs. The system time is influenced only by the operating modes of the CPU.

Application

You can use the system time for example, to measure the duration of processes by comparing the results of two SFC 64 calls.

System Time and Modes

Mode	System Time ...
Startup	... is constantly updated
RUN	
STOP	... is stopped and retains the current value
Hot restart (not with S7-300 and S7-400 H)	... continues with the value saved at the change to the STOP mode
Warm restart	... is deleted and restarts with "0"
Cold restart	

Parameter	Declaration	Data Type	Memory Area	Description
RET_VAL	OUTPUT	TIME	I, Q, M, D, L	The RET_VAL parameter contains the read system time in the range from 0 to 2^{31}-1 ms.

Error Information

SFC 64 "TIME_TCK" does not provide any error information.

7 SFCs for Transferring Data Records

7.1 Writing and Reading Data Records

Principle

Some modules have a write-only system data area to which your program can transfer data records. This area contains data records with numbers from 0 to a maximum of 240. Not every module contains all of the data records (see following table).

Other modules have a read-only system data area in which your program can read data records. This area contains data records with numbers from 0 to a maximum of 240. Not every module contains all of the data records (see following table).

Note

There are modules that have both system data areas. These are physically separate areas and all they have in common is their logical structure.

Write-Only System Data Area

The following table shows the structure of the write-only system data area. This table also shows how long the data records can be and with which SFCs the data records can be written.

Data Record Number	Contents	Size	Restriction	Can be Written with SFC
0	Parameters	With S7-300: from 2 to 14 bytes	Can only be written by an S7-400	56 "WR_DPARM" 57 "PARM_MOD"
1	Parameters	With S7-300: from 2 to 14 bytes Data records 0 and 1 together have a total of exactly 16 bytes.	-	55 "WR_PARM" 56 "WR_DPARM" 57 "PARM_MOD"
2 to 127	User data	Each = 240 bytes	-	55 "WR_PARM" 56 "WR_DPARM" 57 "PARM_MOD" 58 "WR_REC"
128 to 240	Parameters	Each = 240 bytes	-	55 "WR_PARM" 56 "WR_DPARM" 57 "PARM_MOD" 58 "WR_REC"

Read-only System Data Area

The following table shows the structure of the read-only system data area. This table also shows how long the data records can be and with which SFCs the data records can be read.

Data Record Number	Contents	Size	Can be Read with SFC
0	Module-specific diagnostic data (set as standard for the whole system)	4 bytes	51 "RDSYSST" (SSL_ID 00B1H) 59 "RD_REC"
1	Channel-specific diagnostic data (including data record 0)	• with S7-300: 16 bytes • with S7-400: 4 to 220 Bytes	51 "RDSYSST" (SSL_ID 00B2H and 00B3H) 59 "RD_REC"
2 to 127	User data	Each ≤ 240 bytes	59 "RD_REC"
128 to 240	Diagnostic data	Each ≤ 240 bytes	59 "RD_REC"

System Resources

If you start several asynchronous data record transfers one after the other with only short intervals between them, the allocation of system resources by the operating system ensures that all the jobs are executed and that they do not interfere with each other.

If all the available system resources are being used, this is indicated in RET_VAL. You can remedy this temporary error situation by simply repeating the job.

The maximum number of "simultaneously" active SFC jobs depends on the CPU. Refer to /**70**/ and /**101**/ for more detailed information.

7.2 Reading Defined Parameters with SFC 54 "RD_DPARM"

Description

With SFC 54 "RD_DPARM" (read defined parameter), you read the data record with the number RECNUM of the addressed module from the corresponding SDB1xy. The data record that is read is entered in the destination area opened by the parameter RECORD.

Parameter	Declaration	Data Type	Memory Area	Description
IOID	INPUT	BYTE	I, Q, M, D, L, constant	ID of the address area: B#16#54 =Peripheral input (PI) B#16#55 =Peripheral output (PQ) If the module is a mixed module, specify the area ID of the lowest address. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical base address of the module. With mixed modules, specify the lower of the two addresses.
RECNUM	INPUT	BYTE	I, Q, M, D, L, constant	Data record number (permitted values: 0 to 240)
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Length of the data record read in bytes if the read data record fits in the destination area and no error occurred in the transfer. If an error occurs while the function is active, the return value contains an error code.
RECORD	OUTPUT	ANY	I, Q, M, D, L	Destination area for the read data record. Only the data type BYTE is permitted.

Error Information

Assigning Parameters to a Module with SFC 57 "PARM_MOD"

7.3 Reading Predefined Parameters with SFC 102 "RD_DPARA"

Description

With SFC102 "RD_DPARA" you can read the data record with the number RECNUM of a selected module in the corresponding SDB1xy. The read data record is entered into the target area opened with the parameter RECORD.

Operating principle

The SFC 102 "RD_DPARA" operates asynchronously, that is, processing covers multiple SFC calls. Start the job by calling SFC102 with REQ = 1.

The job status is displayed via the output parameters RET_VAL and BUSY.

Refer also to Meaning of REQ, RET_VAL and BUSY with Asynchronously Operating SFCs

Parameters	Declaration	Data type	Memory area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	REQ = 1: Read request
LADDR	INPUT	WORD	I, Q, M, D, L, Const.	Logical start address of the module. With a mixed module group you must specify the lower address of two addresses.
RECNUM	INPUT	BYTE	I, Q, M, D, L, Const.	Data record number (permitted values: 0 to 240
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code. In addition: Length of the data record read in bytes if the read data record fits in the destination area and no error occurred in the transfer.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: The job is not yet closed.
RECORD	OUTPUT	ANY	I, Q, M, D, L	Target area for the fetched data record. Only the data type BYTE is permitted.

Error Information

See Configuring Modules with SFC 57 "PARM_MOD"

7.4 Writing Dynamic Parameters with SFC 55 "WR_PARM"

Description

With SFC 55 "WR_PARM" (write parameter), you transfer the data record RECORD to the addressed module. The parameters transferred to the module do not overwrite the parameters of this module in the corresponding SDB if they exist there.

Requirements

The data record to be transferred must not be static:

- It must not be data record 0 (data record 0 is static throughout the system).
- If the data record is referenced in SDBs 100 to 129, the static bit must not be set.

Refer to /**71**/ and /**101**/ for more information on static data records.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	REQ = 1: Write request
IOID	INPUT	BYTE	I, Q, M, D, L, constant	ID of the address area: B#16#54 =Peripheral input (PI) B#16#55 =Peripheral output (PQ) If the module is a mixed module, specify the area ID of the lowest address. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical base address of the module. With mixed modules, specify the lower of the two addresses.
RECNUM	INPUT	BYTE	I, Q, M, D, L, constant	Data record number
RECORD	INPUT	ANY	I, Q, M, D, L	Data record
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: Writing is not yet completed.

Input Parameter RECORD

The data to be transferred are read from the parameter RECORD during the first SFC call. If the transfer of the data record takes longer than the duration of a call, the contents of the parameter RECORD are no longer relevant for the subsequent SFC calls (for the same job).

Error Information

Assigning Parameters to a Module with SFC 57 "PARM_MOD"

Note
(S7-400 only)

If the general error W#16#8544 occurs, this only indicates that access to at least one byte of the I/O memory area containing the data record was denied. The data transfer was continued.

7.5 Writing Default Parameters with SFC 56 "WR_DPARM"

Description

With SFC 56 "WR_DPARM" (write default parameter), you transfer the data record with the number RECNUM from the corresponding SDB1xy to the addressed module. With this function, it is irrelevant whether the data record is static or dynamic.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	REQ = 1: Write request
IOID	INPUT	BYTE	I, Q, M, D, L, constant	ID of the address area: B#16#54 = Peripheral Input (PI) B#16#55 = Peripheral Output (PO) If the module is a mixed module, specify the area ID of the lowest address. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical base address of the module. With mixed modules, specify the lower of the two addresses.
RECNUM	INPUT	BYTE	I, Q, M, D, L, constant	Data record number
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: Writing is not yet completed.

Error Information

Assigning Parameters to a Module with SFC 57 "PARM_MOD"

7.6 Assigning Parameters to a Module with SFC 57 "PARM_MOD"

Description

With SFC 57 "PARM_MOD" (assign parameters to a module) you transfer all the data records of a module that you configured with STEP 7 in the corresponding SDB to the module. With this function, it is irrelevant whether the data records are static or dynamic.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	REQ = 1: Write request
IOID	INPUT	BYTE	I, Q, M, D, L, constant	ID of the address area: B#16#54 = Peripheral input (PI) B#16#55 = Peripheral output (PQ) If the module is a mixed module, specify the area ID of the lowest address. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical base address of the module. With mixed modules, specify the lower of the two addresses.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: Writing is not yet completed.

Error Information

The "real" error information (error codes W#16#8xyz) can be divided into two classes:

- Temporary errors (error codes W#16#80A2 to 80A4, 80Cx):

  With this type of error, it is possible that the error will be eliminated without you taking any action, in other words, it is advisable to call the SFC again (if necessary more than once).
  An example of a temporary error is when required resources are currently being used (W#16#80C3).

- Permanent errors (error codes W#16#809x, 80A1, 80Bx, 80Dx):

  This type of error will not be eliminated without you taking action. Calling the SFC again will only be successful after the error has been eliminated.
  An example of a permanent error is entering the wrong length in RECORD (W#16#80B1).

Note

If you transfer data records to a DPV1 Slave with SFCs 55, 56 or 57 and if this slave operates in DPV1 mode the DP master evaluates the error information it has received from this slave as follows:

If the error information lies within the range from W#16#8000 to W#16#80FF or W#16#F000 to W#16#FFFF the DP master passes the error information to the SFC. If it lies out of this range, the CPU passes the value W#16#80A2 to the SFC and suspends the slave.

Specific error information for SFC 54 "RD_DPARM," SFC 55 "WR_PARM," SFC 56 "WR_DPARM," and SFC 57 "PARM_MOD."

Error Code (W#16#...)	Explanation	Restriction
7000	First call with REQ=0: No data transfer active; BUSY has the value 0.	-
7001	First call with REQ=1: Data transfer started; BUSY has the value 1.	Distributed I/Os
7002	Interim call (REQ irrelevant): Data transfer active; BUSY has the value 1.	Distributed I/Os
8090	Specified logical base address invalid: There is no assignment in SDB1/SDB2x or there is no base address.	-
8092	The type specified in the ANY reference is not BYTE.	Only with S7-400 for SFC 54 "RD_PARM" and SFC 55 "WR_PARM"
8093	This SFC is not permitted for the module specified by LADDR and IOID (the following modules are permitted: S7-300 modules for an S7-300, S7-400 modules for an S7-400, S7-DP modules for an S7-300 and S7-400).	-
80A1	Negative acknowledgement when sending the data record to the module (the module was removed or became defective during transfer).	1)
80A2	DP protocol error at layer 2, possibly hardware/interface fault in DP slave	Distributed I/Os 1)
80A3	DP protocol error with user interface/user.	Distributed I/Os 1)
80A4	Communication problem on communication bus.	Error occurs between the CPU and external DP interface module 1)
80B0	SFC for module type not possible, module does not recognize the data record.	1)
80B1	The length of the transferred data record is incorrect. With SFC 54 "RD_PARM": the length of the destination area opened by RECORD is too short.	-
80B2	The configured slot is not occupied.	1)
80B3	Actual module type does not match the required module type in SDB1.	1)
80C1	The data of the previous write job for the same data record on	1)

Error Code (W#16#...)	Explanation	Restriction
	the module have not yet been processed by the module.	
80C2	The module is currently processing the maximum possible number of jobs for a CPU.	1)
80C3	The required resources (memory, etc.) are currently occupied.	1)
80C4	Internal temporary error. The job could not be processed. • Repeat the job. If this error occurs frequently check your system for electrical disturbance sources.	1)
80C5	Distributed I/Os not available.	Distributed I/Os 1)
80C6	Data record transfer was stopped due to a priority class abort (hot restart or background)	Distributed I/Os 1)
80D0	There is no entry for the module in the corresponding SDB.	-
80D1	The data record number is not configured in the corresponding SDB for the module (data record numbers = 241 are rejected by STEP 7).	-
80D2	The module cannot be assigned parameters according to its type identifier.	-
80D3	The SDB cannot be accessed since it does not exist.	-
80D4	SDB structure error: The SDB internal pointer points to a value outside the SDB.	only with S7-300
80D5	The data record is static.	only with SFC 55 "WR_PARM"

1) Does not occur in SFC 54 "RD_DPARM"

7.7 Writing a Data Record with SFC 58 "WR_REC"

Description

With SFC 58 "WR_REC" (write record), you transfer the data record contained in RECORD to the addressed module.
You start the write job by assigning the value 1 to the input parameter REQ when SFC 58 is called. If the write job could be executed immediately, the SFC returns the value 0 at the output parameter BUSY. If BUSY has the value 1, writing is not yet completed.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	REQ = 1: Write request
IOID	INPUT	BYTE	I, Q, M, D, L, constant	ID of the address area: B#16#54 = Peripheral input (PI) B#16#55 = Peripheral output (PQ) With mixed modules, specify the area ID of the lowest address. With the same addresses, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical address of the module. With mixed modules, specify the lower of the two addresses.
RECNUM	INPUT	BYTE	I, Q, M, D, L, constant	Data record number (permitted values: 2 to 240)
RECORD	INPUT	ANY	I, Q, M, D, L	Data record. Only the data type BYTE is permitted.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: Writing is not yet completed.

Input Parameter RECORD

The data to be transferred are read from the parameter RECORD during the first SFC call. If the transfer of the data record takes longer than the duration of a call, the contents of the parameter RECORD are no longer relevant for the subsequent SFC calls (for the same job).

Error Information

Reading a Data Record with SFC 59 "RD_REC"

Note

If the general error W#16#8544 occurs, this only indicates that access to at least one byte of the I/O memory area containing the data record was denied. The data transfer was continued.

7.8 Reading a Data Record with SFC 59 "RD_REC"

Description

With SFC 59 "RD_REC" (read record), you read the data record with the number RECNUM from the addressed module. You start the read job by calling SFC 59 and assigning the value 1 to the input parameter REQ. If the read job could be executed immediately, the SFC returns the value 0 in the BUSY output parameter. If BUSY has the value 1, the read job is not yet completed (see Section 0). The data record read is entered in the destination area indicated by the RECORD parameter providing the data transfer was free of errors.

Note

If you read a data record with a number higher than one from an FM or CP purchased before February 1997 (called an "older module" below), The reaction of SFC 59 is different from that with a new module. This special situation is described in the section "Using Older S7-300 FMs and CPs with Data Record Numbers Higher than 1."

Note

The following applies to S7-400 CPUs with a lower version than shown in the table below: If the destination area is smaller than the data record to be read, W#16#80B1 is entered in RET_VAL and the destination area remains unchanged. If the lengths of the destination area and the data record to be read are identical, instead of the value 0 (no error) the length of the data record is entered as a positive value in RET_VAL.

CPU	Order Number	Version (or Higher)
CPU 412-1	6ES7412-1XF01-0AB0	03
CPU 413-1	6ES7413-1XG01-0AB0	03
CPU 413-2DP	6ES7413-2XG01-0AB0	03
CPU 414-1	6ES7414-1XG01-0AB0	03
CPU 414-2DP	6ES7414-2XG01-0AB0	03
CPU 414-2DP	6ES7414-2XJ00-0AB0	03
CPU 416-1	6ES7416-1XJ01-0AB0	03
CPU 416-2DP	6ES7416-2XK00-0AB0	03
CPU 416-2DP	6ES7416-2XL00-0AB0	03

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	REQ = 1: Read request
IOID	INPUT	BYTE	I, Q, M, D, L, constant	ID of the address area: B#16#54 = Peripheral input (PI) B#16#55 = Peripheral output (PQ) If the module is a mixed module, specify the area ID of the lowest address. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical address of the module. With mixed modules, specify the lower of the two addresses.
RECNUM	INPUT	BYTE	I, Q, M, D, L, constant	Data record number (permitted values 0 to 240)
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code. The length of the data record actually transferred in bytes (possible values: +1 to +240) is also entered if the destination area is larger than the transferred data record and if no error occurred in the transfer.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: Reading is not yet completed.
RECORD	OUTPUT	ANY	I, Q, M, D, L	Destination area for the data record read. With asynchronous execution of SFC 59, make sure that the actual parameters of RECORD have the same length information for all calls. Only data type BYTE is permitted.

Output Parameter RET_VAL

- If an error occurred while the function was being executed, the return value contains an error code.
- If no error occurred, RET_VAL contains the following:
  - 0: if the entire destination area was filled with data from the selected data record (the data record can also be incomplete).
  - The length of the data record actually transferred in bytes (possible values: +1 to + 240) if the destination area is larger than the transferred data record.

Note

If the general error W#16#8545 occurs, this only indicates that access to at least one byte of the I/O memory area containing the data record was blocked. The data record was read by the module correctly and written to the I/O memory area.

Setting RECORD

Note

If you want to ensure that the entire data record is always read, select a destination area with a length of 241 bytes. If the data transfer is error-free, RET_VAL contains the actual data record length.

Using Older S7-300 FMs and CPs with Data Record Numbers Higher than 1

If you want to read out a data record with a number higher than 1 from an older S7-300 FM or older S7-300 CP using SFC 59 "RD_REC," remember the following points:

- If the destination area is larger than the actual length of the required data record, no data are entered in RECORD.
  RET_VAL has the value W#16#80B1.
- If the destination area is smaller than the actual length of the required data record, the CPU reads as many bytes beginning at the start of the record as are specified in the length information of RECORD and enters this number of bytes in RECORD.
  RET_VAL has the value 0.
- If the length specified in RECORD is the same as the actual length of the required data record, the CPU reads the data record and enters it in RECORD.
  RET_VAL has the value 0.

Error Information

The "real" error information (error codes W#16#8xyz) in the following table can be divided into two classes:

- Temporary errors (error codes W#16#80A2 to 80A4, 80Cx):
  With this type of error, it is possible that the error will be eliminated without you taking any action, in other words, it is advisable to call the SFC again (if necessary, more than once).
  An example of a temporary error is when required resources are currently being used (W#16#80C3).
- Permanent errors (error codes W#16#809x, 80A1, 80Bx, 80Dx):
  This type of error will not be eliminated without you taking action. Calling the SFC again will only be successful after the error has been eliminated. An example of a permanent error is entering the wrong length in RECORD (W#16#80B1).

Note

If you transfer data records to a DPV1 slave with SFC58 "WR_REC" or if you fetch data records from a DPV1 slave with SFC59 "RD_REC" and if this DPV1 slave operates in DPV1 mode, the DP Master evaluates the error information it received from the Slave as follows:

If the error information lies within the range from W#16#8000 to W#16#80FF or W#16#F000 to W#16#FFFF the DP master passes the error information to the SFC. If it lies out of this range, the CPU passes the value W#16#80A2 to the SFC and suspends the slave.

For a description of the error information received from DPV1-Slaves, see Receiving an Interrupt from a DP-Slave with SFB 54 "RALRM" STATUS[3].

Specific error information for SFC 58 "WR_REC" and SFC 59 "RD_REC."

Error Code (W#16#...)	Explanation	Restriction
7000	First call with REQ=0: No data transfer active; BUSY has the value 0.	-
7001	First call with REQ=1: No data transfer active; BUSY has the value 1.	Distributed I/Os
7002	Interim call (REQ irrelevant): Data transfer already active; BUSY has the value 1.	Distributed I/Os
8090	Specified logical base address invalid: There is no assignment in SDB1/SDB2x or there is no base address.	-
8092	The type specified in the ANY reference is not BYTE.	S7-400 only
8093	This SFC is not permitted for the module specified by LADDR and IOID (the following modules are permitted: S7-300 modules for an S7-300, S7-400 modules for an S7-400, S7-DP modules for an S7-300 and S7-400).	-
80A0	Negative acknowledgement when reading from the module • the module was removed during the read job or is defective • additionally with H-Systems: unilateral I/O of the standby CPU not available (for example, standby CPU in STOP)	SFC 59 "RD_REC" only
80A1	Negative acknowledgement when sending the data record to the module • the module was removed during transfer or is defective • additionally with H-Systems: unilateral I/O of the standby CPU not available (for example, standby CPU in STOP)	SFC 58 "WR_REC" only
80A2	DP protocol error at layer 2	Distributed I/Os
80A3	DP protocol error with user interface/user	Distributed I/Os
80A4	Communication problem on the communication bus	The error occurs between the CPU and the external DP interface module.

Error Code (W#16#...)	Explanation	Restriction
80B0	SFC not possible for module type. The module does not recognize the data record. Data record number w 241 not permitted. With SFC 58 (WR_REC), data records 0 and 1 are not permitted.	-
80B1	The length specified in the RECORD parameter is incorrect.	• SFC 58 "WR_REC": Length incorrect • SFC 59 "RD_REC" (only when using older S7-300 FMs and S7-300 CPs): specified length > record length • With SFC 13 "DPNRM_DG": specified length > record length
80B2	The configured slot is not occupied.	-
80B3	Actual module type does not match the required module type in SDB1	-
80C0	For SFC 59 (RD_REC): The module has the data record, but there are still no data to be read. For SFC 13 (DPNRM_DG): There are no diagnostic data available.	Only for SFC 59 "RD_REC" or SFC 13 "DPNRM_DG"
80C1	The data of the previous write job for the same data record on the module have not yet been processed by the module.	-
80C2	The module is currently processing the maximum possible number of jobs for a CPU.	-
80C3	The required resources (memory, etc.) are currently occupied.	-
80C4	Internal temporary error. The job could not be processed. • Repeat the job. If this error occurs frequently check your system for electrical disturbance sources.	-
80C5	Distributed I/Os not available.	Distributed I/Os
80C6	Data record transfer was stopped due to a priority class abort (restart or background)	Distributed I/Os

7.9 Reading a Data Record with SFC 59 "RD_REC" on S7-300 CPUs

Applicability

The following description of SFC 59 "RD_REC" applies to the CPUs listed below:

CPU	Order Number
CPU 312 IFM	6ES7312-5AC00-0AB0
CPU 313	6ES7313-1AD00-0AB0
CPU 314	6ES7314-1AE01-0AB0
CPU 314 IFM	6ES7314-5AE00-0AB0
CPU 315	6ES7315-1AF00-0AB0
CPU 315-2DP	6ES7315-2AF00-0AB0
CPU 614	6ES7614-1AH00-0AB3

Description

With SFC 59 "RD_REC" (read data record), you read the data record with the number RECNUM from the addressed module. The data record read is entered in the destination area indicated by the RECORD parameter providing the data transfer was free of errors.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	REQ = 1: Read request
IOID	INPUT	BYTE	I, Q, M, D, L, constant	ID of the address area: B#16#54 = Peripheral input (PI) B#16#55 = Peripheral output (PQ) If the module is a mixed module, specify the area ID of the lowest address. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical address of the module. With mixed modules, specify the lower of the two addresses.
RECNUM	INPUT	BYTE	I, Q, M, D, L, constant	Data record number (permitted values 0 to 240)
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains an error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: Reading is not yet completed.
RECORD	OUTPUT	ANY	I, Q, M, D, L	Destination area for the data record read. With asynchronous execution of SFC 59, make sure that the actual parameters of RECORD have the same length information for all calls. Only data type BYTE is permitted.

RECORD

The length information in the output parameter RECORD is interpreted as follows:

Length of the data to be read from the selected data record. This means that the length information of RECORD must not be longer than the actual data record length.

It is advisable to select the length for RECORD exactly the same as the actual data record length.

Principle of Data Transfer

With the read job, the CPU informs the addressed module of the length of the RECORD parameter. The following points depend on whether or not the module belongs to a DP station:

- The module is in a central or expansion rack.
- If the length specified by RECORD is shorter than the actual length of the required data record, the CPU reads as many bytes from the start of the data record as specified in the length information of RECORD and enters them in RECORD. RET_VAL has the value 0.
- If the length in RECORD is longer than the actual length of the required data record, the CPU enters an error code in RET_VAL.
- If the length information in RECORD is the same as the actual length of the required data record, the CPU reads the required data record and enters it in RECORD. The value 0 is entered in RET_VAL.
- The module is located in a DP S7 slave.
- The communications processor of the DP S7 slave evaluates the length information received from the CPU.
- If the length in RECORD is less than the length of the required data record, a DP S7-300 slave returns the required part of the selected data record to the CPU. If the length in RECORD is longer than the length of the required data record, a DP S7-300 slave returns error information to the CPU. The CPU evaluates the error or length information received from the DP S7 slave:
- If the DP S7 slave provides error information, the corresponding error code is entered in RET_VAL.
- If the DP S7 slave returns the length of the data read out, this length is compared with the length information in RECORD. Depending on the result of the comparison, an entry is made in the output parameters RET_VAL and RECORD (The response is the same as when the module is located in a central or expansion rack.)

Note

With asynchronous processing of SFC 59, make sure that the actual parameters of RECORD have the same length information in all calls.

Error Information

Error Code (W#16#...)	Explanation	Restriction
7000	First call with REQ=0: No data transfer active; BUSY has the value 0.	-
7001	First call with REQ=1: No data transfer active; BUSY has the value 1.	Distributed I/Os
7002	Interim call (REQ irrelevant): Data transfer already active; BUSY has the value 1.	Distributed I/Os
8090	Specified logical base address invalid: There is no assignment in SDB1/SDB2x or there is no base address.	-
8093	This SFC is not permitted for the module specified by LADDR and IOID (the following modules are permitted: S7-300 modules and S7-300 DP modules).	-
80A0	Negative acknowledgement when reading from the module (the module was removed during the read job or is defective).	-
80A2	DP protocol error at layer 2	Distributed I/Os
80A3	DP protocol error with user interface/user	Distributed I/Os
80A4	Communication problem on the communication bus	The error occurs between the CPU and the external DP interface module.
80B0	SFC not possible for module type. The module does not recognize the data record. Data record number w 241 not permitted.	-
80B1	The length specified in the RECORD parameter is incorrect.	Specified length > record length
80B2	The configured slot is not occupied.	-
80B3	Actual module type does not match the required module type in SDB1.	-
80C0	The module has the data record, but there are still no data to be read.	
80C2	The module is currently processing the maximum possible number of jobs for a CPU.	-
80C3	The required resources (memory, etc.) are currently occupied.	-
80C4	Internal temporary error. The job could not be processed. • Repeat the job. If this error occurs frequently check your system for electrical disturbance sources.	-
80C5	Distributed I/Os not available.	Distributed I/Os
80C6	Data record transfer was stopped due to a priority class abort (hot restart or background)	Distributed I/Os

7.10 Further Error Information for SFCs 55 to 59

S7-400 Only

With the S7-400, the SFCs 55 to 59 can also return the error information W#16#80Fx. In this case an error occurred that could not be localized. Please contact the maintenance department in this case.

8 DPV1 SFBs According to PNO AK 1131

8.1 Reading a Data Record from a DP Slave with SFB 52 "RDREC"

Note

The SFB52 "RDREC" interface is identical to the FB "RDREC" defined in the standard "PROFIBUS Guideline PROFIBUS Communication and Proxy Function Blocks according to IEC 61131-3".

Description

With the SFB52 "RDREC" (read record) you read a data record with the number INDEX from a DP Slave component (module or modules) that has been addressed via ID.

Specify the maximum number of bytes you want to read in MLEN. The selected length of the target area RECORD should have at least the length of MLEN bytes.

TRUE on output parameter VALID verifies that the data record has been successfully transferred into the target area RECORD. In this case, the output parameter LEN contains the length of the fetched data in bytes.

The output parameter ERROR indicates whether a data record transmission error has occurred. In this case, the output parameter STATUS contains the error information.

Operating principle

The SFC 52 "RDREC" operates asynchronously, that is, processing covers multiple SFC calls. Start the job by calling SFC52 with REQ = 1.

The job status is displayed via the output parameter BUSY and bytes 2 and 3 of output parameter STATUS. Here, the STATUS bytes 2 and 3 correspond with the output parameter RET_VAL of the asynchronously operating SFCs (see also Meaning of REQ, RET_VAL and BUSY with Asynchronously Operating SFCs).

Data record transmission is completed when the output parameter BUSY = FALSE.

Parameters	Declaration	Data type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, Const.	REQ = 1: Transfer data record
ID	INPUT	DWORD	I, Q, M, D, L, Const.	logical address of the DP Slave component (module)

Parameters	Declaration	Data type	Memory Area	Description
INDEX	INPUT	INT	I, Q, M, D, L, Const.	Data record number.
MLEN	INPUT	INT	I, Q, M, D, L, Const.	maximum length in bytes of the data record information to be fetched
VALID	OUTPUT	BOOL	I, Q, M, D, L	New data record was received and valid
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: The read process is not yet terminated.
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR = 1: A read error has occurred.
STATUS	OUTPUT	DWORD	I, Q, M, D, L	Call ID (bytes 2 and 3) or error code
LEN	OUTPUT	INT	I, Q, M, D, L	Length of the fetched data record information
RECORD	IN_OUT	ANY	I, Q, M, D, L	Target area for the fetched data record.

Error Information

See Receiving an Interrupt from a DP Slave with SFB54 "RALRM"

8.2 Writing a Data Record in a DP Slave with SFB53 "WRREC"

Note

The SFB52 "WRREC" interface is identical to the FB "WRREC" defined in the standard "PROFIBUS Guideline PROFIBUS Communication and Proxy Function Blocks according to IEC 61131-3"

Description

With the SFB52 "WRREC" (write record) you transfer a data record with the number INDEX to a DP Slave component (module) that has been addressed via ID.

Specify the byte length of the data record to be transmitted. The selected length of the source area RECORD should, therefore, have at least the length of LEN bytes.

TRUE on output parameter DONE verifies that the data record has been successfully transferred to the DP Slave.

The output parameter ERROR indicates whether a data record transmission error has occurred. In this case, the output parameter STATUS contains the error information.

Operating principle

The SFC 53 "WRREC" operates asynchronously, that is, processing covers multiple SFC calls. Start the job by calling SFC53 with REQ = 1.

The job status is displayed via the output parameter BUSY and bytes 2 and 3 of output parameter STATUS. Here, the STATUS bytes 2 and 3 correspond with the output parameter RET_VAL of the asynchronously operating SFCs (see also Meaning of REQ, RET_VAL and BUSY with Asynchronously Operating SFCs).

Please note that you must assign the same value to the actual parameter of RECORD for all SFB53 calls that belong to one and the same job. The same applies to the LEN parameters.

Data record transmission is completed when the output parameter BUSY = FALSE.

Parameters	Declaration	Data type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, Const.	REQ = 1: Transfer data record
ID	INPUT	DWORD	I, Q, M, D, L, Const.	logical address of the DP Slave component (module)
INDEX	INPUT	INT	I, Q, M, D, L, Const.	Data record number.
LEN	INPUT	INT	I, Q, M, D, L, Const.	maximum byte length of the data record to be transferred
DONE	OUTPUT	BOOL	I, Q, M, D, L	Data record was transferred
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: The write process is not yet terminated.
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR = 1: A write error has occurred.
STATUS	OUTPUT	DWORD	I, Q, M, D, L	Call ID (Bytes 2 and 3) or error code
RECORD	IN_OUT	ANY	I, Q, M, D, L	Data record

Error Information

See Receiving an Interrupt from a DP Slave with SFB54 "RALRM"

8.3 Receiving an Interrupt from a DP Slave with SFB 54 "RALRM"

Note

The SFB52 "RALRM" interface is identical to the FB "RALRM" defined in the standard "PROFIBUS Guideline PROFIBUS Communication and Proxy Function Blocks according to IEC 61131-3"

Description

The SFB "RALRM" receives an interrupt with all corresponding information from a peripheral module (centralized structure) or from a DP Slave component. It supplies this information to its output parameters.

The information in the output parameters contains the start information of the called OB as well as information of the interrupt source.

Call the SFB54 only within the interrupt OB started by the CPU operating system as a result of the peripheral interrupt that is to be examined.

Note

If you call SFB54 "RALRM" in an OB for which the start event was not triggered by peripherals, the SFB supplies correspondingly reduced information on its outputs.
Make sure to use different instance DBs when you call SFB 54 "RALRM" in different OBs. If you want to evaluate data that are the result of an SFB54 call outside of the associated interrupt OB you should moreover use a separate instance DB per OB start event.

Call of the SFB 54

You can call the SFB54 "RALRM" in three operating modes (MODE). They are explained in the table below.

MODE	The SFB54 ...
0	... shows the component that triggered the interrupt in the output parameter ID and sets the output parameter NEW to TRUE.
1	... describes all output parameters, independent on the interrupt triggering component.
2	... checks whether the component specified in input parameter F_ID has triggered the interrupt. • If not, NEW = FALSE • If yes, NEW = TRUE, and all other outputs parameters are described

Parameters	Declaration	Data type	Memory Area	Description
MODE	INPUT	INT	I, Q, M, D, L, Const.	Operating mode
F_ID	INPUT	DWORD	I, Q, M, D, L, Const.	logical start address of the component (module) from which interrupts are to be received
MLEN	INPUT	INT	I, Q, M, D, L, Const.	maximum length in bytes of the data interrupt information to be received
NEW	OUTPUT	BOOL	I, Q, M, D, L	A new interrupt was received.
STATUS	OUTPUT	DWORD	I, Q, M, D, L	Error code of the SFB or DP Master
ID	OUTPUT	DWORD	I, Q, M, D, L	Logical start address of the component (module) from which an interrupt was received. The highest value bit contains the I/Q ID: 0 for an input address; 1 for and output address
LEN	OUTPUT	INT	I, Q, M, D, L	Length of the received interrupt information
TINFO	IN_OUT	ANY	I, Q, M, D, L	(task information) Target range for OB start and management information
AINFO	IN_OUT	ANY	I, Q, M, D, L	(interrupt information) Target area for header information and additional interrupt information For AINFO you should provide a length of at least MLEN bytes.

Caution

If you select a target area TINFO or AINFO that is too short the SFB54 cannot enter the full information.

Data structure of the target area TINFO

Byte	Meaning
0 to 19	Start information of the OB in which SFB54 was currently called
20 to 27	Management information

Structure of the Management Information

Byte no. for TINFO	Data type	Meaning				
20	BYTE	central:	0			
		distributed:	DP master system ID (possible values) 1 to 255			
21	BYTE	central:	Module rack number (possible values: 0 to 31			
		distributed:	Number of the DP station (possible values: 0 to 127			
22	BYTE	central:	0			
		distributed:	• Bit 0 to 3:	Slave type	0000: 0001: 0010: 0011: as of 0100:	DP DPS7 DPS7 V1 DPV1 reserved
			• Bit 4 to 7:	Profile type	0000: as of 0001:	DP reserved
23	BYTE	central:	0			
		distributed:	• Bit 0 to 3:	Interrupt info type	0000:	Transparent (Interrupt originates from a configured distributed module)
					0001:	Representative (Interrupt originating from a non-DPV1 Slave or a slot that is not configured)
					0010:	Generated (interrupt generated in the CPU)
					as of 0011:	reserved
			• Bit 4 to 7:	Structure version	0000: as of 0001:	Initial reserved
24	BYTE	central:	0			
		distributed:	Flags of the DP Master interface			
			• Bit 0 = 0:	Interrupt originating from an integrated DP interface		
			• Bit 0 = 1:	Interrupt originating from an external DP interface		
			• Bit 1 to 7:	reserved		

Byte no. for TINFO	Data type	Meaning		
25	BYTE	central:	0	
		distributed:	Flags of the DP Slave interface	
			• bit 0:	EXT_DIAG_Bit of the diagnostic message frame, or 0 if this bit does not exist in the interrupt
			• Bit 1 to 7:	reserved
26 to 27	WORD	central:	0	
		distributed:	PROFIBUS ID number	

Data structure of the target area AINFO

Byte	Meaning	
0 to 3	Header information	
4 to 223	Additional interrupt information: module specific data for the respective interrupt:	
	• central:	ARRAY[0] to ARRAY[220]
	• distributed:	ARRAY[0] to ARRAY[59]

Structure of the Header Information

Byte	Data type	Meaning		
0	BYTE	Length of the received interrupt information in bytes		
		• central: • distributed:	1 to 224 4 to 63	
1	BYTE	central:	reserved	
		distributed:	ID for the interrupt type	
			1:	Diagnostic interrupt
			2:	Hardware interrupt
			3:	Removal interrupt
			4:	Insertion interrupt
			5:	Status interrupt
			6:	Update Interrupt
			31:	failure of an expansion device, DP Master system or DP station
			32 to 126	manufacturer specific interrupt
2	BYTE	Slot number of the interrupt triggering component		

Byte	Data type	Meaning		
3	BYTE	central:	reserved	
		distributed:	Specifier	
			Bits 0 and 1	0: no further information 1: upcoming event, disrupted slot 2: outgoing event, slot not disrupted anymore 3: outgoing event, slot still disrupted
			Bit 2:	Add_Ack
			Bits 3 to 7:	Sequence number

Target Area TINFO and AINFO

Dependent on the respective OB in whichSFB54 is called, the target areas TINFO and AINFO are only partially written. Refer to the table below for information on which info is entered respectively.

Interrupt type	OB	TINFO OB status information	TINFO management information	AINFO header information	AINFO additional interrupt information	
Hardware interrupt	4x	Yes	Yes	Yes	central:	No
					distributed:	as delivered by the DP Slave
Status interrupt	55	Yes	Yes	Yes	Yes	
Update Interrupt	56	Yes	Yes	Yes	Yes	
manufacturer specific interrupt	57	Yes	Yes	Yes	Yes	
Peripheral redundancy error	70	Yes	Yes	No	No	
Diagnostic interrupt	82	Yes	Yes	Yes	central:	Data record 1
					distributed:	as delivered by the DP Slave
Removal/ Insertion interrupt	83	Yes	Yes	Yes	central:	No
					distributed:	as delivered by the DP Slave
Module rack/ Station failure	86	Yes	Yes	No	No	
...	all other OBs	Yes	No	No	No	

Error Information

The output parameter STATUS contains information. It is interpreted as ARRAY[1...4] OF BYTE the error information has the following structure:

Field element	Name:	Meaning
STATUS[1]	Function_Num	• B#16#00, if no error • Function ID from DPV1-PDU: In error case B#16#80 is OR linked. If no DPV1 protocol element is used: B#16#C0.
STATUS[2]	Error_Decode	Location of the error ID
STATUS[3]	Error_Code_1	Error ID
STATUS[4]	Error_Code_2	manufacturer specific error ID expansion

STATUS[2] can have the following values:

Error_Decode (B#16#...)	Source	Meaning
00 to 7F	CPU	no error or no warning
80	DPV1	Error according to IEC 61158-6
81 to 8F	CPU	B#16#8x shows an error in the nth call parameter of the SFB.
FE, FF	DP Profile	profile specific error

STATUS[3] can have the following values:

Error_Decode (B#16#...)	Error_Code_1 (B#16#...)	Explanation according to DVP1	Meaning
00	00		no error, no warning
70	00	reserved, reject	initial call; no active data record transfer
	01	reserved, reject	initial call; data record transfer has started
	02	reserved, reject	intermediate call; data record transfer already active
80	90	reserved, pass	invalid logical start address
	92	reserved, pass	illegal type for ANY pointer
	93	reserved, pass	The DP component addressed via ID or F_ID is not configured.

Error_Decode (B#16#...)	Error_Code_1 (B#16#...)	Explanation according to DVP1	Meaning
	96		A master-reserve switchover has occurred in an H system, and the previous master CPU has gone into STOP mode. At that time, an OB was being processed. SFB 54 cannot supply the OB start information, management information, header infromation or additional interrupt infromation. You can read out the OB start information with SFC 6 "RD_SINFO". In addition, you can use SFC 13 "DPNRM_DG" to synchronously read the current diagnostic frame of the affected DP slave for OBs 4x, 55, 56, 57, 82 and 83 (Address information from the OB start information).
	A0	read error	Negative acknowledgement while reading the module.
	A1	write error	negative acknowledgement while writing the module
	A2	module failure	DP protocol error at layer 2, possible hardware failure
	A3	reserved, pass	DP protocol error with Direct-Data-Link-Mapper or User-Interface/User, possibly hardware failure
	A4	reserved, pass	Bus communication disrupted
	A5	reserved, pass	–
	A7	reserved, pass	DP slave or modules is occupied (temporary error
	A8	version conflict	DP slave or modules reports non-compatible versions
	A9	feature not supported	Feature not supported by DP slave or module
	AA to AF	user specific	DP slave or module reports a manufacturer-specific error in its application. Please check the documentation from the manufacturer of the DP slave or module.
	B0	invalid index	Data record not known in module Illegal data record number ≥256
	B1	write length error	Wrong length specified in parameter RECORD; with SFB54: length error in AINFO
	B2	invalid slot	Configured slot not occupied.
	B3	type conflict	Actual module type not equal to specified module type
	B4	invalid area	DP slave or module reports access to an invalid area
	B5	status conflict	DP slave or module not ready
	B6	access denied	DP slave or module denies access
	B7	invalid range	DP slave or module reports an invalid range for a parameter or value
	B8	invalid parameter	DP slave or module reports an invalid parameter
	B9	invalid type	DP slave or module reports an invalid type
	BA to BF	user specific	DP slave or module reports a manufacturer-specific error when accessing. Please check the documentation from the manufacturer of the DP slave

Error_Decode (B#16#...)	Error_Code_1 (B#16#...)	Explanation according to DVP1	Meaning
			or module.
	C0	read constrain conflict	The module has the data record, however, there are no read data yet.
	C1	write constrain conflict	The data of the previous write request to the module for the same data record have not yet been processed by the module.
	C2	resource busy	The module currently processes the maximum possible jobs for a CPU.
	C3	resource unavailable	The required operating resources are currently occupied.
	C4		Internal temporary error. Job could not be carried out. Repeat the job. If this error occurs often, check your plant for sources of electrical interference.
	C5		DP slave or module not available.
	C6		Data record transfer was canceled due to priority class cancellation
	C7		Job canceled due to restart (warm restart) or cold restart of DP master
	C8 bis CF		DP slave or module reports a manufacturer-specific resource error. Please check the documentation from the manufacturer of the DP slave or module.
	Dx	user specific	DP Slave specific. Refer to the description of the DP Slave.
81	00 to FF		Error in the initial call parameter (with SFB54: MODE)
	00		Illegal operating mode
82	00 to FF		Error in the second call parameter
:	:		:
88	00 to FF		Error in the eighth call parameter (with SFB54: TINFO)
	01		Wrong syntax ID
	23		Quantity frame exceeded or target area too small
	24		Wrong range ID
	32		DB/DI no. out of user range
	3A		DB/DI no. is NULL for area ID DB/DI or specified DB/DI does not exist
89	00 to FF		Error in the ninth call parameter (with SFB54: AINFO)
	01		Wrong syntax ID
	23		Quantity frame exceeded or target area too small
	24		Wrong range ID
	32		DB/DI no. out of user range

Error_Decode (B#16#...)	Error_Code_1 (B#16#...)	Explanation according to DVP1	Meaning
	3A		DB/DI no. is NULL for area ID DB/DI or specified DB/DI does not exist
8A	00 to FF		Error in the 10th call parameter
:	:		:
8F	00 to FF		Error in the 15th call parameter
FE, FF	00 to FF		Profile-specific error

With DPV1 errors, the DP Master passes on STATUS[4] to the CPU and to the SFB. Without DPV1 error, this value is set to 0, with the following exceptions for the SFB52:

- STATUS[4] contains the target area length from RECORD, if MLEN > the target area length from RECORD
- STATUS[4]=MLEN, if the actual data record length < MLEN < the target area length from RECORD

9 SFCs for Handling Time-of-Day Interrupts

9.1 Handling Time-of-Day Interrupts

Definition

A time-of-day interrupt results in one of the time-of-day interrupt OBs (OB10 to OB17) being called.

Conditions for the Call

Before a time-of-day interrupt OB can be called by the operating system, the following conditions must be met:

- The time-of-day interrupt OB must have parameters assigned to it (start date and time, execution) using either
  - STEP 7 or
  - SFC 28 "SET_TINT" in the user program.
- The time-of-day interrupt OB must be activated using
  - STEP 7 or
  - SFC 30 "ACT_TINT" in the user program.
- The time-of-day interrupt OB must not be deselected with STEP 7.
- The time-of-day interrupt OB must exist in the CPU.
- If you set the interrupt with SFC 30 "ACT_TINT" and if you have specified the execution of the OB as **once only**, the start date and time must not yet have passed. If you have selected **periodic** execution, the time-of-day interrupt OB will be called when the next period is completed (start time + multiple of the specified period).

Hint

You can assign parameters to the time-of-day interrupt using STEP 7 and then activate the interrupt in your user program (SFC 30 "ACT_TINT").

Purpose of SFC 28 to SFC 31

The system functions SFC 28 to SFC 31 described in the following sections are used as follows:

- To set time-of-day interrupts (SFC 28 "SET_TINT")
- To cancel time-of-day interrupts (SFC 29 "CAN_TINT")
- To activate time-of-day interrupts (SFC 30 "ACT_TINT")
- To query time-of-day interrupts (SFC 31 "QRY_TINT")

9.2 Characteristics of SFCs 28 to 31

What Happens If...

The following table lists a number of different situations and explains the effect they have on a time-of-day interrupt.

If ...	Then ...
A time-of-day interrupt is set (by calling SFC 28; SET_TINT)	The current time-of-day interrupt is canceled.
The time-of-day interrupt is canceled (by calling SFC 29; CAN_TINT)	The start date and time are cleared. The time-of-day interrupt must then be set again before it can be activated.
The time-of-day interrupt OB does not exist when it is called.	The priority class error is generated automatically, which means that the operating system calls OB85. If OB85 does not exist, the CPU changes to STOP.
The real-time clock is synchronized or the	
• clock adjusted forward	If the start date/time is skipped because the clock is moved forward: • The operating system calls OB80[1]. • Following OB80, every skipped time-of-day interrupt OB is called (once, regardless of the number of periods that were skipped) provided that it was not manipulated in OB80[2]. If OB80 does not exist, the CPU changes to STOP.
• clock adjusted back	If the time-of-day interrupt OBs had already been called during the time by which the clock has been moved back, they are not called again the second time around.

1) OB80 contains encoded start event information, indicating which time-of-day interrupt OBs could not be called due to moving the clock forward. The time in the start event information corresponds to the time adjusted forward.
2) The time in the start event information of the time-of-day interrupt activated later after being skipped corresponds to the start time of the first skipped time-of-day interrupt.

Warm Restart or Cold Restart

During a warm restart or a cold restart, all the time-of-day interrupt settings made in the user program by SFCs are cleared.

The parameters of the "time-of-day interrupts" parameter field set using STEP 7 are then effective.

Executing the Time-of-Day Interrupt OBs

The following table shows the different effects of the "execution" parameter. You set this parameter with STEP 7 or with SFC 28 "SET_TINT" (input parameter PERIOD).

Execution of the Time-of-Day Interrupt OBs	Reaction
None (can only be set with STEP 7)	The time-of-day interrupt OB is not executed even when it exists in the CPU. Parameters can be re-assigned in the user program using SFC 28 "SET_TINT" (set time-of-day interrupt).
Once	The time-of-day interrupt is canceled after the time-of-day interrupt OB has been called. It can then be set and activated again.
Periodic (every minute, hour, day, week, month, year)	If the start date and time have already passed when the interrupt is activated, the time-of-day interrupt OB interrupts the cyclic program at the next possible point "start date/time + multiple of the selected period." In extremely rare situations, processing of the time-of-day interrupt OB may not yet be completed when it is called again. Result: • Time error, (the operating system calls OB80; if OB80 does not exist, the CPU changes to STOP). • The time-of-day interrupt OB is executed later.

9.3 Setting a Time-of-Day Interrupt with SFC 28 "SET_TINT"

Description

With SFC 28 "SET_TINT" (set time-of-day interrupt), you set the start date and time of time-of-day interrupt organization blocks. The seconds and milliseconds of the specified start time are ignored and set to 0.

Parameter	Declaration	Data Type	Memory Area	Description
OB_NR	INPUT	INT	I, Q, M, D, L, constant	Number of the OB started at the time SDT + multiple of PERIOD (OB10 to OB17).
SDT	INPUT	DT	D, L, constant	Start date and time: The seconds and milliseconds of the specified start time are ignored and set to 0.
PERIOD	INPUT	WORD	I, Q, M, D, L, constant	Periods from start point SDT onwards: W#16#0000 = once W#16#0201 = every minute W#16#0401 = hourly W#16#1001 = daily W#16#1202 = weekly W#16#1401 = monthly W#16#1801 = yearly W#16#2001 = at month's end
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the actual parameter of RET_VAL contains an error code.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred
8090	Incorrect parameter OB_NR
8091	Incorrect parameter SDT
8092	Incorrect parameter PERIOD
80A1	The set start time is in the past.

9.4 Canceling a Time-of-Day Interrupt with SFC 29 "CAN_TINT"

Description

With SFC 29 "CAN_TINT" (cancel time-of-day interrupt), you cancel an activated time-of-day organization block

Parameter	Declaration	Data Type	Memory Area	Description
OB_NR	INPUT	INT	I, Q, M, D, L, constant	Number of the OB, in which the start date and time will be canceled (OB10 to OB17).
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the actual parameter of RET_VAL contains an error code.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	Incorrect parameter OB_NR
80A0	No start date/time specified for the time-of-day interrupt OB

9.5 Activating a Time-of-Day Interrupt with SFC 30 "ACT_TINT"

Description

With SFC 30 "ACT_TINT" (activate time-of-day interrupt), you can activate a time-of-day interrupt organization block.

Parameter	Declaration	Data Type	Memory Area	Description
OB_NR	INPUT	INT	I, Q, M, D, L, constant	Number of the OB to be activated (OB10 to OB17).
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the actual parameter of RET_VAL contains an error code.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	Incorrect parameter OB_NR.
80A0	Start date/time-of day not set for the respective time-of-day interrupt OB.
80A1	The activated time is in the past. This error only occurs if execution = once is selected.

9.6 Querying a Time-of-Day Interrupt with SFC 31 "QRY_TINT"

Description

Using the system function SFC 31 "QRY_TINT" (query time-of-day interrupt), you can display the status of a time-of-day interrupt organization block at the output parameter STATUS.

Parameter	Declaration	Data Type	Memory Area	Description
OB_NR	INPUT	INT	I, Q, M, D, L, constant	Number of the OB, whose status will be queried (OB10 to OB17).
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the actual parameter of RET_VAL contains an error code.
STATUS	OUTPUT	WORD	I, Q, M, D, L	Status of the time-of-day interrupt; see following table.

Output Parameter STATUS

Bit	Value	Meaning
0	0	Time-of-day interrupt is enabled by operating system.
1	0	New time-of-day interrupts are accepted.
2	0	Time-of-day interrupt is not activated or has elapsed.
3	-	-
4	0	Time-of-day interrupt OB is not loaded.
5	0	The execution of the time-of-day interrupt OB is not disabled by an active test function.
6	0	Base for the time-of-day interrupt is the basic time
	1	Base for the time-of-day interrupt is the local time

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	Incorrect parameter OB_NR

10 SFCs for Handling Time-Delay Interrupts

10.1 Handling Time-Delay Interrupts

Definition

After you have called SFC 32 "SRT_DINT," the operating system generates an interrupt after the specified delay time has elapsed, in other words, the selected time-delay interrupt OB is called. This interrupt is known as a time-delay interrupt.

Conditions for the Call

Before a time-delay interrupt OB can be called by the operating system, the following conditions must be met:

- The time-delay interrupt OB must be started by SFC 32 "SRT_DINT."
- The time-delay interrupt OB must not be deselected with STEP 7.
- The time-delay interrupt OB must exist in the CPU.

Purpose of SFC 32 to SFC 34

The system functions SFC 32 to SFC 34 described in the following sections are used as follows:

- To start time-delay interrupts (SFC 32 "SRT_DINT")
- To cancel time-delay interrupts (SFC 33 "CAN_DINT")
- To query time-delay interrupts (SFC 34 "QRY_DINT").

What Happens if...

The following table lists a number of different situations and explains the effect they have on a time-delay interrupt.

If ...	and ...	Then ...
A time-delay interrupt is started (by calling SFC 32 "SRT_DINT").	The time-delay interrupt has already started.	The delay time is overwritten; the time-delay interrupt is started again.
	The time-delay interrupt OB does not exist at the time of the call.	The operating system generates a priority class error (calls OB85). If OB85 does not exist, the CPU changes to STOP.
	The interrupt is started in a startup OB and the delay time elapses before the CPU changes to RUN.	The call of the time-delay interrupt OB is delayed until the CPU is in the RUN mode.
The delay time has elapsed.	A previously started time-delay interrupt OB is still being executed.	The operating system generates a time error (calls OB80). If OB80 does not exist, the CPU changes to STOP.

Warm Restart and Cold Restart

During a warm or cold restart, all the time-delay interrupt settings made in the user program by SFCs are cleared.

Starting in a Startup OB

A time-delay interrupt can be started in a startup OB. To call the time-delay interrupt OB, the following two conditions must be met:

- The delay time must have elapsed.
- The CPU must be in the RUN mode.

If the delay time has elapsed and the CPU is not yet in the RUN mode, the time-delay interrupt OB call is delayed until the CPU is in the RUN mode. The time-delay interrupt OB is then called before the first instruction in OB1 is executed.

10.2 Starting a Time-Delay Interrupt with SFC 32 "SRT_DINT"

Description

With SFC 32 "SRT_DINT" (start time-delay interrupt), you start a time-delay interrupt that calls a time-delay interrupt organization block once the delay time has elapsed (parameter DTIME).

With the SIGN parameter, you can enter an identifier that identifies the start of the time-delay interrupt. The values of DTIME and SIGN appear again in the start event information of the specified OB when it is executed.

Parameter	Declaration	Data Type	Memory Area	Description
OB_NR	INPUT	INT	I, Q, M, D, L, constant	Number of the OB, to be started after a time delay (OB20 to OB23).
DTIME	INPUT	TIME	I, Q, M, D, L, constant	Time delay value (1 to 60000 ms) You can realize longer times, for example, by using a counter in a time-delay interrupt OB.
SIGN	INPUT	WORD	I, Q, M, D, L, constant	Identifier which appears in the start event information of the OB when the time-delay interrupt OB is called.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the system function is active, the actual parameter of RET_VAL contains an error code.

Accuracy

The time between calling SFC 32 "SRT_DINT" and the start of the time-delay interrupt OB is a maximum of **one millisecond** less than the selected time providing that no interrupt event delays the call.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	Incorrect parameter OB_NR
8091	Incorrect parameter DTIME

10.3 Querying a Time-Delay Interrupt with SFC 34 "QRY_DINT"

Description

With SFC 34 "QRY_DINT" (query time-delay interrupt), you can query the status of a time-delay interrupt OB. Time-delay interrupts are managed by organization blocks OB20 to OB23.

Parameter	Declaration	Data Type	Memory Area	Description
OB_NR	INPUT	INT	I, Q, M, D, L, constant	Number of the OB, whose STATUS will be queried (OB20 to OB23).
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being processed, the actual parameter of RET_VAL contains an error code.
STATUS	OUTPUT	WORD	I, Q, M, D, L	Status of the time-delay interrupt, see following table.

Output Parameter STATUS

Bit	Value	Meaning
0	0	Time-delay interrupt is enabled by the operating system.
1	0	New time-delay interrupts are not rejected.
2	0	Time-delay interrupt is not activated or has elapsed.
3	-	-
4	0	Time-delay interrupt-OB is not loaded.
5	0	The execution of the time-delay interrupt OB is not disabled by an active test function.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred
8090	Incorrect parameter OB_NR

10.4 Canceling a Time-Delay Interrupt with SFC 33 "CAN_DINT"

Description

With SFC 33 "CAN_DINT" (cancel time-delay interrupt), you cancel a time-delay interrupt that has already started (see Section Starting a Time-Delay Interrupt with SFC 32 "SRT_DINT"). The time-delay interrupt OB is then not called.

Parameter	Declaration	Data Type	Memory Area	Description
OB_NR	INPUT	INT	I, Q, M, D, L, constant	Number of the OB to be canceled (OB20 to OB23).
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the actual parameter of RET_VAL contains an error code.

Error Information

Error Code (W#16#...)	Explanation
0000	No error has occurred.
8090	Incorrect parameter OB_NR
80A0	Time-delay interrupt has not started.

11 SFCs for Handling Synchronous Errors

11.1 Masking Synchronous Errors

Introduction

Synchronous errors are programming and access errors. Such errors occur as a result of programming with incorrect address areas, numbers, or incorrect addresses. **Masking** these synchronous errors means the following:

- Masked synchronous errors do not trigger an error OB call and do not lead to a programmed alternative reaction.
- The CPU "records" the masked errors that have occurred in an error register.

Masking is carried out by calling the SFC 36 "MSK_FLT".

Unmasking errors means canceling a previously set mask and clearing the corresponding bit in the event status register of the current priority class. Masking is canceled as follows:

- By calling SFC 37 "DMSK_FLT"
- When the current priority class has been completed.

If an error occurs after it has been unmasked, the operating system starts the corresponding error OB. You can program OB121 for a reaction to programming errors and OB122 for a reaction to access errors.

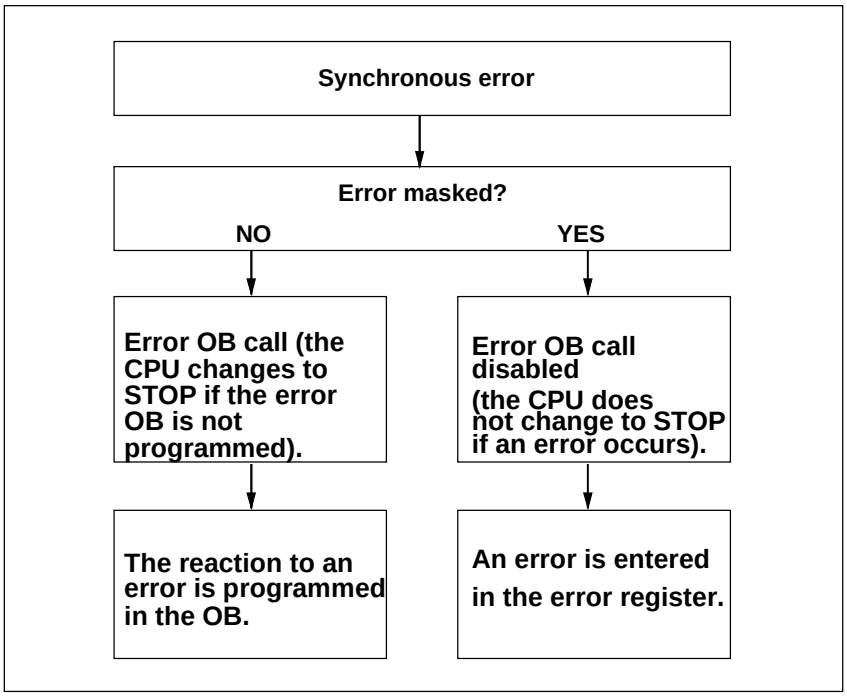
You can use SFC 38 "READ_ERR" to read out the masked errors that have occurred.

Note: With the S7-300 (except CPU 318), regardless of whether an error is masked or unmasked, the error is entered in the diagnostic buffer and the group error LED of the CPU is lit.

Handling Errors in General

If programming and access errors occur in a user program, you can react to them in different ways:

- You can program an error OB that is called by the operating system when the corresponding error occurs.
- You can disable the error OB call individually for each priority class. In this case, the CPU does not change to STOP when an error of this type occurs in the particular priority class. The CPU enters the error in an error register. From this entry, however, you cannot recognize when or how often the error occurred.



Filters

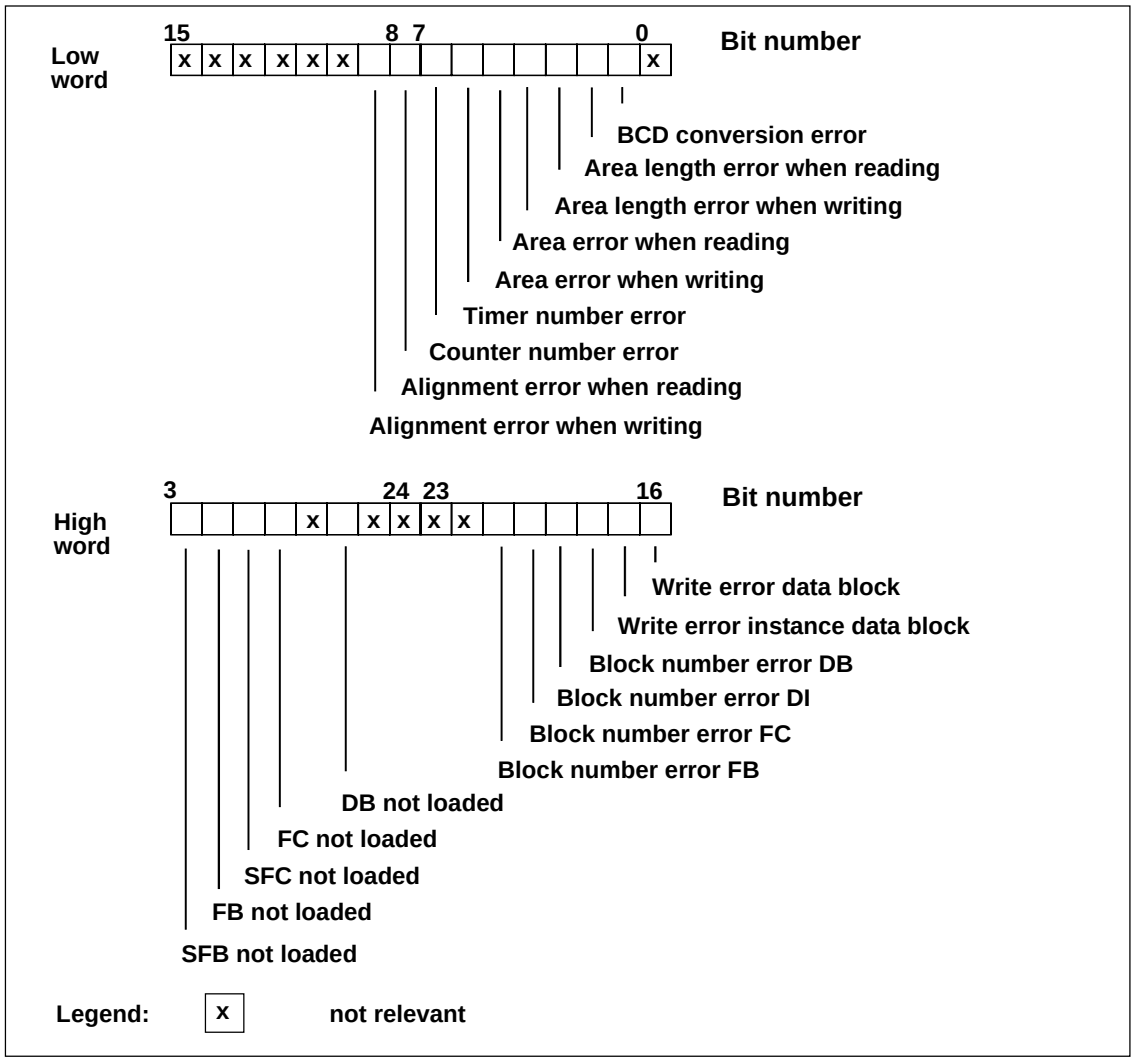
Synchronous errors are assigned to a particular bit pattern known as the **error filter (mask)**. This error filter is also in the input and output parameters of SFCs 36, 37 and 38.

The synchronous errors are divided into programming and access errors that you can mask using two error filters. The error filters are illustrated in the following Figures.

Programming Error Filter

The following figure shows the bit pattern of the error filter for programming errors. The error filter for programming errors is located in the parameters PRGFLT_...

Refer to the Possible Error Causes for Programming Errors, Low Word or the Possible Error Causes for Programming Errors High Word



Note

Bits 29 ("SFC not loaded") and 31 ("SFB not loaded") in the high word of the programming error filter are only relevant for S7-400 and CPU 318.

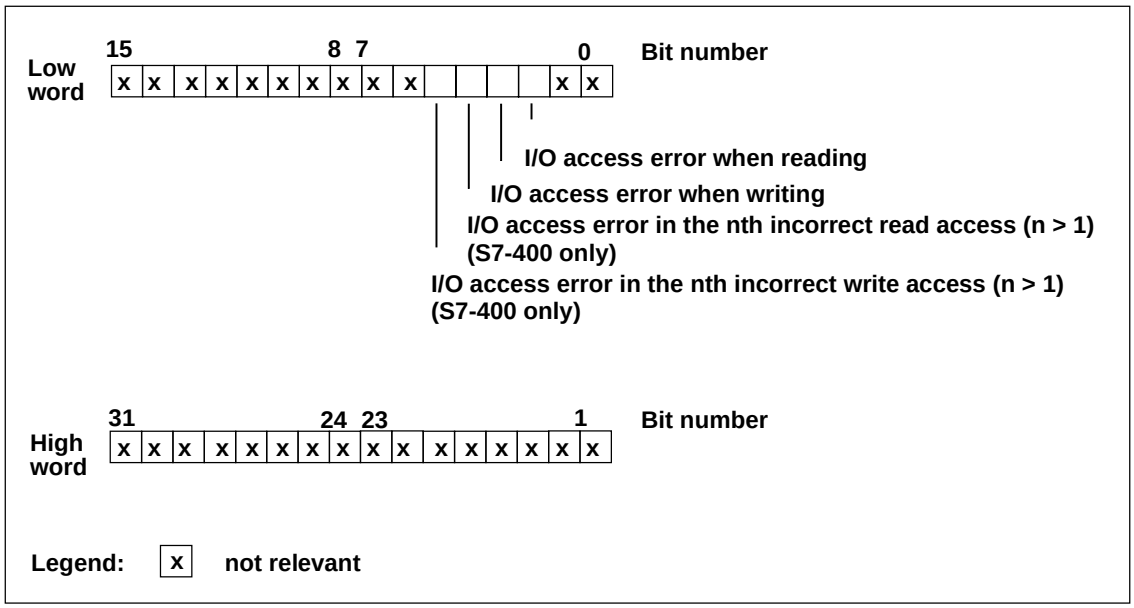
Non-Relevant Bits

In the figure above, **x** means ...

• ... input parameters	for SFC 36, 37, 38	= **"0"**
• ... output parameters	for SFC 36, 37	= **"1"** for S7-300 = **"0"** for S7-400
	for SFC 38	= **"0"**

Access Error Filter for all CPUs

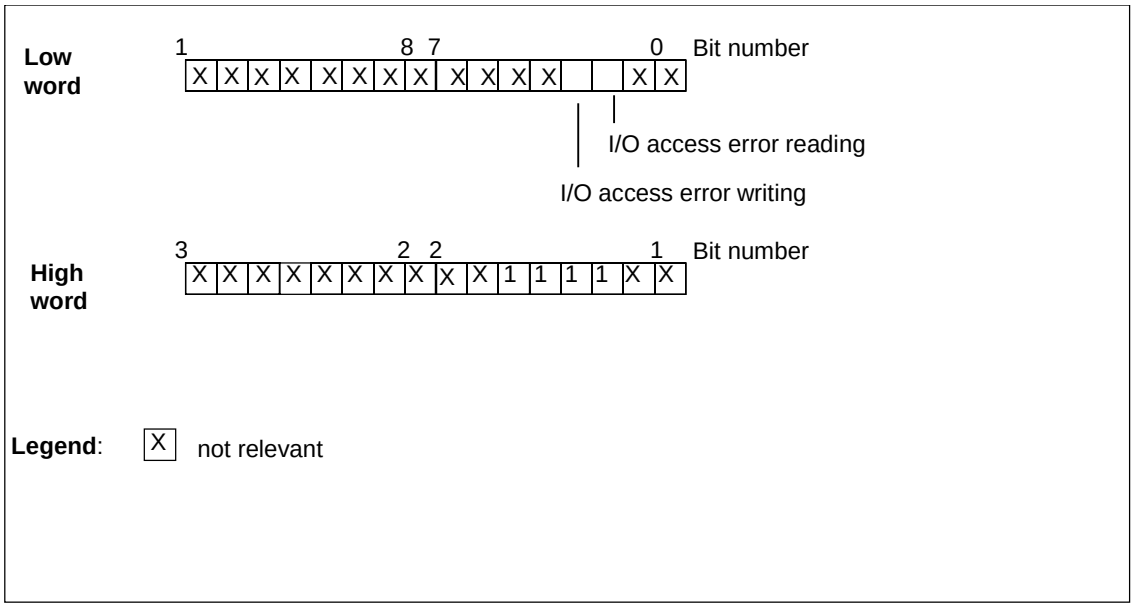
The following figure shows you the bit pattern of the error filter for access errors for all CPUs except CPU 417 and CPU 417H. The error filter for access errors is in the parameters ACCFLT_... For an explanation of the access errors, refer to the tables "Possible Causes of Errors for all CPUs 31x except the CPU 318" or "Possible Causes of Errors for all CPUs except CPUs 41x and CPU 318."



Non-Relevant Bits

In the figure above, **x** means ...

• ... input parameters	for SFC 36, 37, 38	= **"0"**
• ... output parameters	for SFC 36, 37	= **"1"** for S7-300
		= **"0"** for S7-400 and CPU 318
	for SFC 38	= **"0"**



Non-Relevant Bits CPUs 41x and CPU 318

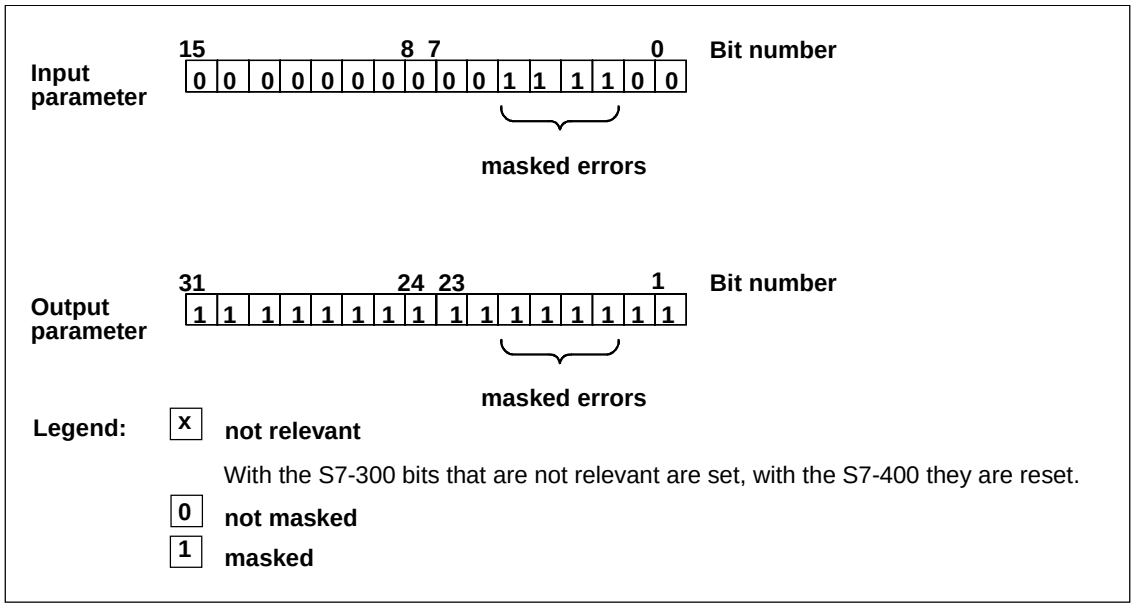
In the figure above, **x** means ...

• ... input parameters	for SFC 36, 37, 38	= **"0"**
• ... output parameters	for SFC 36, 37	= **"0"**
	for SFC 38	= **"0"**

Example for the CPUs 31x (not CPU 318)

The following figure shows the low word of the error filter for access errors with all masked errors for all CPUs except CPU 417 and CPU 417H:

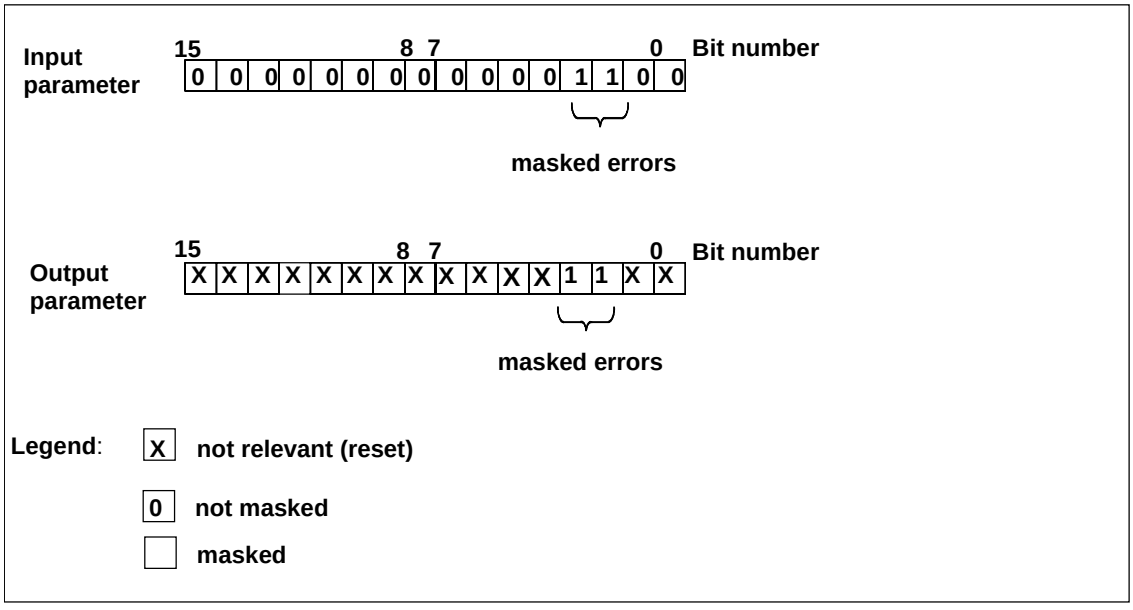
- As an input parameter for SFC 36
- As an output parameter for SFC 36



Example of the CPU 417 and CPU 417H

The following diagram shows how the low word of the error filter for access errors with all masked errors appears for the CPU 417 and CPU 417H.

- As input parameter for SFC 36
- As output parameter for SFC 36.



Programming Error Low Word

The following table lists the errors assigned to the low word of the error filter for programming errors. The table also shows the possible causes of the errors.

Possible Causes of Programming Errors, Low Word

Error	Event ID (W#16#...)	Error Occurs ...
BCD conversion error	2521	... when the value to be converted is not a BCD number (for example, 5E8)
Area length error when reading	2522	... when an address is being used that is not completely within the possible address area. Example: MW 320 must be read although the memory area is only 256 bytes long.
Area length error when writing	2523	... when an address is being used that is not completely within the possible address area. Example: A value must be written to MW 320 although the memory area is only 256 bytes long.
Area error when reading	2524	... when an incorrect area identifier is specified for the address when using indirect, area-crossing addressing. Example: correct: LAR1 P#E 12.0 L W[AR1, P#0.0] incorrect:LAR1 P#12.0 L W[AR1, P#0.0] For this operation, the area length error is signaled.
Area error when writing	2525	... when an incorrect area identifier is specified for the address when using indirect, area-crossing addressing. Example: correct: LAR1 P#E 12.0 T W[AR1, P#0.0] incorrect:LAR1 P#12.0 T W[AR1, P#0.0] For this operation, the area length error is signaled.
Timer number error	2526	... when a non-existent timer is accessed. Example: SP T [MW 0] where MW 0 = 129; timer 129 must be started although there are only 128 timers available.
Counter number error	2527	... when a non-existent counter is accessed. Example: CU C [MW 0] where MW 0 = 600; counter 600 must be accessed although there are only 512 counters available (CPU 416-D).
Alignment error when reading	2528	... when a byte, word or double word address is addressed with a bit address ≠ 0. Example: correct: LAR1 P#M12.0 L B[AR1, P#0.0] incorrect:LAR1 P#M12.4 L B[AR1, P#0.0]
Alignment error when writing	2529	... when a byte, word or double word address is addressed with a bit address ≠ 0. Example: correct: LAR1 P#M12.0 T B[AR1, P#0.0] incorrect:LAR1 P#M12.4 T B[AR1, P#0.0]

Programming Error High Word

The following table lists the errors assigned to the high word of the error filter for programming errors. The possible causes of errors are also listed.

Possible Causes of Programming Errors, High Word

Error	Event ID (W#16#...)	Error Occurs ...
Write error data block	2530	... when the data block to be written to is read only.
Write error instance data block	2531	... when the instance data block to be written to is read only.
Block number error DB	2532	... when a data block must be opened whose number is higher than the highest permitted number.
Block number error DI	2533	... when an instance data block must be opened whose number is higher than the highest permitted number.
Block number error FC	2534	... when a function is called whose number is higher than the highest permitted number.
Block number error FB	2535	... when a function block is called whose number is higher than the highest permitted number.
DB not loaded	253A	... when the data block to be opened is not loaded.
FC not loaded	253C	... when the called function is not loaded.
SFC does not exist	253D	... when the called system function does not exist.
FB not loaded	253E	... when the function block to be called is not loaded.
SFB not existing	253F	... when the called system/standard function block does not exist.

Access Errors

The following table lists the errors assigned to the error filter for access errors for all CPUs. The possible causes of the errors are also listed.

Error	Event ID (W#16#...)	Error Occurs ...
I/O access error when reading	2942	... when no signal module is assigned to the address in the I/O area. Or ... when access to this I/O area is not acknowledged within the selected module watchdog time (timeout).
I/O access error when writing	2943	... when no signal module is assigned to the address in the I/O area. Or ... when access to this I/O area is not acknowledged within the selected module watchdog time (timeout).

11.2 Masking Synchronous Errors with SFC 36 "MSK_FLT"

Description

With SFC 36 "MSK_FLT" (mask synchronous errors), you can control the reaction of the CPU to synchronous errors. With this SFC, you can mask the synchronous errors using the error filter (see Masking Synchronous Errors). When you call SFC 36, you mask the synchronous errors in the current priority class.

If you set individual bits of the synchronous error filter to "1" in the input parameters, other bits that were set previously retain their value "1." You therefore obtain new error filters that you can read out using the output parameters. The synchronous errors you have masked do not call an OB but are simply entered in an error register. You can read out the error register with SFC 38 "READ_ERR".

Parameter	Declaration	Data Type	Memory Area	Description
PRGFLT_SET_MASK	INPUT	DWORD	I, Q, M, D, L, constant	Programming error to be masked
ACCFLT_SET_MASK	INPUT	DWORD	I, Q, M, D, L, constant	Access error to be masked
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
PRGFLT_MASKED	OUTPUT	DWORD	I, Q, M, D, L	Masked program errors
ACCFLT_MASKED	OUTPUT	DWORD	I, Q, M, D, L	Masked access errors

Error Information

Error Code (W#16#...)	Explanation
0000	None of the errors was already masked.
0001	At least one of the errors was already masked. Nevertheless the other errors will be masked.

11.3 Unmasking Synchronous Errors with SFC 37 "DMSK_FLT"

Description

With SFC 37 "DMSK_FLT" (unmask synchronous errors), you unmask the errors masked with SFC 36 "MSK_FLT." To do this, you must set the corresponding bits of the error filter to "1" in the input parameters. With the SFC 37 call, you unmask the corresponding synchronous errors of the current priority class. At the same time, the entries are cleared in the error register. You can read out the new error filters using the output parameters.

Parameter	Declaration	Data Type	Memory Area	Description
PRGFLT_RESET_MASK	INPUT	DWORD	I, Q, M, D, L, constant	Programming errors to be unmasked
ACCFLT_RESET_MASK	INPUT	DWORD	I, Q, M, D, L, constant	Access errors to be unmasked
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
PRGFLT_MASKED	OUTPUT	DWORD	I, Q, M, D, L	Still masked programming errors
ACCFLT_MASKED	OUTPUT	DWORD	I, Q, M, D, L	Still masked access errors

Error Information

Error Code (W#16#...)	Explanation
0000	All specified errors were unmasked.
0001	At least one of the errors was not masked. Nevertheless the other errors will be unmasked.

11.4 Reading the Error Register with SFC 38 "READ_ERR"

Description

Using SFC 38 "READ_ERR" (read error register), you can read the error register. The structure of the error register corresponds to that of the programming and access error filters which you can program as input parameters with SFC 36 and SFC 37.

In the input parameters, you enter the synchronous errors you want to read from the error register. When you call SFC 38, you read the required entries from the error register and at the same time clear the entries.

The error register contains information that tells you which of the masked synchronous errors in the current priority class occurred at least once. If a bit is set, this means that the corresponding masked synchronous error occurred at least once.

Parameter	Declaration	Data type	Memory area	Description
PRGFLT_QUERY	INPUT	DWORD	I, Q, M, D, L, Const.	Query program error
ACCFLT_QUERY	INPUT	DWORD	I, Q, M, D, L, Const.	Query access error
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error Information
PRGFLT_CLR	OUTPUT	DWORD	I, Q, M, D, L	Occurred programming errors
ACCFLT_CLR	OUTPUT	DWORD	I, Q, M, D, L	Occurred access errors

Error Information

Error Code (W#16#...)	Explanation
0000	All queried errors are masked.
0001	At least one of the queried errors is not masked.

12 SFCs for Handling Interrupts and Asynchronous Errors

12.1 Delaying and Disabling Interrupt and Asynchronous Errors

Purpose of SFC 39 to SFC 42

With these SFCs, you can achieve the following:

- Disable interrupts and asynchronous errors with SFC 39 "DIS_IRT" for all subsequent CPU cycles.
- Delay higher priority classes with SFC 41 "DIS_AIRT" until the end of the OB.
- Enable interrupts and asynchronous errors with SFC 40 "EN_IRT" or SFC 42 "EN_AIRT."

You program the handling of interrupts and asynchronous errors in the user program. You must also program the corresponding OBs.

Advantage of SFC 41 and SFC 42

Delaying higher priority interrupts and asynchronous errors by disabling them with SFC 41 "DIS_AIRT" and then enabling them again with SFC 42 "EN_AIRT" has the following advantages:

The number of interrupts delayed is counted by the CPU. If you have delayed interrupts and asynchronous errors, the delay cannot be canceled by standard FC calls if the interrupts and asynchronous errors are also disabled and then enabled again in the standard FCs themselves.

Interrupt Classes

The interrupts are divided into various classes. The following table lists all the interrupt classes and the corresponding OBs.

Interrupt Class	OB
Time-of-day interrupts	OB10 to OB17
Time-delay interrupts	OB20 to OB23
Cyclic interrupts	OB30 to OB38
Hardware interrupts	OB40 to OB47
Interrupts for DPV1	OB55 to OB57
Multicomputing interrupt	OB60
Redundancy error interrupts	OB70, OB72
Asynchronous error interrupts	OB80 to OB87 (se below)
Synchronous error interrupts	OB121, OB122 (You can mask or unmask the processing of synchronous error interrupts with SFC 36 to SFC 38)

Asynchronous Errors

The following table lists all the asynchronous errors to which you can react with an OB call in the user program.

Asynchronous Errors	OB
Time error (for example, cycle time exceeded)	OB80
Power supply error (for example, battery fault)	OB81
Diagnostic interrupt (for example, defective fuse on a signal module)	OB82
Remove/insert module interrupt	OB83
CPU hardware fault (for example, memory card removed)	OB84
Program error	OB85
Rack failure	OB86
Communication error	OB87

12.2 Disabling the Processing of New Interrupts and Asynchronous Errors with SFC 39 "DIS_IRT"

Description

With SFC 39 "DIS_IRT" (disable interrupt), you disable the processing of new interrupts and asynchronous errors. This means that if an interrupt occurs, the operating system of the CPU reacts as follows:

- It **neither** calls an interrupt OB or asynchronous error OB,
- **Nor** triggers the normal reaction if an interrupt OB or asynchronous error OB is not programmed.

If you disable interrupts and asynchronous errors, this remains in effect for all priority classes. The effects of "DIS_IRT" can only be canceled again by calling SFC 40 "EN_IRT" (see Section 0) or by a warm or a cold restart.

Whether the operating system writes interrupts and asynchronous errors to the diagnostic buffer when they occur depends on the input parameter setting you select for MODE.

Note

Remember that when you program the use of SFC 39 "DIS_IRT," all interrupts that occur are lost!

Parameter	Declaration	Data Type	Memory Area	Description
MODE	INPUT	BYTE	I, Q, M, D, L, constant	Specifies which interrupts and asynchronous errors are disabled.
OB_NR	INPUT	INT	I, Q, M, D, L, constant	OB number
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code.

MODE

MODE	Meaning
00	All newly occurring interrupts and asynchronous errors are disabled. (Synchronous errors are not disabled.) Assign the OB_NR parameter the value 0. Entries continue to be made in the diagnostic buffer.
01	All newly occurring events belonging to a specified interrupt class are disabled. Identify the interrupt class by specifying the smallest possible OB no. for this class, even if this OB does not exist on your CPU. Examples: • If you want to disable all watchdog interrupts, specify 30 in OB_NR (even if OB30 is not the first watchdog OB for your CPU). • If you want to disable all hardware interrupts, specify 40 in OB_NR. Entries into the diagnostic buffer are continued.
01	All newly occurring events belonging to a specified interrupt class are disabled. Identify the interrupt class by specifying it as follows: • Time- of-day interrupts: 10 • Time-delay interrupts: 20 • Cyclic interrupts: 30 • Hardware interrupts: 40 • Interrupts for DPV1: 50 • Multicomputing interrupts: 60 • Redundancy error interrupts: 70 • Asynchronous error interrupts: 80 Entries into the diagnostic buffer are continued.
02	All new occurrences of a specified interrupt are disabled. You specify the interrupt using the OB number. Entries continue to be made in the diagnostic buffer.
80	All newly occurring interrupts and asynchronous errors are disabled and are no longer entered in the diagnostic buffer. The operating system enters event W#16#5380 in the diagnostic buffer.
81	All newly occurring belonging to a specified interrupt class are disabled and are no longer entered in the diagnostic buffer. The operating system enters event W#16#5380 in the diagnostic buffer.
82	All newly occurring belonging to a specified interrupt are disabled and are no longer entered in the diagnostic buffer. The operating system enters event W#16#5380 in the diagnostic buffer.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	The input parameter OB_NR contains an illegal value.
8091	The input parameter MODE contains an illegal value.

12.3 Enabling the Processing of New Interrupts and Asynchronous Errors with SFC 40 "EN_IRT"

Description

With SFC 40 "EN_IRT" (enable interrupt), you enable the processing of new interrupts and asynchronous errors that you previously disabled with SFC 39 "DIS_IRT." This means that if an interrupt event occurs, the operating system of the CPU reacts in one of the following ways:

- It calls an interrupt OB or asynchronous error OB.
- It triggers the standard reaction if the interrupt OB or asynchronous error OB is not programmed.

Parameter	Declaration	Data Type	Memory Area	Description
MODE	INPUT	BYTE	I, Q, M, D, L, constant	Specifies which interrupts and asynchronous errors will be enabled.
OB_NR	INPUT	INT	I, Q, M, D, L, constant	OB number
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active the return value contains an error code.

MODE

MODE (B#16#...)	Meaning
0	All newly occurring interrupts and asynchronous errors are enabled.
1	All newly occurring events belonging to a specified interrupt class are enabled. Identify the interrupt class by specifying it as follows: • Tim- of-day interrupts: 10 • Time-delay interrupts: 20 • Cyclic interrupts: 30 • Hardware interrupts: 40 • Interrupts for DPV1: 50 • Multicomputing interrupts: 60 • Redundancy error interrupts: 70 • Asynchronous error interrupts: 80
2	All newly occurring events of a specified interrupt are enabled. You specify the interrupt using the OB number.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	The input parameter OB_NR contains an illegal value.
8091	The input parameter MODE contains an illegal value.

12.4 Delaying the Processing of Higher Priority Interrupts and Asynchronous Errors with SFC 41 "DIS_AIRT"

Description

With SFC 41 "DIS_AIRT" (disable alarm interrupts), you delay the processing of interrupt OBs and asynchronous error OBs which have a higher priority than that of the current OB. You can call SFC 41 more than once in an OB. The SFC 41 calls are counted by the operating system. Each of these calls remains in effect until it is canceled again specifically by an SFC 42 "EN_AIRT" call or until the current OB has been completely processed.

Once they are enabled again, the interrupts and asynchronous errors that occurred while SFC 41 was in effect are processed as soon as they are enabled again with SFC 42 "EN_AIRT" or as soon as the current OB has been executed.

Parameter	Declaration	Data Type	Memory Area	Description
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Number of delays (= number of SFC 41 calls)

Return Value

The following table shows the return value for SFC 41 that is output with the RET_VAL parameter.

Return Value	Description
n	"n" shows the number of times that processing was disabled, in other words the number of SFC 41 calls (interrupt processing is only enabled again when n = 0; see Enabling the Processing of Higher Priority Interrupts and Asynchronous Errors with SFC 42 "EN_AIRT").

12.5 Enabling the Processing of Higher Priority Interrupts and Asynchronous Errors with SFC 42 "EN_AIRT"

Description

With SFC 42 "EN_AIRT" (enable alarm interrupts), you enable the processing of higher priority interrupts and asynchronous errors that you previously disabled with SFC 41 "DIS_AIRT." Each SFC 41 call must be canceled by an SFC 42 call.

Example

If, for example, you have disabled interrupts five times with five SFC 41 calls, you must cancel these calls with five SFC 42 calls.

Parameter	Declaration	Data Type	Memory Area	Description
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Number of delays still programmed on completion of SFC 42 or error message.

Return Value and Error Information

Refer to Evaluating Errors with the Output Parameter RET_VAL

How you evaluate the error information of the RET_VAL parameter is explained in Chapter 0. This chapter also contains the general error information for the SFCs. The following table contains all the error information specific to SFC 42 that can be output with the RET_VAL parameter.

Return Value and Error Information	Description
N	"n" shows the number of SFC 41 calls not yet canceled by SFC 42 calls (interrupt processing is only enabled again when "n" = 0).
W#16#8080	The function has been called again although interrupt processing was already enabled.

13 SFCs for Diagnostics

13.1 System Diagnostics

The CPUs maintain internal data about the status of the programmable logic controller. With the system diagnostics functions, you can read out the most important data. Some of the data can be displayed on the programming device using STEP 7.

You can also access the data required for system diagnostics in your program, by using the SFCs "RD_SINFO" and "RDSYSST."

13.2 Reading OB Start Information with SFC 6 "RD_SINFO"

Description

With SFC 6 "RD_SINFO" (read start information), you can read the start information about the following:

- The last OB to be called that has not yet been completely executed

  and

- The last startup OB to be started.

There is no time stamp in either case. If the call is in OB100 or OB101 or OB102, two identical start information messages are returned.

Parameter	Declaration	Data Type	Memory Area	Description
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
TOP_SI	OUTPUT	STRUCT	D, L	Start information of the current OB
START_UP_SI	OUTPUT	STRUCT	D, L	Start information of the startup OB last started

TOP_SI and START_UP_SI

The output parameters TOP_SI and START_UP_SI are two structures with identical elements (see following table).

Structure Element	Data Type	Description
EV_CLASS	BYTE	• Bits 0 to 3: Event ID • Bits 4 to 7: Event class
EV_NUM	BYTE	Event number
PRIORITY	BYTE	Number of the priority class
NUM	BYTE	OB number
TYP2_3	BYTE	Data ID 2_3: identifies the information entered in ZI2_3
TYP1	BYTE	Data ID 1: identifies the information entered in ZI1
ZI1	WORD	Additional information 1
ZI2_3	DWORD	Additional information 2_3

Note

The structure elements listed in the table and temporary variables of an OB have an identical content.

Please note that temporary variables of the individual OBs could however have different names and different data types. Also note that the call interface of each OB includes additional information which is the date and the time of the OB request.

Bits 4 to 7 of the EV_CLASS structure element contain the event class. The following values are possible here:

- 1: Start events from standard OBs
- 2: Start events from synchronous error OBs
- 3: Start events from asynchronous error OBs

The PRIORITY structure element supplies the priority class belonging to the current OB (see Chapter 0).

Apart from these two elements, NUM is also relevant. NUM contains the number of the current OB or the startup OB that was started last.

Example

The OB that was called last and that has not yet been completely processed serves as OB80. The start-up OB that was started last serves as OB100.

The following table shows the assignment of the structure elements of parameter TOP_SI of SFC 6 "RD_SINFO" and the respective local variables of OB80.

TOP_SI		**OB80**	
Structure Element	**Data Type**	**Local Variable**	**Data Type**
EV_CLASS	BYTE	OB80_EV_CLASS	BYTE
EV_NUM	BYTE	OB80_FLT_ID	BYTE
PRIORITY	BYTE	OB80_PRIORITY	BYTE
NUM	BYTE	OB80_OB_NUMBR	BYTE
TYP2_3	BYTE	OB80_RESERVED_1	BYTE
TYP1	BYTE	OB80_RESERVED_2	BYTE
ZI1	WORD	OB80_ERROR_INFO	WORD
ZI2_3	DWORD	OB80_ERR_EV_CLASS	BYTE
		OB80_ERR_EV_NUM	BYTE
		OB80_OB_PRIORITY	BYTE
		OB80_OB_NUM	BYTE

The following table shows the assignment of the structure elements of parameter START_UP_SI of SFC 6 "RD_SINFO" and the respective local variables of OB100.

START_UP_SI		**OB100**	
Structure Element	**Data Type**	**Local Variable**	**Data Type**
EV_CLASS	BYTE	OB100_EV_CLASS	BYTE
EV_NUM	BYTE	OB100_STRTUP	BYTE
PRIORITY	BYTE	OB100_PRIORITY	BYTE
NUM	BYTE	OB100_OB_NUMBR	BYTE
TYP2_3	BYTE	OB100_RESERVED_1	BYTE
TYP1	BYTE	OB100_RESERVED_2	BYTE
ZI1	WORD	OB100_STOP	WORD
ZI2_3	DWORD	OB100_STRT_INFO	DWORD

Error Information

SFC 6 "RD_SINFO" does not provide any specific error information but only general error information. The general error codes and how to evaluate them are described in detail in Evaluating Errors with the Output Parameter RET_VAL in the section entitled "General Parameters for SFCs".

13.3 Reading a System Status List or Partial List with SFC 51 "RDSYSST"

Description

With system function SFC 51 "RDSYSST" (read system status), you read a system status list or a partial system status list.

You start the reading by assigning the value 1 to the input parameter REQ when SFC 51 is called. If the system status could be read immediately, the SFC returns the value 0 at the BUSY output parameter. If BUSY has the value 1, the read function is not yet completed (see Section 0).

Note

If you call SFC 51 "RDSYSST" in the diagnostic interrupt OB with the SSL-ID W#16#00B1 or W#16#00B2 or W#16#00B3 and access the module that initiated the diagnostic interrupt, the system status is read immediately.

System Resources

If you start several asynchronous read functions (the jobs with SSL_ID W#16#00B4 and W#16#4C91 and W#16#4092 and W#16#4292 and W#16#4692 and possibly W#16#00B1 and W#16#00B3) one after the other at brief intervals, the operating system ensures that all the read jobs are executed and that they do not interfere with each other. If the limits of the system resources are reached, this is indicated in RET_VAL. You can remedy this temporary error situation by repeating the job.

The maximum number of "simultaneously" active SFC 51 jobs depends on the CPU. You will find this information in /**70**/ and /**101**/.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	REQ=1: Starts processing
SSL_ID	INPUT	WORD	I, Q, M, D, L, constant	SSL-ID of the system status list or partial list to be read (the partial lists are explained in Chapter 0).
INDEX	INPUT	WORD	I, Q, M, D, L, constant	Type or number of an object in a partial list.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while executing the SFC, the RET_VAL parameter contains an error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	TRUE: Reading not yet completed.
SSL_HEADER	OUTPUT	STRUCT	D, L	See below.

DR	OUTPUT	ANY	I, Q, M, L, D	Destination area of the SSL list read or the SSL partial list read: • If you have only read out the header information of an SSL list, you must not evaluate DR but only SSL_HEADER. • Otherwise, the product of LENTHDR and N_DR indicates how many bytes were entered in DR.

SSL_HEADER

The SSL_HEADER parameter is a structure defined as follows:

```
SSL_HEADER: STRUCT
    LENTHDR:    WORD
    N_DR:       WORD
END_STRUCT
```

LENTHDR is the length of a data record of the SSL list or the SSL partial list.

- If you have only read out the header information of an SSL list, N_DR contains the number of data records belonging to it.
- Otherwise, N_DR contains the number of data records transferred to the destination area.

Error Information

Error Code (W#16#...)	Description
0000	No error.
0081	Result field too short. (Nevertheless as many data records as possible are supplied. The SSL header indicates this number.)
7000	First call with REQ=0: No data transfer active; BUSY has the value 0.
7001	First call with REQ=1: Data transfer started; BUSY has the value 1.
7002	Interim call (REQ irrelevant): Data transfer already active; BUSY has the value 1.
8081	Result field too short (not enough space for one data record).
8082	SSL_ID is wrong or is unknown in the CPU or SFC.
8083	INDEX wrong or not permitted.
8085	Due to a problem in the system, information is not currently available (for example, due to a lack of resources).
8086	The data record cannot be read due to a system error (bus, modules, operating system).
8087	Data record cannot be read because the module does not exist or does not acknowledge.
8088	Data record cannot be read because the actual module identifier is different from the expected module identifier.
8089	Data record cannot be read because the module is not capable of diagnostics.

Error Code (W#16#...)	Description
80A2	DP protocol error (layer 2 error) (temporary error)
80A3	DP protocol error with user interface/user (temporary error)
80A4	Communication problem on communication bus (error occurs between the CPU and the external DP interface module)
80C5	Distributed I/Os not available (temporary error).
80C6	Data record transfer stopped due to priority class abort (restart or background)

SSL_IDs

Note

For the partial lists that can be read out with SFC 51 "RDSYSST" refer to

- /**70**/ for the S7-300
- The following table for the S7-400.

SSL_ID (W#16#...)	Partial List	INDEX (W#16#...)
	Module ID	
0111	One identification data record	
	Identification of the module	0001
	Identification of the basic hardware	0006
	Identification of the basic hardware	0007
	CPU characteristics	
0012	All characteristics	Irrelevant
0112	Characteristics of one group	
	MC7 processing unit	0000
	Time system	0100
	System behavior	0200
	MC7 language description	0300
	Availability of SFCs	0400
0F12	Only SSL partial list header information	Irrelevant
	User memory areas	
0113	One data record for the memory area specified	
	Work memory	0001
	System areas	
0F14	Data records of all system areas	Irrelevant
0F14	Only SSL partial list header information	Irrelevant
	Module types	
0015	Data records of all module types	Irrelevant
	Status of the module LEDs (cannot be read out from all CPUs, see **/102/**).	

SSL_ID (W#16#...)	Partial List	INDEX (W#16#...)
0019	Status of all LEDs	Irrelevant
0F19	Only SSL partial list header information	Irrelevant
	Identification of one component	
001C	Identification of all components	Irrelevant
011C	Identification of one component	
	Name of the automation system	0001
	Name of the CPU	0002
	System ID of the CPU	0003
	Copyright entry	0004
	reserved for the operating system	0006
0F1C	Only SSL partial list header information	Irrelevant
	Interrupt status	
0222	Data record of the interrupt specified	OB number
	Communication status data	
0132	Status data for one communication unit	
	Diagnostics	0005
	Time system	0008
0232	Status data for one communication unit	
	CPU protection level and operator control settings	0004
	H CPU group information	
0071	Information about the current state of the H system	Irrelevant
0F71	Only SSL partial list header information	Irrelevant
	State of the module LEDs (cannot be read out from all CPUs, see **/102/**).	
0174	State of an LED	LED ID
	DP Master system information	
0090	Information DP Master systems known to the CPU	0000
0190	Information about a DP Master system	DP master system ID
0F90	Only SSL partial list header information	0000
	Module status information (a maximum of 27 data records is supplied)	
0091	Status information of all modules / submodules inserted	Irrelevant
0191	Module status information of all modules / racks with incorrect type ID	Irrelevant
0291	Module status information of all faulty modules	Irrelevant
0391	Module status information of all unobtainable modules	Irrelevant
0591	Module status information of all submodules of the host module	Irrelevant
0991	Module status information of all submodules of the host module in the rack specified	Rack or DP master

SSL_ID (W#16#...)	Partial List	INDEX (W#16#...)
		system ID
0A91	Module status information of all DP master systems	Irrelevant
0C91	Module status information of a module in a central configuration or connected to an integrated DP communications processor	Logical base address
4C91	Module status information of a module connected to an external DP communications processor	Logical base address
0D91	Module status information of all modules in the rack / DP station specified	Rack or DP master system ID or DP master system ID and station number
0E91	Module status information of all modules allocated	Irrelevant
	Rack/station status information	
0092	Expected status of the rack in the central configuration / of the stations of a DP master system	0 / DP master system ID
4092	Expected status of the stations of a DP master system connected to an external DP interface	DP master system ID
0292	Actual status of the rack in the central configuration / of the stations of a DP master system	0 / DP master system ID
4292	Actual status of the stations of a DP master system connected via an external DP interface module	DP master system ID
0692	OK state of the expansion racks in a central configuration / of the stations of a DP master system connected via an integrated DP interface module	0 / DP master system ID
4692	OK state of the stations of a DP master system connected via an external DP interface module	DP master system ID
	Diagnostic buffer (a maximum of 21 data records is supplied)	
00A0	All entries that can be supplied in the currently active operating mode	Irrelevant
01A0	The most recent entries, the number is specified in the index	Quantity
0FA0	Only SSL partial list header information	Irrelevant
	Diagnostic data on modules	
00B1	The first four diagnostic bytes of one module (data record 0)	Logical base address
00B2	All diagnostic data of one module ($\leq$ 220 bytes, data record 1) (no DP module)	Rack, slot
00B3	All diagnostic data of one module ($\leq$ 220 bytes, data record 1)	Logical base address
00B4	Diagnostic data of a DP slave	Configured diagnostic address

13.4 Writing a User-Defined Diagnostic Event to the Diagnostic Buffer with SFC 52 "WR_USMSG"

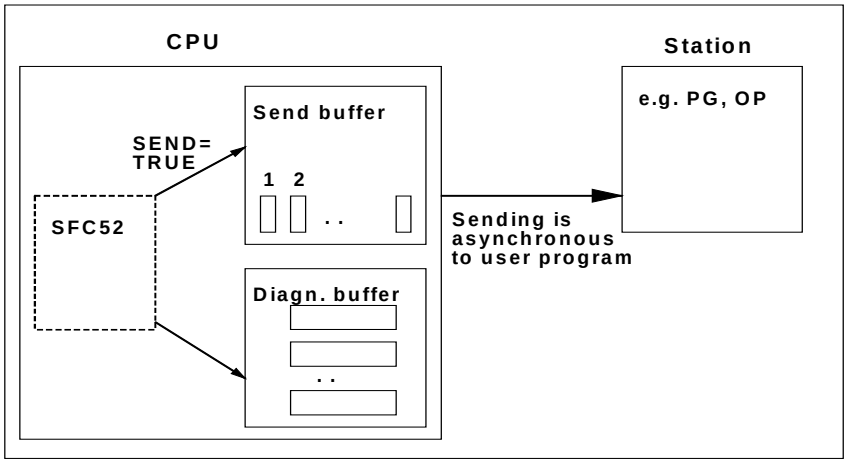
Description

With SFC 52 "WR_USMSG" (write user element in diagnostic buffer), you write a user-defined diagnostic event to the diagnostic buffer. You can also send the corresponding diagnostic message to all stations logged on for this purpose (by setting the input parameter SEND = TRUE). If an error occurs, the output parameter RET_VAL provides the error information.

Sending a User-Defined Diagnostic Message

SFC 52 writes a user-defined diagnostic event to the diagnostic buffer. You can then also send the corresponding diagnostic message to any station logged on for this purpose (by setting the input parameter SEND = TRUE). The user-defined diagnostic message is then written to the send buffer and automatically sent to the logged on stations.

You can check whether the sending of user-defined diagnostic messages is currently possible. To do this, call SFC 51 "RDSYSST" with the parameters SSL_ID = W#16#0132 and INDEX = W#16#0005. The fourth word of the data record obtained indicates whether sending a user element is currently possible (1) or not (0).



Send Buffer Full

The diagnostic message can only be entered in the send buffer if the send buffer is not full. The number of entries that can be made in the send buffer depends on the type of CPU you are using.

If the send buffer is full, then:

- The diagnostic event is nevertheless entered in the diagnostic buffer,

- The parameter RET_VAL indicates that the send buffer is full (RET_VAL = W#16#8092).

Station Not Logged On

If a user-defined diagnostic message is to be sent (SEND = TRUE) and no station is logged on,

- The user-defined diagnostic event is entered in the diagnostic buffer,
- The parameter RET_VAL indicates that no station is logged on (RET_VAL = W#16#8091 or W#16#8091. The value W#16#8091 appears with older versions of the CPU).

General Structure

The internal structure of an element in the diagnostic buffer is as follows:

Byte	Contents
1 and 2	Event ID
3	Priority class
4	OB number
5 and 6	Reserved
7 and 8	Additional information 1
9, 10, 11, and 12	Additional information 2
13 to 20	Time stamp

Event ID

An event ID is assigned to every event.

Additional Information

This is additional information about the event. The additional information can be different for each event. When you create a diagnostic event, you can decide on the content of these entries yourself.

When you send a user-defined diagnostic message, you can integrate the additional information as associated values in the (event ID-specific) message text.

Time Stamp

The time stamp is of the type Date_and_Time.

Parameter	Declaration	Data Type	Memory Area	Description
SEND	INPUT	BOOL	I, Q, M, D, L, constant	Enable the sending of the user-defined diagnostic message to all logged-on stations

Parameter	Declaration	Data Type	Memory Area	Description
EVENTN	INPUT	WORD	I, Q, M, D, L, constant	Event ID - You assign the event ID. This is not assigned by the message server.
INFO1	INPUT	ANY	I, Q, M, D, L	Additional information 1 word long
INFO2	INPUT	ANY	I, Q, M, D, L	Additional information 2 words long
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information

SEND

If SEND = TRUE, the user-defined diagnostic message is sent to all logged-on stations. The message is only sent if the station is logged on and if the send buffer is not full. The sending of the element is asynchronous to the user program.

EVENTN

The EVENTN parameter contains the event ID of the user event. You can enter event IDs of the types W#16#8xyz, W#16#9xyz, W#16#Axyz, W#16#Bxyz.

IDs in the format W#16#8xyz and W#16#9xyz belong to predefined events, IDs in the format W#16Axyz and W#16#Bxyz belong to freely defined events.

An event entering the state is indicated by x = 1, an event leaving the state by x = 0. With events in class A and B, "yz" is the message number assigned to the message in the message configuration in hexadecimal format.

The structure of the event ID is explained in Section 26.1.

INFO1

The INFO1 parameter contains information that is one word long. The following data types are permitted for INFO1:

- WORD
- INT
- ARRAY [0 to 1] OF CHAR

You can integrate the parameter INFO1 as an associated value in the message text and therefore add up-to-date information to the message.

INFO2

The INFO2 parameter contains information that is two words long. The following data types are permitted for INFO2:

- DWORD
- DINT
- REAL
- TIME
- ARRAY [0 to 3] OF CHAR

You can integrate the parameter INFO2 as an associated value in the message text and therefore add up-to-date information to the message.

Error Information

Error Code (W#16#...)	Explanation
0000	No error
0091	No station logged on (diagnostic event entered in the diagnostic buffer)
8083	Data type of INFO1 not permitted
8084	Data type of INFO2 not permitted
8085	EVENTN not permitted
8086	Length of INFO1 not permitted
8087	Length of INFO2 not permitted
8091	(This error code appears only with older versions of the CPU.) No station logged on (diagnostic event entered in the diagnostic buffer).
8092	Sending not possible at present, send buffer full (diagnostic event entered in the diagnostic buffer).

13.5 Query the Actual Connection Status with SFC 87 "C_DIAG"

Description

You can use SFC 87 "C_DIAG" to determine the actual status of all S7 connections and of all currently available S7 connections (or their partial connection).

Suitable evaluation of these connection data lets you recognize failures of S7 connections as well as of actual S7 connections and report these, should the occasion arise, to an operating and visualization system. Monitored connections can be a connection between automation systems as well as the connection of an automation system to an operating and visualization system.

Operating Mode

The SFC 87 "C_DIAG" is an asynchronous SFC operation, that is to say, processing can be extended over multiple SFC calls.

You start the job by calling the SFC 87 with REQ=1.

If it was possible to execute the job immediately, the SFC returns the value 0 in the output parameter BUSY. If BUSY is 1 the job is still active.

When Do You Call the SFC 87?

To recognize the failure of S7 connections and actual S7 connections, call the SFC87 in a cyclic interrupt OB that is started, for example, every 10 seconds by the operating system.

Since the status of a connection normally does not change, it is appropriate to copy the connection data to the user program with these cyclic calls only if they have changed since their last call (call with MODE=B#16#02, see below).

How Do You Call the SFC 87?

The SFC 87 "C_DIAG" offers four possible operating modes which are explained in the table below.

MODE (B#16#...)	SFC Copies Connection Data to the User Program	SFC Transfers Acknowledgement Information to the Operating System
00	No	Yes
01	Yes	Yes
02	• Yes, if connection data have changed • No, if connection data has not changed	Yes
03	Yes	No

The status changes of the connection data since the last call of SFC 87 (with MODE=B#16#00, 01 or 02) are confirmed by transferring the acknowledgement information to the operating system.

Note

If you operate SFC 87 in a cyclic interrupt OB in "Conditional Copying" mode (MODE=B#16#02), you must ensure that no initializing values are contained in the target area after a cold start of the CPU. You can achieve this in OB 102 with a single call of SFC 87 in "Unconditional Copying with Acknowledgement" mode (MODE=B#16#01).

Parameters	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	E, A, M, D, L, constant.	Control parameter request to activate REQ=1: Initialize the job, if not already started
MODE	INPUT	BYTE	E, A, M, D, L, constant.	Job designation Possible values: • B#16#00: The SFC does not copy connection data, but merely transfers an acknowledgement information to the operating system . • B#16#01: Regardless of the status change, the SFC copies all connection data to the user program and transfers an acknowledgement information to the operating system. • B#16#02: If connection data have changed, the SFC copies them to the user program. If not changed, they are not copied. In both cases the SFC transfers an acknowledgement information to the operating system. • B#16#03: The SFC copies the connection data to the user program, independent of the changed status. It does not transfer an acknowledgement information to the operating system.
RET_VAL	OUTPUT	INT	E, A, M, D, L	Return value (Error code or job status)
BUSY	OUTPUT	BOOL	E, A, M, D, L	BUSY =1: The job is not completed.
N_CON	OUTPUT	INT	E, A, M, D, L	Index of the last structure in CON_ARR with .DIS_PCON or .DIS_CON value TRUE. Thus, in the user program only the first N_CON elements of CON_ARR need to be checked. Note: The first structure in the field CON_ARR has the index 1.
CON_ARR	OUTPUT	ANY	E, A, M, D, L	Target area for the received connection data. Only the data type BYTE is permissible. A structure is assigned to each connection. Choose a target area size which can receive all structures even at the maximum number of possible connections for your CPU.

Organizing the Target Area CON_ARR

The read target area is a structure field. A structure is assigned to each connection.

The field does not need to be occupied from the beginning and it can contain invalid entries between two valid connections.

The connections are not sorted by connection reference.

Note

Date consistency of a connection is ensured if you copy connection data from the operating system to the selected target area

Structure Organization

Parameters	Data Type	Description
CON_ARR[i]. CON_ID	WORD	Connection reference which you have assigned in NETPRO for this connection W#16#FFFF: Invalid designation, that is to say, the connection is not configured. If CON_ARR[i].DIS_PCON or CON_ARR[i].DIS_CON (see below) is set, this connection has been reconfigured or deleted since the last call of the SFC 87.
CON_ARR[i]. STAT_CON	BYTE	Actual status of the S7 connection or actual S7 connection Possible values: • B#16#00: S7 connection not established • B#16#10: Actual S7 connection not terminated • B#16#01: S7 connection is currently being established • B#16#11: Actual S7 connection is currently terminated • B#16#02: S7 connection is established • B#16#12: Actual S7 connection is established (one partial connection is established) • B#16#13: Actual S7 connection is established with two partial connections
CON_ARR[i]. PROD_CON	BYTE	Partial connection number of the productive connection. Possible Values: 0, 1, 2, 3
CON_ARR[i]. STBY_CON	BYTE	Partial connection number of the standby connection (B#16#FF: no standby connection) Possible values: 0, 1, 2, 3 Note: Only an actual S7 connection can have a standby connection.

Error Information

Error Code (W#16#...)	Description
0000	• MODE=B#16#00, 01 or 02: No connection status change (structure element STAT_CON) since the last call. The call was executed without error. • MODE=B#16#03: The copy procedure was carried out without error.
0001	• MODE=B#16#00, 01 or 02: Connection status change (structure element STAT_CON) with at least one connection since the last call. The job was carried out without error. • MODE=B#16#03: RET_VAL W#16#0001 is not possible:
7000	First call with REQ=0. The job specified in MODE cannot be processed. BUSY value is 0.
7001	First call with REQ=1. The job specified in MODE has been initialized. BUSY value is 1
7002	Intermediate call (REQ irrelevant). Job still running. BUSY value is 1.
8000	Illegal value in the MODE parameter .
8001	Illegal data type in the CON_ARR parameter .
8002	Length description in the CON_ARR parameter too small. SFC copies no data to the target area.

14 SFCs and SFBs for Updating the Process Image and Processing Bit Fields

14.1 Updating the Process Image Input Table with SFC 26 "UPDAT_PI"

Description

With SFC 26 "UPDAT_PI" (update process image), you update the OB1 process image input table (=process image section 0) or a process image input section defined with STEP 7.

If you configured the repeated signaling of I/O access errors for the system process image table update, the selected process image table will be updated constantly by SFC 26.

Otherwise, SFC 26 will only update the process image table when the selected process image section is not updated by the system, in other words:

- When you have not assigned this process image section to an interrupt OB, or
- When you selected process image section 0 and have disabled updating of the OB1 process image section in the configuration.

Note

Each logical address you assign to a section of the process image input table with STEP 7 no longer belongs to the OB1 process image input table.

The updating of the OB1 process image input table and the process image input sections that you assigned to an interrupt OB is not influenced by SFC 26 calls.

Parameter	Declaration	Data Type	Memory Area	Description
PART	INPUT	BYTE	I, Q, M, D, L, constant	Number of the process image input section to be updated. Maximum value range (depends on the CPU): 0 to 15 (0 means OB1 process image, n where 1 ≤n ≤ 15 means process image section n)
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
FLADDR	OUTPUT	WORD	I, Q, M, D, L	Address of the first byte to cause an error if an access error occurred.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	Illegal value for the input parameter PART.
8091	The specified process image section was not defined or is not in the permitted process image table area on the CPU.
8092	The process image section is updated by the system with an OB and you have not configured repeated signaling of all I/O access errors. The process image was not updated by SFC 26 "UPDAT_PI"
80A0	An access error was detected during the updating.

Note

If you use SFC 26 "UPDAT_PI" for process image partitions of DP standard slaves for which you have defined a consistency area larger than 32 bytes, the errror codes from SFC 14 "DPRD_DAT" are also possible.

14.2 Updating the Process Image Output Table with SFC 27 "UPDAT_PO"

Description

With SFC 27 "UPDAT_PO" (update process outputs), you transfer the signal states of the OB1 process image output table (=process image section 0) or a process image section defined with STEP 7 to the output modules.

If you have specified a consistency range for the part process image corresponding data is transferred consistent to the respective peripheral module.

Note

Each logical address you assign to a section of the process image output table with STEP 7 no longer belongs to the OB1 process image output table.

The transfer of the OB1 process image output table and the process image output sections that you assigned to an interrupt OB is not influenced by SFC 27 calls.

Parameter	Declaration	Data Type	Memory Area	Description
PART	INPUT	BYTE	I, Q, M, D, L, constant	Number of the process image output section to be updated. Maximum value range (depending on the CPU): 0 to 15. (0 means OB1 process image, n where $1 \leq n \leq 15$ means process image section n)
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
FLADDR	OUTPUT	WORD	I, Q, M, D, L	Address of the first byte to cause an error if an access error occurred.

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	Illegal value for the input parameter PART.
8091	The specified process image section was not defined or is not in the permitted process image area on the CPU.
80A0	An access error was detected during the updating.

Note

If you use SFC 27 "UPDAT_PO" for process image partitions of DP standard slaves for which you have defined a consistency area larger than 32 bytes, the errror codes from SFC 15 "DPWR_DAT" are also possible.

14.3 Setting a Bit Field in the I/O Area with SFC 79 "SET"

Description

Calling SFC 79 "SET" (set range of outputs) has the following effect:

- The bit field in the peripheral I/O area selected with the parameters N and SA is set.
- The corresponding bits in the process image output table are also set regardless of whether or not they are in a process image section.

The bit field must be the part of the peripheral I/O area assigned to a process image.

If no module is plugged in for part of the selected bit field, SFC 79 still attempts to set the entire bit field. It then returns the appropriate error information in RET_VAL.

Note

When SFC 79 is executed whole bytes are always written to the I/O area.

If the bit field selected with the parameters N and SA does not begin or end at a byte boundary, calling SFC 79 has the following effect:

- The bits in the first and last bytes to be transferred to the peripheral I/O area and that do not belong to the selected bit field contain the value of the corresponding bits in the process image output table. This can lead to unintended reactions such as starting a motor or turning off a cooling system.
- The bits belonging to the selected bit field are set as explained above.

If you assign the value 0 to the N parameter, calling SFC 79 has no effect. If the master control relay is not set, calling SFC 79 has no effect.

Parameter	Declaration	Data Type	Memory Area	Description
N	INPUT	INT	I, Q, M, D, L, constant	Number of bits to be set
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
SA	OUTPUT	POINTER	P	Pointer to the first bit to be set

Error Information

How you evaluate the error information of the parameter RET_VAL is explained in Evaluating Errors with the Output Parameter RET_VAL. This chapter also contains the general error information of the SFCs. SFC 79 does not provide any specific error information with the RET_VAL parameter.

14.4 Resetting a Bit Field in the I/O Area with SFC 80 "RSET"

Description

Calling SFC 80 "RSET" (reset range of outputs) has the following effect:

- The bit field in the peripheral I/O area selected with the parameters N and SA is reset.
- The corresponding bits in the process image output table are also reset regardless of whether or not they are in a process image section.

The bit field must be located in the part of the peripheral I/O area to which a process image is assigned.

If no module is plugged in for part of the selected bit field, SFC 80 still attempts to reset the entire bit field. It then returns the appropriate error information in RET_VAL.

Note

When SFC 80 is executed, whole bytes are written to the peripheral I/O area.

If the bit field selected with the parameters N and SA does not begin or end at a byte boundary, calling SFC 80 has the following effect:

- The bits in the first and last bytes to be transferred to the peripheral I/O area and that do not belong to the selected bit field contain the value of the corresponding bits in the process image output table. This can lead to unintended reactions such as starting a motor or turning off a cooling system.
- The bits belonging to the selected bit field are set as explained above.

If you assign the value 0 to the N parameter, calling SFC 80 has no effect. If the master control relay is not set, calling SFC 80 has no effect.

Parameter	Declaration	Data Type	Memory Area	Description
N	INPUT	INT	I, Q, M, D, L, constant	Number of bits to be reset
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
SA	OUTPUT	POINTER	P	Pointer to the first bit to be reset

Error Information

How you evaluate the error information of the parameter RET_VAL is explained in Evaluating Errors with the Output Parameter RET_VAL. This chapter also contains the general error information of the SFCs. SFC 80 does not provide any specific error information with the RET_VAL parameter.

14.5 Implementing a Sequencer with SFB 32 "DRUM"

Description

SFB 32 "DRUM" implements a sequencer with a maximum of 16 steps. You specify the number of the first step with the DSP parameter and the number of the last step with the LST_STEP parameter.

During each step, all 16 output bits OUT0 to OUT15 and the output parameter OUT_WORD (in which the output bits are collected together) are written. An output bit is assigned either the corresponding bit of the OUT_VAL array that you specify or the output bit is assigned the value of the corresponding output bit of the previous step. Which value is assigned depends on how you set the mask bits in the S_MASK parameter (see following table).

SFB 32 "DRUM" switches to the next step when there is a rising edge at the JOG input compared with the previous SFB call. If the SFB has already reached the last step, a rising edge at JOG sets the variables Q and EOD; DCC has the value 0; and the SFB remains in the last step until 1 is set at the RESET input.

You can also assign parameters so that switching to the next step is time dependent. To do this, you must set the DRUM_EN parameter to 1. The sequencer then switches to the next step when:

- The event bit EVENT is set for the current step and
- The time programmed for the current step has expired.

  This time is the product of the DTBP time base and the time factor valid for the current step (from the S_PRESET array)

Note

The execution time remaining in the current step (DCC) is only reduced when the corresponding event bit EVENT is set.

If a 1 is set at the RESET input when the SFB is called, the sequencer goes to the step you assigned to the DSP input.

Note

If you set a 1 for DRUM_EN, you can achieve the following special situation:

- Purely time-dependent enabling of the steps by selecting EVENT = 1 where DSP ≤ i ≤ LST_STEP.
- Purely event-dependent enabling of the steps using the event bits EVENT by setting 0 at DTBP.

You can also move on to the next step in the sequencer at any time (even if DRUM_EN=1) via the JOG input.

When the block is called for the first time, you must set 1 at the RESET input.

When the sequencer is in the last step (DSC has the value LST_STEP) and when the execution time for this step has expired, outputs Q and EOD are set and the SFB remains in the last step until you set 1 at the RESET input.

A DRUM timer runs only in the STARTUP and RUN modes.

The operating system resets SFB 32 "DRUM" during a cold restart but not during a warm restart. If you want to initialize SFB 32 "DRUM" after a warm restart, call it with RESET = 1 in OB100.

Parameter	Declaration	Data Type	Memory Area	Description
RESET	INPUT	BOOL	I, Q, M, D, L, constant	Signal level 1 resets the sequencer. When calling the block for the first time, you must set RESET to 1.
JOG	INPUT	BOOL	I, Q, M, D, L, constant	A rising edge (compared to the last SFB call) switches the sequencer to the next step if it is not yet in the last step. The next step is enabled depending on the value you assign to DRUM_EN.
DRUM_EN	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter that specifies whether time-dependent switching to the next step is possible (1: time-dependent switching possible)
LST_STEP	INPUT	BYTE	I, Q, M, D, L, constant	Number of the last step; possible values: 1 to 16
EVENT, 1 ≤ i ≤ 16	INPUT	BOOL	I, Q, M, D, L, constant	Event bit number i (belongs to step i)
OUTj,0 ≤ j ≤ 15	OUTPUT	BOOL	I, Q, M, D, L	Output bit number j (identical to the bit number j of OUT_WORD)
Q	OUTPUT	BOOL	I, Q, M, D, L	Status parameter that specifies whether the execution time you selected for the last step has expired.
OUT_WORD	OUTPUT	WORD	I, Q, M, D, L, P	Output bits collected together in a variable
ERR_CODE	OUTPUT	WORD	I, Q, M, D, L, P	If an error occurs during execution of the SFB, ERR_CODE contains the error information.

Parameter	Declaration	Data Type	Memory Area	Description
JOG_HIS	VAR	BOOL	I, Q, M, D, L, constant	(No relevance for the user: JOG input parameter of the previous SFB call)
EOD	VAR	BOOL	I, Q, M, D, L, constant	Identical to the output parameter Q
DSP	VAR	BYTE	I, Q, M, D, L, P, constant	Number of the first step; possible values: 1 to 16
DSC	VAR	BYTE	I, Q, M, D, L, P, constant	Number of the current step
DCC	VAR	DWORD	I, Q, M, D, L, P, constant	The execution time still remaining in the current step in ms (only relevant if DRUM_EN = 1 and the corresponding event bit is set to = 1)
DTBP	VAR	WORD	I, Q, M, D, L, P, constant	The time base valid for all steps in ms
PREV_TIME	VAR	DWORD	I, Q, M, D, L, constant	(Not relevant for the user: system time of the previous SFB call)
S_PRESET	VAR	ARRAY of WORD	I, Q, M, D, L, constant	One-dimensional array with the time factor for each step. A sensible selection of the indices would be: [1 to 16].In this case, S_PRESET [x] has the time factor of step x.
OUT_VAL	VAR	ARRAY of BOOL	I, Q, M, D, L, constant	Two-dimensional array with the values output in each step if they have not been masked out using S_MASK. A sensible selection for the indices would be: [1 to 16, 0 to 15]. In this case, OUT_VAL [x, y] has the value assigned to the output bit OUTy in step x.
S_MASK	VAR	ARRAY of BOOL	I, Q, M, D, L, constant	Two-dimensional array with the mask bits for each step. A sensible selection of the indices would be: [1 to 16, 0 to 15]. In this case, S_MASK [x, y] contains the mask bit for the y-th value to be output in step x. Meaning of the mask bits: • 0: The value of the previous step is assigned to the corresponding output bit. • 1: The corresponding value from OUT_VAL is assigned to the corresponding output bit.

Error Information

If one of the conditions listed in the following table occurs, SFB 32 "DRUM" remains in its current status and the ERR_CODE output is set.

ERR_CODE (W#16#...)	Explanation
0000	No error
8081	Illegal value for LST_STEP
8082	Illegal value for DSC
8083	Illegal value for DSP
8084	The product DCC = DTBP * S_PRESET[DSC] exceeds the value 2**32-1 (approximately 24.86 days)

15 System Functions for Addressing Modules

15.1 Querying the Logical Base Address of a Module with SFC 5 "GADR_LGC"

Description

Based on the channel of a signal module, the corresponding module slot and the offset user data address area of the module are known. With SFC 5 "GARD_LGC" (convert geographical address to logical address), you can obtain the corresponding logical address of the module, i.e. the least input or output address.

Parameter	Declaration	Data Type	Memory Area	Description
SUBNETID	INPUT	BYTE	I, Q, M, D, L, constant	Area identifier: • 0, if the slot is in one of the racks 0 (central rack) or 1 to 21 (expansion rack). • DP master ID of the corresponding distributed I/O system if the slot is in a distributed I/O device.
RACK	INPUT	WORD	I, Q, M, D, L, constant	• Number of the rack if the area identifier is 0. • Station number of the distributed I/O device if the area identifier > 0.
SLOT	INPUT	WORD	I, Q, M, D, L, constant	Slot number
SUBSLOT	INPUT	BYTE	I, Q, M, D, L, constant	Submodule slot (if no submodule can be plugged in, 0 must be specified here)
SUBADDR	INPUT	WORD	I, Q, M, D, L, constant	Offset in the user data address area of the module
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
IOID	OUTPUT	BYTE	I, Q, M, D, L	Area identifier: B#16#54: Peripheral input (PI) B#16#55: Peripheral output (PQ) In case of a mixed module, the SFC supplies the area identifier of the lower address. If the addresses are equal the SFC supplies the identifier B#16#54.
LADDR	OUTPUT	WORD	I, Q, M, D, L	Logical base address of the module

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8094	No subnet was configured with the specified SUBNETID.
8095	Illegal value for the RACK parameter.
8096	Illegal value for the SLOT parameter.
8097	Illegal value for the SUBSLOT parameter.
8098	Illegal value for the SUBADDR parameter.
8099	The slot is not configured.
809A	The subaddress of the selected slot is not configured.

15.2 Querying the Module Slot Belonging to a Logical Address with SFC 49 "LGC_GADR"

Description

With SFC 49 "LGC_GADR" (convert logical address to geographical address), you obtain the module slot belonging to a logical address and the offset in the user data address area of the module.

Parameter	Declaration	Data Type	Memory Area	Description
IOID	INPUT	BYTE	I, Q, M, D, L, constant	ID of the address area: • B#16#00: Bit 15 of LADDR specifies whether an input (Bit15=0) or output address (Bit 15=1) exists. • B#16#54 = Peripheral input (PI) • B#16#55 = Peripheral output (PO) If the module is a mixed module, specify the area ID of the lowest address. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical address. With mixed modules, specify the lower of the two addresses.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
AREA	OUTPUT	BYTE	I, Q, M, D, L	Area ID: this specifies how the remaining output parameters must be interpreted.
RACK	OUTPUT	WORD	I, Q, M, D, L	Rack number
SLOT	OUTPUT	WORD	I, Q, M, D, L	Slot number
SUBADDR	OUTPUT	WORD	I, Q, M, D, L	Offset in the user data address area of the corresponding module.

Output Parameter AREA

The output parameter AREA specifies how the output parameters RACK, SLOT and SUBADDR must be interpreted (see following table).

Value of AREA	System	Meaning of RACK, SLOT and SUBADDR
0	S7-400	RACK : Module rack no. SLOT : Slot no. SUBADDR : Difference between logical address and logical base address
1	S7-300	RACK: Module rack no. SLOT : Slot no. SUBADDR : Difference between logical address and logical base address
2	DP	RACK: (low byte) Station number RACK : (high byte) DP Master system ID SLOT : Slot no. in the station SUBADDR : Offset in the user data address area of the corresponding module.
3	S5 P area	RACK: Module rack no. SLOT : Slot no. of the adaptation capsule SUBADDR : Address in the S5 x range
4	S5 O area	RACK: Module rack no. SLOT : Slot no. of the adaptation capsule SUBADDR : Address in the S5 x range
5	S5 IM3 area	RACK: Module rack no. SLOT : Slot no. of the adaptation capsule SUBADDR : Address in the S5 x range
6	S5 IM4 area	RACK: Module rack no. SLOT : Slot no. of the adaptation capsule SUBADDR : Address in the S5 x range

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	Specified logical address invalid or illegal value for the IOID parameter

15.3 Querying all Logical Addresses of a Module with SFC 50 "RD_LGADR"

Description

You start with one logical address of a module. With SFC 50 "RD_LGADR" (read module logical addresses), you obtain all the declared logical addresses of this module. You have already assigned addresses to modules previously with STEP 7. SFC 50 enters the logical addresses obtained in the field PEADDR or in the field PAADDR in ascending order.

Parameter	Declaration	Data Type	Memory Area	Description
IOID	INPUT	BYTE	I, Q, M, D, L, constant	Area identifier: • B#16#00: Bit15 of LADDR specifies whether an input (Bit15=0) or output address (Bit15=1) exists. • B#16#54: peripheral input (PI) • B#16#55: peripheral output (PQ)
LADDR	INPUT	WORD	I, Q, M, D, L, constant	One logical address
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
PEADDR	OUTPUT	ANY	I, Q, M, D, L	Field for the PI addresses, field elements must be of the data type WORD.
PECOUNT	OUTPUT	INT	I, Q, M, D, L	Number of returned PI addresses
PAADDR	OUTPUT	ANY	I, Q, M, D, L	Field for the PQ addresses, field must be of the data type WORD.
PACOUNT	OUTPUT	INT	I, Q, M, D, L	Number of returned PQ addresses

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	Specified logical address invalid or illegal value for the IOID parameter.
80A0	Error in the output parameter PEADDR: The data type of the field elements is not WORD.
80A1	Error in the output parameter PAADDR: The data type of the field elements is not WORD.
80A2	Error in the output parameter PEADDR: The specified field could not accommodate all the logical addresses.
80A3	Error in the output parameter PAADDR: The specified field could not accommodate all the logical addresses.

16 SFCs for Distributed I/Os

16.1 Triggering a Hardware Interrupt on the DP Master with SFC 7 "DP_PRAL"

Description

With SFC 7 "DP_PRAL," you trigger a hardware interrupt on the DP master from the user program of an intelligent slave. This interrupt starts OB40 on the DP master.

Using the input parameter AL_INFO, you can identify the cause of the hardware interrupt. This interrupt identifier is transferred to the DP master and you can evaluate the identifier in OB40 (variable OB40_POINT_ADDR).

The requested hardware interrupt is uniquely specified by the input parameters IOID and LADDR. For each configured address area in the transfer memory, you can trigger exactly one hardware interrupt at any time.

How the SFC Operates

SFC 7 "DP_PRAL" operates asynchronously, in other words, it is executed over several SFC calls. You start the hardware interrupt request by calling SFC 7 with REQ=1.

The status of the job is indicated by the output parameters RET_VAL and BUSY, see Meaning of the Parameters REQ, RET_VAL and BUSY with Asynchronous SFCs. The job is completed when execution of OB40 is completed on the DP master.

Note

If you operate the DP slave as a standard slave, the job is completed as soon as the diagnostic frame is obtained by the DP master.

Identifying a Job

The input parameters IOID and LADDR uniquely specify the job.

If you have called SFC 7 "DP_PRAL" on a DP slave and you call this SFC again before the master has acknowledged the requested hardware interrupt, the way in which the SFC reacts depends largely on whether the new call involves the same job: if the parameters IOID and LADDR match a job that is not yet completed, the SFC call is interpreted as a follow-on call regardless of the value of the parameter AL_INFO, and the value W#16#7002 is entered in RET_VAL.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	REQ=1: Hardware interrupt on the DP master belonging to the slave
IOID	INPUT	BYTE	I, Q, M, D, L, constant	Identifier of the address range in the transfer memory (from the point of view of the DP slave): B#16#54= Peripheral input (PI) B#16#55= Peripheral output (PQ) The identifier of a range belonging to a mixed module is the lower of the two addresses. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Start address of the address range in the transfer memory (from the point of view of the DP slave). If this is a range belonging to a mixed module, specify the lower of the two addresses.
AL_INFO	INPUT	DWORD	I, Q, M, D, L, constant	Interrupt ID This is transferred to the OB40 that will be started on the DP master (variable OB40_POINT_ADDR). If you operate the intelligent slave with a remote master, you must evaluate the diagnostic frame on the master. (see ***/70/***)
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains an error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: The triggered hardware interrupt has not yet been acknowledged by the DP master.

Error Information

Error Code (W#16#...)	Explanation
0000	The job was executed without errors.
7000	First call with REQ=0. No hardware interrupt request is active; BUSY has the value 0.
7001	First call with REQ=1. A hardware interrupt request has already been sent to the DP master; BUSY has the value 1.
7002	Interim call (REQ irrelevant): the triggered hardware interrupt has not yet been acknowledged by the DP master; BUSY has the value 1.
8090	Start address of the address range in the transfer memory is incorrect.
8091	Interrupt is blocked (block configured by user)
8093	The parameters IOID and LADDR address a module that is not capable of a hardware interrupt request.
80B5	Call in the DP master not permitted.
80C6	Distributed I/Os not currently available.

16.2 Synchronizing Groups of DP Slaves with SFC 11 "DPSYC_FR"

Description

With SFC 11 "DPSYC_FR," you can synchronize one or more groups of DP slaves.

The function involves sending one of the control commands below or a combination of them to the relevant groups:

- SYNC (simultaneous output and freezing of output states on the DP slaves)
- UNSYNC (cancels the SYNC control command)
- FREEZE (freeze the input states on the DP slaves read in the frozen inputs)
- UNFREEZE (cancels the FREEZE control command)

Note

Note that the control commands SYNC and FREEZE also remain valid when you perform a cold restart.

Requirements

Before you send the control commands listed above, you must assign the DP slaves to groups using STEP 7 (see **/231/**). You must know which DP slave is assigned to which group with which number and know the reactions of the various groups to SYNC/FREEZE.

How the SFC Operates

SFC 11 "DPSYC_FR" is an asynchronous SFC; in other words, its execution takes several SFC calls. You start the job by calling SFC 11 with REQ=1.

The status of the job is indicated by the output parameters RET_VAL and BUSY, also refer to Meaning of the Parameters REQ, RET_VAL and BUSY with Asynchronous SFCs.

Identifying a Job

If you have triggered a SYNC/FREEZE job and called SFC 11 again before the first job was completed, the response of the SFC depends on whether the new call is for the same job. If the input parameters LADDR, GROUP and MODE match, the SFC call is interpreted as a follow-on call.

Writing Outputs of DP Modules

The writing of outputs of DP modules is triggered as follows:

- By transfer commands to the DP I/Os,
- By writing the process image output table to the modules (by the operating system at the end of OB1 or by calling SFC 27 "UPDAT_PO"),
- By calling SFC 15 "DPWR_DAT."

In normal operation, the DP master transfers the output bytes cyclically (within the cycle of the PROFIBUS DP bus) to the outputs of the DP slaves.

If you want to have certain output data (possibly distributed on several slaves) applied to the outputs to the process at exactly the same time, you can send the SYNC command to the relevant DP master using SFC 11 "DPSYC_FR."

What are the Effects of SYNC?

With the SYNC control command, the DP slaves of the selected groups are switched to the Sync mode. In other words, the DP master transfers the current output data and instructs the DP slaves involved to freeze their outputs. With the following output frames, the DP slaves enter the output data in an internal buffer and the state of the outputs remains unchanged.

Following each SYNC control command, the DP slaves of the selected groups apply the output data of their internal buffer to the outputs to the process.

The outputs are only updated cyclically again when you send the UNSYNC control command using SFC 11 "DPSYC_FR."

Note

If the DP slaves of the selected group(s) are not currently connected to the network or have failed when the control command has been sent, they will not be switched to SYNC mode. This information will not be communicated in the return value of the SFC.

Particular Points for ET 200M and ET 200X

In the case of the modules IM 153-1 (Order No. ...-1AA01 and ...-1AA02), BM 141, BM 142, BM 143 and BM 147 it is possible that no values of the outputs are passed to the I/O devices after a general reset, cold restart and an immediate call of the SFC 11. You then have to call the SFC 11 again to achieve synchronization.

Reading Inputs of DP Modules

The input data of the DP modules are read as follows:

- Using load commands to the DP I/Os,
- When the process image input table is updated (by the operating system at the start of OB1 or by calling SFC 26 "UPDAT_PI"),
- By calling SFC 14 "DPRD_DAT."

In normal operation, the DP master receives this input data cyclically (within the cycle of the PROFIBUS DP bus) from its DP slaves and makes them available to the CPU.

If you want to have certain input data (possibly distributed on several slaves) to be read from the process at exactly the same time, send the FREEZE control command to the relevant DP master using SFC 11 "DPSYC_FR."

What are the Effects of FREEZE?

With the FREEZE control command, the DP slaves involved are switched to the Freeze mode, in other words the DP master instructs the DP slaves to freeze the current state of the inputs. It then transfers the frozen data to the input area of the CPU.

Following each FREEZE control command, the DP slaves freeze the state of their inputs again.

The DP master only receives the current state of the inputs cyclically again after you have sent the UNFREEZE control command with SFC 11 "DPSYC_FR."

Note

If the DP slaves of the selected group(s) are not currently connected to the network or have failed when the control command has been sent, they will not be switched to FREEZE mode. This information will not be communicated in the return value of the

Data Consistency

Because SFC 11 "DPSYC_FR" functions are acyclic and can be interrupted by higher priority classes, you should make sure that the process images are consistent with the actual inputs and outputs when using SFC 11 "DPSYC_FR".

This is guaranteed if you keep to the following consistency rules:

- Define suitable process image sections for the "SYNC outputs" and the "FREEZE inputs" (only possible on the S7-400). Call SFC 27 "UPDAT_PO" immediately before the first call for a SYNC job. Call SFC 26 "UPDAT_PI" immediately after the last call for a FREEZE job.
- As an alternative: Use only direct I/O access for outputs involved in a SYNC job and for inputs involved in a FREEZE job. You must not write to these outputs when a SYNC job is active and not read these inputs when a FREEZE job is active.

Using SFC 15 and SFC 14

If you use SFC 15 "DPWR_DAT," this SFC must be completed before you send a SYNC job to the outputs involved.

If you use SFC 14 "DPRD_DAT," this SFC must be completed before you send a FREEZE job to the inputs involved.

SFC 11 "DPSYC_FR" and Startup

The user alone must take responsibility for sending the SYNC and FREEZE control commands in the startup OBs.

If you want the outputs of one or more groups to be in the Sync mode when the user program starts, you must initialize these outputs during startup and execute SFC 11 "DPSYC_FR" with the SYNC control command completely.

If you want the inputs of one or more groups to be in the FREEZE mode when the user program starts, you must execute SFC 11 "DPSYC_FR" with the FREEZE control command completely for these inputs during startup.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	Level-triggered control parameter REQ=1: trigger SYNC/FREEZE job
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical address of the DP master
GROUP	INPUT	BYTE	I, Q, M, D, L, constant	Group selection Bit 0 = 1: group 1 selected Bit 1 = 1: group 2 selected : Bit 7 = 1: group 8 selected You can select several groups per job. The value B#16#0 is invalid.

Parameter	Declaration	Data Type	Memory Area	Description
MODE	INPUT	BYTE	I, Q, M, D, L, constant	Job ID (coding complying with EN 50 170 Volume 2, PROFIBUS) Bit 0: reserved (value 0) Bit 1: reserved (value 0) Bit 2 = 1: UNFREEZE is executed = 0: no meaning Bit 3 = 1: FREEZE is executed = 0: no meaning Bit 4 = 1: UNSYNC is executed = 0: no meaning Bit 5 = 1: SYNC is executed = 0: no meaning Bit 6: reserved (value 0) Bit 7: reserved (value 0) Possible values: • with exactly one ID per job: - B#16#04 (UNFREEZE) - B#16#08 (FREEZE) - B#16#10 (UNSYNC) - B#16#20 (SYNC) • with more than one ID per job: - B#16#14 (UNSYNC, UNFREEZE) - B#16#18 (UNSYNC, FREEZE) - B#16#24 (SYNC, UNFREEZE) - B#16#28 (SYNC, FREEZE)
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code. You must evaluate RET_VAL each time after the block has been executed.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: The SYNC/FREEZE job is not yet completed.

Error Information

Error Code (W#16#...)	Explanation
0000	The job was completed without errors.
7000	First call with REQ=0. The job specified with LADDR, GROUP and MODE is not active; BUSY has value 0.
7001	First call with REQ=1. The job specified with LADDR, GROUP and MODE was triggered; BUSY has value 1.
7002	Interim call (REQ irrelevant). The activated SYNC /FREEZE job is still active; BUSY has value 1.
8090	The module selected with LADDR is not a DP master.
8093	This SFC is not permitted for the module selected with LADDR (configuration or version of the DP master).
8094	Bad GROUP parameter
8095	Bad MODE parameter
80B0	The group selected with GROUP is not configured.
80B1	The group selected with GROUP is not assigned to this CPU.
80B2	The SYNC job specified with MODE is not permitted on the group selected with GROUP.
80B3	The FREEZE job specified with MODE is not permitted on the group selected with GROUP.
80C2	Temporary shortage of resources on the DP master: The DP master is currently processing the maximum number of jobs for a CPU.
80C3	This SYNC /UNSYNC job cannot be activated at present since only one SYNC/UNSYNC job can be triggered at any one time. Check your user program.
80C4	This FREEZE/UNFREEZE job cannot be activated at present since only one FREEZE-/UNFREEZE job can be triggered at any one time. Check your user program.
80C5	Distributed I/Os not accessible: failure of a DP subsystem
80C6	Job aborted due to I/O disconnection by CPU
80C7	Job aborted due to warm or cold restart on the DP master

16.3 Deactivating and Activating DP Slaves with SFC 12 "D_ACT_DP"

Description

With SFC 12 "D_ACT_DP, you can specifically deactivate and reactivate configured DP slaves. In addition, you can determine whether each assigned DP slave is currently activated or deactivated.

SFC 12 cannot be used on PROFIBUS PA field devices which are connected by a DP/PA link to a DP master system.

Purpose

If you configure DP slaves in a CPU which are not actually present or not currently required, the CPU will nevertheless continue to access these DP slaves at regular intervals. After the slaves are deactivated, further CPU accessing will stop. In this way, the fastest possible DP bus cycle can be achieved and the corresponding error events no longer occur.

Application Examples

From a machine builder's point of view, there are numerous device options possible in series production of machines. However, each delivered machine includes only one combination of selected options.

Every one of these possible machine options is configured as a DP slave by the manufacturer in order to create and maintain a common user program having all possible options. With SFC 12, you can deactivate all DP slaves which are not present at machine startup.

A similar situation exists for machine tools having numerous tooling options available but actually using only a few of them at any given time. These tools are implemented as DP slaves. With SFC 12, the user program activates the tools currently needed and deactivates those required later.

How the SFC Operates

SFC 12 "D_ACT_DP" operates asynchronously, in other words, it is executed over several SFC calls. You start the request by calling SFC 12 with REQ=1.

The status of the job is indicated by the output parameters RET_VAL and BUSY; see also Section 2.2 of the Ref. HB "System Software for S7–300/400 System /- and Standard Functions."

Identifying a Job

If you have started a deactivation or activation job and you call SFC 12 again before the job is completed, the way in which the SFC reacts depends largely on whether the new call involves the same job: If the parameter LADDR matches, the SFC call is interpreted as a follow-on call.

Deactivating DP Slaves

When you deactivate a DP slave with SFC 12, its process outputs are set to the configured substitute values or to 0 (secure state). The assigned DP master does not continue to address this DP slave. Deactivated DP slaves are not identified as faulty or missing by the error LEDs on the DP master or CPU.

The process image of the inputs of deactivated DP slaves is updated with 0, that is, it is handled just as for failed DP slaves.

If you are using your program to directly access the user data of a previously deactivated DP slave, the I/O access error OB (OB 122) is called, and the corresponding start event is entered in the diagnostic buffer. If you attempt to access a deactivated DP slave with SFC (i.e. SFC 59 “RD_REC”), you receive the error information in RET_VAL as for an unavailable DP slave.

Deactivating a DP slave does not start the program error OB (OB 85), even if its inputs or outputs belong to the system-side process image to be updated. No entry is made in the diagnostic buffer.

Deactivating a DP slave does not start the rack failure OB (OB 86), and the operating system also does not make an entry in the diagnostic buffer.

If a DP station fails after you have deactivated it with SFC 12, the operating system does not detect the failure. As a result, there is no subsequent start of OB86 or diagnostic buffer entry. The station failure is detected only after the station has been reactivated and indicated in RET_VAL.

If you wish to deactivate DP slaves functioning as transmitters in cross communication, we recommend that you first deactivate the receivers (listeners) that detect which input data the transmitter is transferring to its DP master. Deactivate the transmitter only after you have performed this step.

Activating DP Slaves

When you reactivate a DP slave with SFC 12, it is configured and assigned parameters by the designated DP master (as with the return of a failed station). This activation is completed when the slave is able to transfer user data.

Activating a DP slave does not start the program error OB (OB85), even if its inputs or outputs belong to the system-side process image to be updated. An entry in the diagnostic buffer is also not made.

Activating a DP slave does not start the rack failure OB (OB86), and the operating system also does not make an entry in the diagnostic buffer.

If you attempt to use SFC 12 to activate a slave which has been deactivated and is physically separated from the DP bus, the LED "DP-BUSF" on the CPU will blink for about one minute. After this monitoring period has expired, the SFC returns the error message W#16#80A2 and turns off the LED. The slave remains deactivated. If the slave is reconnected to the DP bus at a later time, it must be reactivated with SFC 12.

Note

Activating a DP slave may be time-consuming. Therefore, if you wish to cancel a current activation job, start SFC 12 again with the same value for LADDR and MODE = 2. Repeat the call of SFC 12 until successful cancellation of the activation is indicated by RET_VAL = 0.

If you wish to activate DP slaves which take part in the cross communication, we recommend that you first activate the transmitters and then the receivers (listeners).

CPU Startup

Depending on the startup mode, the CPU operating system behaves as follows regarding DP slaves:

- In the startup modes cold and warm restart, slaves are activated automatically.
- In the hot restart mode, the slave activation status remains unchanged, that is, activated slaves remain activated and deactivated slaves remain deactivated.

After the CPU start-up, the CPU cyclically attempts to contact all configured and not deactivated slaves that are either not present or not responding.

Note

The startup OBs do not support the call of SFC 12.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant.	Level-triggered control parameter REQ=1: execute activation or deactivation
MODE	INPUT	BYTE	I, Q, M, D, L, constant	Job ID. Possible values: • 0: Request information on whether the addressed DP slave is activated or deactivated • 1: Activate the DP slave • 2: Deactivate the DP slave
LADDR	INPUT	WORD	I, Q, M, D, L, Const.	Any logical address of the DP Slave
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is processed, the return value contains an error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	Active code: • BUSY=1: The job is still active. • BUSY=0: The job was terminated.

Error Information

Error code (W#16#...)	Explanation
0000	The job was completed without errors.
0001	The DP slave is active. (this error code is possible only with MODE = 0.)
0002	The DP slave is deactivated (this error code is possible only with MODE = 0.)
7000	First call with REQ=0. The job specified with LADDR is not active; BUSY has value 0.
7001	First call with REQ=1. The job specified with LADDR was triggered; BUSY has value 1.
7002	Interim call (REQ irrelevant). The activated job is still active; BUSY has value 1.
8090	• You have not configured a module with the address specified in LADDR . • You operate your CPU as I-Slave and you have specified in LADDR an address of this I-Slave.
8092	For the addressed DP slave no activation job is processed at the present (this error code is possible only with MODE = -1.)
8093	No DP slave is assigned to the address stated in LADDR (no projection submitted.), or the parameter MODE is not known
80A1	The addressed DP slave could not be parameterized (this error code is possible only with MODE = 1.) Note: The CPU supplies this information only if the activated slave fails again during parameterization. If parameterization of a single module was unsuccessful the SFC returns the error information W#16#0000
80A2	The addressed DP slave does not return an acknowledgement.
80A3	The DP Master concerned does not support this function.
80A4	The CPU does not support this function for external DP masters.
80A6	Slot error in the DP Slave; user data access not possible (this error code is only available for MODE=1). Note: The SFC returns this error information only if the active slave fails after parameterization and before the SFC ends. If only a single module is unavailable the SFC returns the error information W#16#0000.
80C1	SFC12 was started and continued with another logical address (this error code is only available for MODE=1).
80C3	Temporary resource error: The CPU is currently processing the maximum possible activation and deactivation jobs. (this error code is possible only with MODE = 1 and MODE = 2.)

16.4 Reading Diagnostic Data of a DP Slave with SFC 13 "DPNRM_DG" (Slave Diagnostics)

Slave Diagnostics

Each DP slave provides slave diagnostic data structured in accordance with EN 50 170 Volume 2, PROFIBUS. To read out this diagnostic data, you require SFC 13 "DPNRM_DG."

Refer to the following table for the basic structure of the slave diagnostic data and to the manuals of the DP slaves for further information.

Byte	Meaning
0	Station status 1
1	Station status 2
2	Station status 3
3	Master station number
4	Vendor ID (high byte)
5	Vendor ID (low byte)
6 ...	Further slave-specific diagnostic information

Description

With SFC 13 "DPNRM_DG" (read diagnostic data of a DP slave), you read the current diagnostic data of a DP slave in the format specified by EN 50 170 Volume 2, PROFIBUS. The data that has been read is entered in the destination area indicated by RECORD following error-free data transfer.

You start the read job by assigning 1 to the input parameter REQ in the SFC 13 call.

Function

The read job is executed asynchronously, in other words it requires several SFC 13 calls. The status of the job is indicated by the output parameters RET_VAL and BUSY, also refer to Meaning of the Parameters REQ, RET_VAL and BUSY with Asynchronous SFCs.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	REQ=1: Read request
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Configured diagnostic address of the DP slave. **Note:** Addresses have to be entered in hexadecimal format. For example, diagnostic address 1022 means: LADDR:=W#16#3FE.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code. If no error occurred, the length of the data actually transferred is entered in RET_VAL.
RECORD	OUTPUT	ANY	I, Q, M, D, L	Destination area for the diagnostic data that were read. Only the BYTE data type is permitted. The minimum length of the data record to be read or the destination area is 6. The maximum length of the data record to be sent is 240. Standard slaves can provide more than 240 bytes of diagnostic data up to a maximum of 244 bytes. In this case, the first 240 bytes are transferred to the destination area and the overflow bit is set in the data.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: The read job is not yet completed.

Input Parameter RECORD

The CPU evaluates the actual length of the diagnostic data that were read as following:

- If the length specified for RECORD is less than the number of data bytes supplied, the data are discarded and a corresponding error code is entered in RET_VAL.
- If the length specified for RECORD is greater than or equal to the number of supplied data bytes, the data are accepted in the destination area and the actual length is entered in RET_VAL as a positive value.

Note

You must make sure that the actual parameters of RECORD match in all calls belonging to a job.

A job is uniquely identified by the LADDR input parameter.

Standard Slaves with more than 240 Bytes of Diagnostic Data

With standard slaves on which the number of standard diagnostic data is between 241 and 244 bytes, note the following points:

- If the length specified for RECORD is less than 240 bytes, the data are discarded and the corresponding error information is entered in RET_VAL.
- If the length specified for RECORD is greater than or equal to 240 bytes, the first 240 bytes of the standard diagnostic data are transferred to the destination area and the overflow bit is set in the data.

Output Parameter RET_VAL

- If an error occurs while the function is being executed, the return value contains an error code.
- If no error occurs during the data transfer, RET_VAL contains the length of the data read in bytes as a positive number.

Note

The amount of data read in a DP slave depends on its diagnostic status.

Error Information

How you evaluate the error information of the RET_VAL parameter is explained in Chapter 2. This chapter also contains the general error information for the SFCs. The error information specific to SFC 13 is a subset of the error information for SFC 59 "RD_REC," see Reading a Data Record with SFC 59 "RD_REC"

System Resources for S7-400

When SFC 13 "DPNRM_DG" is called for a job that is not currently being processed, resources of the CPU (memory space) are occupied on the S7-400. You can call SFC 13 in quick succession for several DP slaves providing that you do not exceed the maximum number of "simultaneously" active SFC 13 jobs for your CPU. You will find the maximum number of such jobs in /**101**/.

If you activate several jobs "simultaneously," all the jobs will be executed without interfering with each other.

If you reach the limits of the system resources, this is indicated in RET_VAL. In this case, repeat the job.

16.5 Reading Consistent Data of a DP Standard Slave with SFC 14 "DPRD_DAT"

Data Consistency

Refer to the section Overview of S7 Communication and S7 Basic Communication – Data consistency .

Purpose of SFC 14

You require SFC 14 "DPRD_DAT" because you can only read out a maximum of four continuous bytes using load instructions that access the I/Os or the process image input table.

Note

If required you can also read consistent data via the process image of the inputs. Refer to the technical data to find out whether the CPU supports this functionality.

Description

With SFC 14 "DPRD_DAT" (read consistent data of a DP standard slave), you read the consistent data of a DP standard slave, with the maximum length being fixed for each specific CPU. You will find the maximum length in the technical data of your CPU. If no error occurred during the data transfer, the data that have been read are entered in the destination area identified by RECORD.

The destination area must have the same length as configured for the selected module with STEP 7.

If you read from a DP standard slave with a modular design or with several DP identifiers, you can only access the data of one module/DP identifier per SFC 14 call specifying the configured start address.

Parameter	Declaration	Data Type	Memory Area	Description
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Configured start address from the I area of the module from which the data will be read. **Note:** Addresses have to be entered in hexadecimal format. For example, diagnostic address 100 means: LADDR:=W#16#64.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code.
RECORD	OUTPUT	ANY	I, Q, M, D, L	Destination area for the user data that were read. This must be exactly as long as you configured for the selected module with STEP 7. Only the data type BYTE is permitted.

Error Information

Note

If you access DPV1 slaves, error information from these slaves can be forwarded from the DP master to the SFC. For details on this error information, see Receiving an Interrupt from a DP Slave with SFB 54 "RALRM" STATUS[3].

Error Code (W#16#...)	Explanation
0000	No error occurred.
8090	• You have not configured a module for the specified logical base address or • you have ignored the restriction concerning the length of consistent data or • you have not entered the start address in the LADDR parameter in hexadecimal format.
8092	A type other than BYTE is specified in the ANY reference.
8093	No DP module from which you can read consistent data exists at the logical address specified in LADDR.
80A0	Access error detected while the I/O devices were being accessed.
80B0	Slave failure on external DP interface module.
80B1	The length of the specified destination area is not identical to the user data length configured with STEP 7.
80B2	System error with external DP interface module.
80B3	System error with external DP interface module.
80C0	The data haven't yet been read by the module.
80C2	System error with external DP interface module.
80Fx	System error with external DP interface module.
87xy	System error with external DP interface module.
808x	System error with external DP interface module.

16.6 Writing Consistent Data to a DP Standard Slave with SFC 15 "DPWR_DAT"

Data Consistency

Refer to the section: Overview of S7 Communication and S7 Basic Communication – Data consistency .

Purpose of SFC 15

You require SFC 15 "DPWR_DAT" because you can only write a maximum of four continuous bytes using the transfer instructions that access the I/Os or the process image input table.

Note

If required you can also write consistent data via the process image of the inputs. Refer to the technical data to find out whether the CPU supports this functionality.

Description

With SFC 15 "DPWR_DAT" (write consistent data to a DP standard slave), you transfer the data in RECORD consistently to the addressed DP standard slave and, if required, to the process image (namely if you have configured the respective address area of the DP standard slave as consistency range in a process image). The maximum length of the data to be transferred is fixed for each specific CPU. You will find this information in the technical data for your CPU. The data is transferred synchronously, in other words, on completion of the SFC, the write job is also completed.

The source area must have the same length as you configured for the selected module with STEP 7.

If the DP standard slave has a modular design, you can only access one module of the DP slave.

Parameter	Declaration	Data Type	Memory Area	Description
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Configured start address from the process mage output area of the module to which the data will be written. **Note:** Addresses have to be entered in hexadecimal format. For example, diagnostic address 100 means: LADDR:=W#16#64.
RECORD	INPUT	ANY	I, Q, M, D, L	Source area for the user data to be written. This must be exactly as long as you configured for the selected module with STEP 7. Only the BYTE data type is permitted.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is active, the return value contains an error code.

Error Information

Note

If you access DPV1 slaves, error information from these slaves can be forwarded from the DP master to the SFC. For details on this error information, see Receiving an Interrupt from a DP Slave with SFB 54 "RALRM" STATUS[3].

Error Code (W#16#...)	Explanation
0000	No error occurred.
808x	System error with external DP interface module.
8090	• You have not configured a module for the specified logical base address or • you have ignored the restriction concerning the length of consistent data or • you have not entered the start address in the LADDR parameter in hexadecimal format.
8092	A type other than BYTE is specified in the ANY reference.
8093	No DP module to which you can write consistent data exists at the logical address specified in LADDR.
80A1	Access error detected while I/O devices were being accessed.
80B0	Slave failure on external DP interface module.
80B1	The length of the specified source area is not identical to the user data length configured with STEP 7.
80B2	System error with external DP interface module.
80B3	System error with external DP interface module.
80C1	The data of the previous write job on the module have not yet been processed by the module.
80C2	System error with external DP interface module.
80Fx	System error with external DP interface module.
85xy	System error with external DP interface module.

17 SFCs for Global Data Communication

17.1 Sending a GD Packet with SFC 60 "GD_SND"

Description

With SFC 60 "GD_SND" (global data send), the data of a GD packet are collected and then sent on the path specified in the GD packet. The GD packet must already have been configured with STEP 7.

SFC 60 "GD_SND" can be called at any point in the user program.

The scan rate and the collection and sending of the data by the system at the cycle checkpoint are not influenced by SFC 60 calls.

Interruptability

SFC 60 "GD_SND" can be interrupted by higher priority classes. It is also possible that SFC 60 is called again for the same GD packet in the higher priority class.

The data are then collected and sent in the higher priority class. When the program returns to the interrupted SFC, this is terminated immediately and the data that have already been collected are discarded.

This procedure means that during the processing of the highest priority class, consistent data are transferred (consistency in the sense defined for global data).

Data Consistency with GD

The following rules apply to the consistency of the data collected from the various memory areas and sent.

The following are consistent:

- The simple data types (bit, byte, word, and double word)
- An array of the data types byte, word, and double word up to a maximum length depending on the specific CPU.

Ensuring Consistency for an Entire GD Packet

A GD packet on the CPU sending the data has a structure that does not automatically guarantee that the collected data are consistent. This is, for example, the case when the packet consists of an array of bytes and the number of bytes exceeds the maximum length for the specific CPU.

If, however, you require consistency for the entire GD packet, follow the procedure below in your program:

- Disable or delay the occurrence of higher priority interrupts and asynchronous errors by calling SFC 39 "DIS_IRT" or SFC 41 "DIS_AIRT."
- Call SFC 60 "GD_SND."
- Enable the higher priority interrupts and asynchronous errors again by calling SFC 40 "EN_IRT" or SFC 42 "EN_AIRT."

Parameter	Declaration	Data Type	Memory Area	Description
CIRCLE_ID	INPUT	BYTE	I, Q, M, D, L, constant	Number of the GD circle in which the GD packet to be sent is located. You specify this number when configuring the global data with STEP 7. Permitted values: 1 to 16. The maximum number of possible GD circles can be found in the technical data of your CPU.
BLOCK_ID	INPUT	BYTE	I, Q, M, D, L, constant	Number of the GD packet to be sent in the selected GD circle. This number is set during configuration of the global data by STEP 7. Permitted values: 1 to 3. The maximum number of possible GD circles can be found in the technical data of your CPU.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8081	The GD packet selected with the parameters CIRCLE_ID and BLOCK_ID is not configured.
8082	Illegal value for the parameters CIRCLE_ID or BLOCK_ID or for both parameters.
8083	An error occurred during the execution of the SFC. The type of error is entered in the variable configured for the status information. This can be evaluated by your program.
8084	The execution of the SFC was terminated prematurely because SFC 60 was called again for the same GD packet in a higher priority class (see “Interruptability”).
8085	An error occurred entering the status information in the configured variable.

Note

Following each SFC 60 call, you should evaluate the corresponding GD packet status and, if necessary, reset it.

17.2 Programmed Acceptance of a Received GD Packet with SFC 61 "GD_RCV"

Description

With SFC 61 "GD_RCV" (global data receive), the data from an incoming GD frame for exactly one GD packet are retrieved and entered in the received GD packet. This must already have been configured with STEP 7.

SFC 61 "GD_SND" can be called at any point in the user program.

The scan rate and the retrieving of the data by the system at the cycle checkpoint are not influenced by SFC 61 calls.

Interruptability

SFC 61 can be interrupted by higher priority classes, however, only so that the data consistency defined for global data remains guaranteed. If the processing of the function is interrupted, it is possible that SFC 61 is called again for the same GD packet in the higher priority class.

The data are then entered in the receive GD packet in the higher priority class. When the program returns to the interrupted SFC, this is terminated immediately.

Data Consistency with GD

The following rules apply to the consistency of the data entered in the various memory areas.

The following are consistent:

- The simple data types (bit, byte, word, and double word)
- An array of the data types byte, word, and double word up to a maximum length specific to the receiving CPU.

Ensuring Consistency for an Entire GD Packet

A GD packet on a receiving CPU has a structure that does not automatically guarantee that its data originate from one and the same frame. This is, for example, the case when the packet consists of three GD elements.

If, however, you require consistency for the entire GD packet, follow the procedure below in your program:

- Disable or delay the occurrence of higher priority interrupts and asynchronous errors by calling SFC 39 "DIS_IRT" or SFC 41 "DIS_AIRT."
- Call SFC 60 "GD_SND."
- Enable the higher priority interrupts and asynchronous errors again by calling SFC 40 "EN_IRT" or SFC 42 "EN_AIRT."

Parameter	Declaration	Data Type	Memory Area	Description
CIRCLE_ID	INPUT	BYTE	I, Q, M, D, L, constant	Number of the GD circle into which the incoming GD packet will be entered. This number is specified during configuration of the global data with STEP 7. Permitted values: 1 to 16. The maximum number of possible GD circles can be found in the technical data for your CPU.
BLOCK_ID	INPUT	BYTE	I, Q, M, D, L, constant	Number of the GD packet in the selected GD circle in which the incoming data will be entered. This number is specified during configuration of the global data by STEP 7. Permitted values: 1 to 3. The maximum number of possible GD circles can be found in the technical data for your CPU.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8081	The GD packet selected with the parameters CIRCLE_ID and BLOCK_ID is not configured.
8082	Illegal value for the parameters CIRCLE_ID or BLOCK_ID or for both parameters.
8083	An error occurred during the execution of the SFC. The type of error is entered in the variable configured for the status information. This can be evaluated by your program.
8084	The execution of the SFC was terminated prematurely because SFC 61 was called again for the same GD packet in a higher priority class (see “Interruptability”).
8085	An error occurred entering the status information in the configured variable.

Note

Following each SFC 61 call, you should evaluate the corresponding GD packet status and, if necessary, reset it.

18 Overview over the S7 Communication and the S7 Basic Communication

18.1 Differences between the Blocks of the S7 Communication and the S7 Basic Communication

Selection Criteria

Apart from global data communication, there are two other methods of exchanging data between CPUs/FMs of SIMATIC S7 programmable controllers:

- Data exchange using communication SFCs for non-configured S7 connections
- Data exchange using communication SFBs for configured S7 connections

Which method you choose, depends on the SIMATIC S7 programmable controller you are using (S7-300, S7-400) and on other parameters for data exchange. The following table contains a list of criteria on which you can base your selection.

Criteria	**Communication SFCs for Non-Configured S7 Connections**	**Communication SFBs for Configured S7 Connections**
Availability of the blocks	S7-300 and S7-400 as SFCs	S7-300 as FBs and FC S7-400 as SFBs and SFC
Communication connections	The connection is not configured. It is established when the SFC is active. The connection either remains established after the data have been transferred or it is terminated parameter-controlled. If a connection cannot be established temporarily, the corresponding job cannot be sent.	Connections are configured permanently in the system configuration.
Change to the STOP mode	If the CPU that initiated the data transfer changes to STOP mode, all the connections it established are terminated.	The connection is maintained in the STOP mode.
Several connections to a partner	At any one time, a maximum of one connection is possible to a communication partner.	You can establish several connections to the same partner.
Address range	Modules can be addressed in the local station or in the MPI subnet	Modules can be addressed on the MPI network, on PROFIBUS or on Industrial Ethernet

Criteria	Communication SFCs for Non-Configured S7 Connections	Communication SFBs for Configured S7 Connections
Number of communication partners	The number of communication partners that can be reached one after the other is not restricted by the connection resources available (see **/70/**, **/101**/). (The connections can be established and terminated again while the program is running.)	The number of simultaneously obtainable communication partners is restricted to the number of connection resources available. It also depends on the CPU being used (see **/70**/, **/101**/).
Maximum user data length	A user data length of 76 bytes is guaranteed.	The maximum transferable user data length depends on the block type (USEND / URCV, GET, etc.) and on the communication partner (S7-300, S7-400 or M7).
Number of variables transferred per block call	You can only transfer one variable.	• S7-300: one variable • S7-400: max. four variables
Classification of the blocks	The SFCs for the S7 Basic Communication are system functions. Therefore, they do not require user memory.	The SFBs/FBs for the S7 Basic Communication are system function blocks. Therefore, they require an instance DB for the actual parameters and the static data.
Dynamic modification of the address parameters	Dynamic modification of the address parameters is possible: on completion of the active job, you can address other communication partners.	S7-300: You can reconfigure the addressing parameters while the block is active. The new parameter are validated when the previous job is closed. S7-400: Dynamic modification of address parameters is not possible: the connection is specified and fixed by the first block call and remains unchanged until the next warm or cold restart.

Realization of the IEC 61131-5 with S7-400

The IEC standard 61131-5 is realized with the following blocks:

- USEND(SFB 8)/URCV(SFB 9)
- BSEND(SFB 12)/BRCV(SFB 13)
- PUT(SFB 15)/GET(SFB 14) corresponds to READ/WRITE
- STATUS(SFB 22)/USTATUS(SFB 23)
- ALARM(SFB 33)
- NOTIFY(SFB 36)
- START (SFB 19), STOP (SFB 20) and RESUME (SFB 21) realizes a call interface for the program control functions.

Realization of the IEC 61131-5 with S7-300

The IEC standard 61131-5 is realized with the following blocks:

- USEND(SFB 8)/URCV(SFB 9)
- BSEND(SFB 12)/BRCV(SFB 13)
- PUT(SFB 15)/GET(SFB 14) corresponds to READ/WRITE

18.2 Data Consistency

Definition

The size of the data area which can be modified simultaneously by concurrent processes is called the consistent data area. Data areas which are larger than the consistent data area can thus be falsified as a whole.

This means that a data area which belongs together and which is larger than consistent data area can consist in part of new and of old consistent data blocks at the same time.

Example

AN inconsistency can arise if a communication block is interrupted, for example, by a hardware interrupt OB with a higher priority. If the user program in this OB now changes the data which have already been processed in part by the communication block, the transferred data originate:

- IN part from the time before the hardware interrupt was processed
- And in part from the time after the hardware interrupt was processed

This means that these data are inconsistent (not coherent).

Effect

If larger packages of data are to be transferred in a consistent form, the transfer may not be interrupted. This can, for example, increase the interrupt reaction time in the CPU.

This means: The higher the quantity of data which have to be transferred absolutely consistently, the longer the interrupt reaction time of a system.

Data Consistency at SIMATIC

If the user program includes a communication function, for example BSEND/BRCV, which accesses common data, access to this data area can, for example, be coordinated by using the parameter "DONE“. The data consistency of the communication areas which are transferred locally with a communication block can therefore be ensured in the user program.

However, in the case of S7 communication functions, for example PUT/GET or write/read via OP communication the size of the consistent data area must already be taken into consideration during the programming or configuration stage, since no communication block is available in the user program of the target device (server) to send synchronized communication data to the user program.

At the S7-300 and C7-300 (exception: CPU 318-2 DP) the communication data are copied consistently into the user memory in blocks of 32 bytes in the cycle checkpoint of the operating system. Data consistency is not guaranteed for larger data areas. If a defined data consistency is required, the communication data in the user program may not exceed 32 bytes (maximum of 8 bytes, depending on the version).

In the S7-400 by contrast the communication data are not processed in the cycle checkpoint, but in fixed time slices during the program cycle. The consistency of a variable is ensured by the system.

These communication areas can then be accessed consistently using the PUT/GET functions or reading/writing of variables, for example by an OP on an OS.

Recommendation

For further information on data consistency, please refer to the information describing individual blocks as well as the manual *Communication with SIMATIC*.

18.3 Overview of the S7 Communication Blocks

Classification

For S7 communication, connection configuration is needed. The integrated communication functions are called up with the SFBs/FBs or an SFC/FC from within the application.

These blocks can be classified in the following categories:

- SFBs/FBs for data exchange
- SFBs for changing the operating status
- SFBs for querying the operating status.
- SFC/FC for querying the connection.

Availability of the S7-300/400 Communication Blocks

- The blocks for the S7-400 are in "Standard Library".
- The loadable blocks for the S7-300 are in the "SIMATIC_NET_CP" library.

To run the S7-300 blocks, you need a SIMATIC NET CP in the S7-300 series. For further informaiton, plese see the related documentation.

SFBs/FBs for Data Exchange

Communication SFBs/FBs for data exchange are used to exchange data between two communication partners. If an SFB exists only on the local module, this is referred to as unilateral data exchange. If an SFB/FB exists on the local as well as on the remote module, this is referred to as a bilateral data exchange.

Blocks S7-400	**Blocks S7-300**	**Description**	**Brief Description**
SFB 8 SFB 9	FB 8 FB 9	USEND URCV	Rapid unacknowledged exchange of data irrespective of the sequential execution of the communication function (URCV) at the communication partner (for example, operational and maintenance messages). This means that the data can be overwritten by more recent data at the communication partner.
SFB 12 SFB 13	FB 12 FB 13	BSEND/ BRCV	Secure transfer of a data block to the communication partner. This means that data transmission is not completed until the receive function (BRCV) in the communication partner has accepted the data.
SFB 14	FB 14	GET	Program-controlled reading of variables without additional communication function in the user program of the communication partner.
SFB 15	FB 15	PUT	Program-controlled writing of variables without additional communication function in the user program of the communication partner.
SFB 16	FB 16	PRINT	Sending data to a printer (S7-400 only)

SFBs for Changing the Operating Status (S7-400 only)

With SFBs for changing the operating status, you can control the operating status of a remote device.

Data exchange with the SFBs for changing the operating status is unilateral.

Block S7-400		Brief Description
SFB 19	START	Trigger the RESTART of an S7/M7-300/400 or C7-300 CPU, if it is in the STOP operating mode.
SFB 20	STOP	STOP of an S7/M7-300/400 or C7-300 CPU, if it is in the RUN, HALT operating mode or in a startup.
SFB 21	RESUME	Trigger resume operation of an S7-400-CPU if it is in STOP mode.

SFBs for Querying the Operating Status

With SFBs for querying the operating status, you can obtain information about the operating status of a remote device.

With SFB "STATUS," data exchange is unilateral while with SFB "USTATUS," it is bilateral.

Block S7-400		Brief Description
SFB 22	STATUS	Supplies the operating state of a communication partner (S7-400-CPU, M7-300/400) on user request.
SFB 23	USTATUS	Receives the operating state of an S7-400-CPU when it changes its operate state, if the corresponding connection attribute (send operating state messages) has been set.

SFC/FCs for Querying Connections

Block S7-400	Block S7-300	Brief Description
SFC 62 CONTROL		Querying the state of a connection which belongs to an instance of an SFB/FB.
	FC 62 C CNTRL	Querying a connection state through the connection ID

Tip:

You can also use SFC 87 C_DIAG to perform a diagnosis of the actual connection state (only for S7-400).

Sample Program

A sample program for S7-400 which shows the use of the SFBs for the S7 communication is supplied with STEP 7. This sample program is called step7\examples\com_sfb. It is included under the sample programs in the path folder "..\STEP7\Examples\ZDT01_10".

Work Memory Requirements of the S7 Communication SFBs/FBs

To function smoothly, the S7 communication SFBs/FBs require a temporary memory area within the CPU work memory dependent on the user data (code area). The size of the occupied memory is shown in the following table:

Block inS7-300		**Required Memory in the Working Memory in Bytes**
FB 8	USEND	Block: 4583 bytes, Instance: 368 bytes
FB 9	URCV	Block: 4880 bytes, Instance: 370 bytes
FB 12	BSEND	Block: 5284 bytes, Instance: 372 bytes
FB 13	BRCV	Block: 5258 bytes, Instance: 374 bytes
FB 14	GET	Block: 4888 bytes, Instance: 336 bytes
FB 15	PUT	Block: 4736 bytes, Instance: 384 bytes
FC 62	C_CNTRL	Block: 546 bytes

Note on Interruption Behavior

In, S7-300, SIMATIC_NET communication blocks may only be called in one priority class.

Block in S7-400		**Required Memory in the Working Memory in Bytes**
SFB 8/ SFB 9	USEND/ URCV	68 + Length of the indicated user data when first called up from SD_1,... SD_4/RD_1,... RD_4
SFB 12/ SFB 13	BSEND/ BRCV	54
SFB 14	GET	88 + Length of the indicated user data when first called up from RD_1,... RD_4
SFB 15	PUT	108 + Length of the indicated user data when first called up from SD_1,... SD_4
SFB 16	PRINT	78 + Length specification of FORMAT + Length of the specified user data when first called up from SD_1,... SD_4
SFB 19	START	52 + Length of the indicated parameter when first called up from PI_NAME and ARG.
SFB 20	STOP	48 + Length of the indicated parameter when first called up from PI_NAME.
SFB 21	RESUME	52 + Length of the indicated parameter when first called up from PI_NAME and ARG.
SFB 22	STATUS	50
SFB 23	USTATUS	50

18.4 Overview of the Blocks for the S7 Basic Communication

Classification of the SFCs for the S7 Basic Communication

Connection configuration is not required for the S7 basic communication. The integrated communication functions are called via SFCs in the user program.

The SFCs are divided into two classes:

- SFCs for exchanging data between an S7 CPU and other modules with communication functionality, if the communication partners belong to the same S7 station (identified by the preceding "I“ for internal).
- SFCs for exchanging data between an S7 CPU and other modules with communication functionality, if the communication partners are connected to a common MPI subnet (identified by the preceding "X“ for external).

Communication with stations in other subnets is not possible with the SFCs for S7 basic communication.

The SFCs for basic communication can be run on all CPUs of the S7-300 and S7-400. With these CPUs, you can also write variables to the CPUs of the S7-200 and read variables from them.

SFCs for External Communication

Block		Brief Description
SFC 65/ SFC 66	X_SEND/ X_RCV	Secure transfer of a data block to a communication partner. This means that data transmission is not completed until the receive function (X_RCV) in the communication partner has accepted the data.
SFC 67	X_GET	Reading a variable of a communication partner without your having to place a corresponding SFC on the communication partner. This functionality is implemented in the communication partner of the operating system.
SFC 68	X_PUT	Writing a variable into a communication partner without your having to place a corresponding SFC on the communication partner. This functionality is implemented in the communication partner of the operating system.
SFC 69	X_ABORT	Aborting of an existing connection without data being transferred. The corresponding connection resources are thus released again on both ends.

SFCs for Internal Communication

Block		Brief Description
SFC 72	I_GET	Reading a variable of a communication partner without your having to place a corresponding SFC on the communication partner. This functionality is implemented in the communication partner of the operating system.
SFC 73	I_PUT	Writing a variable into a communication partner without your having to place a corresponding SFC on the communication partner. This functionality is implemented in the communication partner of the operating system.
SFC 74	I_ABORT	Aborting of an existing connection without data being transferred. The corresponding connection resources are thus released again on both ends.

Example Programs

Two example programs for the SFCs for S7 basic communication are supplied with STEP 7. They are contained in the directories step7\examples\com_SFC1 and step7\examples\com_SFC2.

Maximum User Data Length

The communication SFCs for non-configured S7 connections are integrated on all CPUs of the S7-300 and S7-400.

It is guaranteed that 76 bytes of user data can be transferred with all SFCs (parameter SD or RD).

Connection to the Communication Partner

With the communication SFCs for non-configured S7 connections, the connection is established while the SFC is being executed. Depending on the value you assign to the CONT input parameter, the connection either remains established or is terminated on completion of the data exchange. This means that the communication has the following characteristics:

- The number of communication partners that can be reached one after the other is higher than the number of communication partners that can be reached simultaneously (the number depends on the specific CPU, see **/70/**, **/101/**).
- If no connection can currently be established to a communication partner because the connection resources (on the local CPU or on the communication partner) are all being used, this is indicated in RET_VAL. You must then trigger the job again later at a suitable point in time. There is, however, no guarantee that later connection establishment will be successful. If necessary, check the use of connection resources in your program and use a CPU with more resources.

Existing connections of communication SFBs for configured S7 connections cannot be used by the communication SFCs for non-configured S7 connections.

Once you have triggered a job, the connection established for the job can only be used for this particular job. Other jobs involving the same communication partner can then only be executed after the current job is completed.

Note

If your program includes several jobs involving the same communication partner, make sure that you call the SFCs for which W#16#80C0 is entered in RET_VAL again later at a suitable point in time.

Identifying a Job

If you have triggered a data transfer or a connection abort with one of the communication SFCs for non-configured S7 connections and you call this SFC again before the current transfer is completed, the reaction of the SFC depends on whether the new call involves the same job. The following table explains which input parameters specify a job for every SFC, If the parameters match those of a job that is not yet completed, the SFC call counts as a follow-on call.

Block		Job is identified by
SFC 65	X_SEND	DEST_ID, REQ_ID
SFC 67	X_GET	DEST_ID, VAR_ADDR
SFC 68	X_PUT	DEST_ID, VAR_ADDR
SFC 69	X_ABORT	DEST_ID
SFC 72	I_GET	IOID, LADDR, VAR_ADDR
SFC 73	I_PUT	IOID, LADDR, VAR_ADDR
SFC 74	I_ABORT	IOID, LADDR

Reaction to Interrupts

The communication SFCs for non-configured S7 connections can be interrupted by higher priority OBs. If the same SFC with the identical job is called again by the interrupting OB, this second call is aborted and a corresponding entry made in RET_VAL. The execution of the interrupted SFC is then continued.

Access to the Work Memory of the CPU

Regardless of the number of user data to be transferred, the communication functions of the operating system access the work memory of the CPU in fields of the maximum length, so that the interrupt reaction time is not extended by the use of communication functions.

Depending on how you set the maximum cycle load resulting from communication with STEP 7, the work memory can be accessed several times during the execution of a job by the communication functions of the operating system.

Client Changes to STOP

If the CPU that initiated a job (and therefore established the connection) changes to STOP during a data transfer, all the connections it established are terminated.

Making Program Changes

All parts of your program that immediately affect the calls for communication SFCs for non-configured S7 connections must only be modified in the STOP mode. This includes, in particular, deleting FCs, FBs, or OBs containing calls for communication SFCs for non-configured S7 connections.

After modifying the program, you must perform a warm or cold restart.

Not following these rules can lead to resources remaining assigned and the programmable controller being subsequently in an undefined state.

19 S7 Communication

19.1 Common Parameters of the SFBs/FBs and SFCs/FCs for S7 Communication

Classification

The parameters of communication SFBs/FBs for configured S7 connections can be divided into the following five categories according to their functions:

1. Control parameters are used to activate a block.
2. Addressing parameters are used to address the remote communication partner.
3. Send parameters point to the data areas that are to be sent to the remote partner.
4. Receive parameters point to the data areas where the data received from remote partners will be entered.
5. Status parameters are used to monitor whether the block has completed its task without error or for the analysis of any errors that have occurred.

Control Parameters

Data exchange will only be activated if the appropriate control parameters have a defined signal state (for example are set) when the SFB/FB is called or when the signal state has undergone a specific change since the previous SFB/FB call (for example, positive edge).

Note on S7-300

For the first call, set the parameter REQ to FALSE.

Addressing Parameters

Parameter	Description
ID	Reference to the local connection description (specified by the STEP 7 configuration of the connection).
R_ID	Use the R_ID parameter to specify that a send and a receive SFB belong together: The R_ID parameter must be identical at the SFB/FB on the send end and at the SFB on the receive end. This allows the communication of several SFB/FB pairs via the same logic connection. • R_ID must be specified in the form DW#16#wxyzWXYZ. • The block pairs of a logic connection which are specified in R_ID must be unique for this connection.

The parameter PI_NAME is only described at the relevant SFBs (S7-400 only).

Note

S7-300: You can change the parameters in the addressing parameters ID and R ID while they are active. The new parameters are validated with each new job after the previous job has been closed. Here, you can link multiple FB pairs in one instance.

Tip: You have the following possibilities to save instance DBs and therefore working memory:

1. With variable IDs you can use several connections via **one** data instance block.
2. With variable R_IDs you can define several identities of pairs of send and receive FBs for one job.
3. You can combine case 1 and case 2.

Please observe that the new parameters are valid after the last job is executed. If you activate the send job, the R_ID parameter of the sending and the receiving FB must match.

S7-400: The addressing parameters ID and R ID are evaluated only at the first call of the block (the actual parameters or the predefined values from the instance). The first call therefore specifies the communication relation (connection) with the remote partner until the next warm or cold restart.

Status Parameters

With the status parameters, you monitor whether the block has completed its task correctly or whether it is still active. The status parameters also indicate errors.

Note

The status parameters are valid for one cycle only, namely from the first instruction which follows the SFB/FB call until the next SFB/FB call. As a result, you must evaluate these parameters after each block cycle.

Send and Receive Parameters

If you do not use all send or receive parameters of an SFB/FB, the first unused parameter must be a NIL pointer (see ***/232/***) and the parameters used must be located one after the other and without any gaps.

Note for S7-400

During the first call, the ANY pointer specifies the maximum amount of user data that can be transferred for the job. That is to say, a communication buffer is created in the work memory of the CPU to ensure data consistency. This buffer occupies up to 480 bytes of work memory. We recommend you run the first call in the warm or cold restart OB if the block is not reloaded with the SFB call when the CPU is in RUN mode.

At subsequent calls you can send/receive any amount of data, however, no more than with the first call.

The SFBs BSEND and BRCV are an exception to this rule. With them you can transmit up to 64 Kbytes per job (see Sending Segmented Data with SFB 12 "BSEND" and Receiving Segmented Data with SFB 13 "BRCV").

With SFBs/FBs for bilateral communication:

- The number of the SD_i and RD_i parameters used must match on the send and receive side.
- The data types of the SD_i and DR_i parameters that belong together must match on the send and receive side.
- The amount of data to be sent according to the SD_i parameter must not exceed the area made available by the corresponding RD_i parameter (not valid for BSEND/BRCV).

ERROR = 1 and STATUS = 4 indicate that you have violated the above rules.

This maximum user data length depends on whether the remote partner is an S7-300 or an S7-400.

User Data Size

With the SFBs/FBs, USEND, URCV, GET and PUT, the amount of data to be transmitted must not exceed a maximum user data length. The maximum user data size depends on:

- The block type used and
- The communication partner.

The guaranteed minimum size of the user data for an SFB/FB with 1–4 variables is listed in the following table:

Block	**Partner: S7-300/C7-300**	**Partner: S7-400/M7 M7 to M7**
PUT / GET	160 bytes	400 bytes
USEND / URCV	160 bytes	440 bytes
BSEND / BRCV	32768 bytes	65534 bytes

Further information on the user data size can be found in the technical data of the respective CPU.

Exact User Data Size

If the user data size specified above is insufficient you can determine the maximum byte length of the user data as follows:

1. First read the data block size valid for communication from the following table:

Own CPU	**Remote CPU**	**Data block size in bytes**
S7-300	Any	240 (S7-300)
S7-400	S7-300 / C7-300	240 (S7-400)
S7-400	S7-400 or CPU 318	480
S7-400	M7 module	480
M7 module	M7 module	960

2. Use this value in the following table to read the maximum possible user data length in bytes. It applies for an even lengths of the areas SD_i, RD_i, ADDR_i.

		Number of used parameters SD_i, RD_i, ADDR_i			
Data block size	**SFB/FB**	**1**	**2**	**3**	**4**
240 (S7-300)	PUT/GET/ USEND	160	-	-	-
240 (S7-400)	PUT	222	218	214	210
	GET	212	196	180	164
480	PUT	462	458	454	450
	GET	452	436	420	404
	USEND	452	448	444	440
960	PUT	942	938	934	930
	GET	932	916	900	884
	USEND	932	928	924	920

19.2 Startup Routine of SFBs for Configured S7 Connections

Requirements

In the following description for S7-400, it is assumed that:

- The connection descriptions (SDBs) exist on the modules.
- The configured connections have been established.
- The actual parameter for the ID matches the configured connection ID for each SFB.

Warm Restart and Cold Restart

During a warm and a cold restart all SFBs are set to the NO_INIT status. The actual parameters stored in the instance DBs are not changed.

Warm Restart and Cold Restart with SFBs for Bilateral Data Exchange

In general, the two modules with SFBs for bilateral data exchange do not both go through a warm or cold restart simultaneously. The reaction of the SFB is governed by the rules below:

The receive blocks (SFBs URCV, BRCV) react as follows:

- If the SFB has received a job but has not acknowledged this job at the time of the warm or cold restart, it generates a sequence abort frame (CFB, BRCV) and then immediately branches to the NO_INIT status.
- With SFB BRCV, it is possible that another data segment will be received despite having sent the sequence abort. This will be discarded locally.
- SFB URCV changes to the NO_INIT status immediately.

The send blocks (SFBs USEND, BSEND) react as follows:

- If SFB BSEND has started a job sequence that has not yet been completed, it sends a sequence abort when the warm or cold restart is initiated. It then branches to the NO_INIT status immediately afterwards. An acknowledgement that arrives at a later time is discarded locally.
- If SFB BSEND has already sent or received a sequence abort when the warm or cold restart is requested, it changes immediately to the NO_INIT status.
- In all other cases and whenever the SFB sends only messages (for example, SFB USEND), local processing is aborted and the SFB immediately branches to the NO_INIT status.

Warm Restart and Cold Restart with SFBs for Unilateral Data Exchange

It is assumed that the server on the communication partner is operational after the connections have been established, in other words that it can process jobs or output messages at any time.

SFBs that send out jobs and expect acknowledgements react to a complete restart as follows:

The current processing is aborted and the CFB branches to the NO_INIT status immediately afterwards. If an acknowledgement for the job sent prior to the warm or cold restart arrives later, it is discarded locally.

A new job may have been sent before the acknowledgement of the earlier job is received.

SFBs that output or receive messages react as follows:

- The current processing is aborted and the CFB branches to the NO_INIT status immediately afterwards.
- With SFB USTATUS, messages that arrive during the NO_INIT and DISABLED statuses are discarded locally.

Reaction to a Hot Restart

The SFBs for S7 communication are set to the NO_INIT status only during a warm or cold restart. This means that they react like user function blocks that can be resumed following a hot restart.

Reaction to a Memory Reset

A memory reset always causes all connections to be terminated. Since a warm or cold restart is the only possible startup type for the user program after a memory reset, all SFBs for S7 communications (if they still exist) are set to the NO_INIT status and initialized. Partner blocks in a module whose memory was not reset change to the IDLE, ENABLED or DISABLED statuses as a reaction to the connection being terminated.

19.3 How SFBs React to Problems

The following section describes how SFBs for S7 communication in S7-400 react to problems.

Connection Terminated

The connections allocated to the SFB instances are monitored.

If a connection is terminated, the reaction of the SFB depends on its internal status.

If the break down of the connection is detected while the block is in the IDLE or ENABLED status, the SFB reacts as follows:

- It branches to the ERROR status and outputs the error ID "Communication problems" at the ERROR and STATUS output parameters.
- When it is next called, the block returns to its original status and checks the connection again.

A communication SFB that is not in the IDLE or DISABLED statuses reacts as follows:

- It aborts processing, changes to the ERROR status immediately or at the next block call and outputs the error ID "Communication problems" at the ERROR and STATUS output parameters.
- When it is next called, the block changes to the IDLE, DISABLED or ENABLED status. In the IDLE and ENABLED status the connection is checked again.

This procedure will also be executed if the connection has again been set up in the meantime.

Power Down

A power down with battery backup followed by a restart causes all established connections to be terminated. The points made above therefore apply to all blocks involved.

If there is a power down with battery backup followed by an automatic warm or cold restart, the points made about terminated connections and warm or cold restarts apply.

In the special case of an automatic warm or cold restart without battery backup, where a memory reset is executed automatically after power returns, the SFBs for S7 communications react as described in the section "Startup Routine of the SFBs For S7 Communications."

Reaction to Operating Mode Changes

If the operating mode changes between the STOP, START, RUN, and HOLD statuses, the communication SFB remains in its current status (exception: during a warm or cold restart, it changes to the NO_INIT status). This applies both to SFBs for unilateral as well as SFBs for bilateral communication.

Error Interface to the User Program

If an error occurs during the processing of a communication SFB, it always changes to the ERROR status. At the same time the ERROR output parameter is set to 1 and the corresponding error ID is entered in the STATUS output parameter. You can evaluate this error information in your program.

Examples of possible errors:

- Error when collecting send data.
- Error when copying receive data into the receive areas (for example, attempting to access a DB that does not exist).
- The length of the data area sent does not match the length of the receive area specified in the partner SFB.

19.4 Uncoordinated Sending of Data with SFB 8/FB 8 "USEND"

Description

SFB/FB 8 "USEND" sends data to a remote partner SFB/FB of the type "URCV". The send process is carried out without coordination with the SFB/FB partner. This means that the data transfer is carried out without acknowledgement by the partner SFB/FB.

S7-300: The data is sent on a rising edge at REQ. The parameters R_ID, ID and SD_1 are transferred on each rising edge at REQ. After a job has been completed, you can assign new values to the R_ID, ID and SD_1 parameters.

S7-400: The data is sent on a rising edge at control input REQ. The data to be sent is referenced by the parameters SD_1 to SD_4 but not all four send parameters need to be used.

You must, however, make sure that the areas defined by the parameters SD_1 to SD_4/SD_1 and RD_1 to RD_4/RD_1 (at the corresponding partner SFB/FB "URCV") agree in:

- Number
- Length and
- Data type.

The parameter R_ID must be identical at both SFBs.

Successful completion of the transmission is indicated by the status parameter DONE having the logical value 1.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L,	Control parameter request, activates the data exchange at a rising edge
ID	INPUT	WORD	M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
R_ID	INPUT	DWORD	I, Q, M, D, L, constant	Addressing parameter R_ID, refer to Common Parameters of the SFBs and SFC for the S7 communication
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: 0: Job not started or still running 1: Job has been executed error-free
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information ERROR=1 There is an error. STATUS supplies detailed information on the type of error.
S7-300: SD_1 S7-400: SD_i (1≤ i ≤4)	IN_OUT	ANY	M, D, T, Z I, Q, M, D, T, C	Pointer to the i-th send data area. Only the following data types are permissibleBOOL (not allowed: Bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER. **Note:** If the ANY pointer accesses a DB, the DB must always be specified (for example: P# DB10.DBX5.0 Byte 10).

Error Information

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: new job cannot take effect since previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	4	Error in the send data area pointers SD_i involving data length or data type.
1	10	Access to the local user memory is not possible (for example, access to a deleted DB)
1	12	When the SFB was called • An instance DB was specified that does not belong to SFB "USEND" • No instance DB was specified, but rather a global DB. • No instance DB found (loading new instance DB from PG).
1	18	R_ID exists already in the connection ID.
1	20	Insufficient memory. • H-System: SFB first called while update in progress • S7-300: - Maximum number of parallel jobs/instances exceeded - The instances were overloaded at CPU-RUN - Possible when first called

Data Consistency

S7-300: To ensure data consistency, you can only write to the send area SD_1 again after the current send operation is complete. This is the case when the value of the status parameter DONE changes to 1.

S7-400: When a send operation is activated (rising edge at REQ) the data to be sent from the send area SD_i are copied from the user program. After the block call, you can write to these areas without corrupting the current send data.

Note

The send operation is only complete when the DONE status parameter has the value 1.

19.5 Uncoordinated Receiving of Data with SFB/FB 9 "URCV"

Description

SFB/FB 9 "URCV" receives data asynchronously from a remote partner SFB/FB of the type "USEND" and copies them into the configured receive ranges.

The block is ready to receive then there is a logical 1 at the EN_R input. An active job can be cancelled with EN_R=0.

S7-300: The parameters R_ID, ID and RD_1 are applied with every positive edge on EN_R. After a job has been completed, you can assign new values to the R_ID, ID and RD_1 parameters.

S7-400: The receive data areas are referenced by the parameters RD_1 to RD_4.

You must, however, make sure that the areas defined by the parameters RD_i/RD_1 and SD_i/SD_1 (at the corresponding partner SFB/FB "USEND") agree in:

- Number
- Length and
- Data type.

Successful completion of the copying process indicated at the NDR state parameter by a logical 1.

The parameter R_ID must be identical at both SFBs/FBs.

Parameter	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L, constant	The enabled to receive control parameter signalizes that the unit is ready to receive when the input is set.
ID	INPUT	WORD	M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC of S7 communication
R_ID	INPUT	DWORD	I, Q, M, D, L, constant	Addressing parameter R_ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
NDR	OUTPUT	BOOL	I, Q, M, D, L	NDR status parameter: 0: Job not started or still running 1: Job has been executed error-free

Parameter	Declaration	Data Type	Memory Area	Description
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information ERROR=1 There is an error. STATUS supplies detailed information on the type of error.
S7-300: RD_1 S7-400: RD_i (1≤ i ≤4)	IN_OUT	ANY	M, D, T, Z I, Q, M, D, T, Z	Pointer to the i-th receive data area: Only the following data types are permissible: BOOL (not allowed: Bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER. **Note:** If the ANY pointer accesses a DB, the DB must always be specified (for example: p# DB10.DBX5.0 Byte 10).

Error Information

ERROR	STATUS (Decimal)	Explanation
0	9	Overrun warning: older received data are overwritten by newer received data.
0	11	Warning: the new job is not effective since the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	4	Errors in the receive area pointers RD_i involving the data length or the data type.
1	10	Access to the local user memory not possible (for example, access to a deleted DB)
1	12	When the CFB was called • An instance DB was specified that does not belong to SFB "URCV" • No instance DB was specified, but rather a global DB • No instance DB was found (loading a new instance DB from the PG).
1	18	R_ID already exists in the connection ID.

ERROR	STATUS (Decimal)	Explanation
1	19	The corresponding SFB/FB "USEND" is sending data faster than the SFB/FB "URCV" can copy them to the receive areas ."
1	20	Insufficient memory. • H-System: SFB first called while update in progress • S7-300: - Maximum number of parallel jobs/instances exceeded - The instances were overloaded at CPU-RUN - Possible when first called

Data Consistency

The data are received consistently if you remember the following points:

- S7-300: After the status parameter NDR has changed to the value 1, you must immediately call FB 9 "URCV" again with the value 0 at EN_R. This ensures that the receive area is not overwritten before you have evaluated it.
- Evaluate the receive area (RD_1) completely before you call the block with the value 1 at control input EN_R).

S7-400: After the status parameter NDR has changed to the value 1, there are new receive data in your receive areas (RD_i). A new block call may cause these data to be overwritten with new receive data. If you want to prevent this, you must call SFB 9 "URCV" (such as with cyclic block processing) with the value 0 at EN_R until you have finished processing the receive data.

19.6 Sending Segmented Data with SFB/FB 12 "BSEND"

Description

SFB/FB 12 "BSEND" sends data to a remote partner SFB/FB of the type "BRCV". With this type of data transfer, more data can be transported between the communications partners than is possible with all other communication SFBs/FBs for configured S7 connections, namely up to 32768 bytes for S7-300 65534 bytes for S7-400.

The data area to be transmitted is segmented. Each segment is sent individually to the partner. The last segment is acknowledged by the partner as it is received, independently of the calling up of the corresponding SFB/FB "BRCV".

S7-300: The send job is activated on a rising edge at REQ. The parameters R_ID, ID, SD_1 and LEN are transferred on each positive edge at REQ. After a job has been completed, you can assign new values to the R_ID, ID, SD_1 and LEN parameters. For the transmission of segmented data the block must be called periodically in the user program.

The start address and the maximum length of the data to be sent are specified by SD_1. You can determine the job-specific length of the data field with LEN.

S7-400: The send job is activated after calling the block and when there is a rising edge at the control input REQ. Sending the data from the user memory is carried out asynchronously to the processing of the user program.

The start address and the maximum length of the data to be sent are specified by SD_1. You can determine the job-specific length of the data field with LEN. In this case, LEN replaces the length section of SD_1.

The parameter R_ID must be identical at the two corresponding SFBs/FBs.

If there is a rising edge at control input R, the current data transfer is canceled.

Successful completion of the transfer is indicated by the status parameter DONE having the value 1.

A new send job cannot be processed until the previous send process has been completed if the status parameter DONE or ERROR have the value 1.

Due to the asynchronous data transmission, a new transmission can only be initiated if the previous data have been retrieved by the call of the partner SFB/FB. Until the data are retrieved, the status value 7 (see below) will be given when the SFB/FB "BSEND" is called.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter request, activates the data exchange at a rising edge.
R	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter reset, activates an abort at a rising edge while data are still being exchanged.
ID	INPUT	WORD	M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
R_ID	INPUT	DWORD	I, Q, M, D, L, constant	Addressing parameter R_ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication With a connection via the CP 441 to S5 or auxiliary devices, R_ID contains the address information of the remote device. For further information, refer to the description of the CP 441.
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: 0: Job not started or still running 1: Job has been executed error-free
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information ERROR=1 There is an error. STATUS supplies detailed information on the type of error.

Parameter	Declaration	Data Type	Memory Area	Description
SD_1	IN_OUT	ANY	S7-300: M, D S7-400: I, Q, M, D, T, Z	Pointer to the send area. Only the following data types are permissible: BOOL (not allowed: Bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER. **Note:** If the ANY pointer accesses a DB, the DB must always be specified (for example: p# DB10.DBX5.0 Byte 10).
LEN	IN_OUT	WORD	I, Q, M, D, L	Length of the data field to be sent in bytes.

Error Information

The following table contains all the error information specific to SFB/FB 12 that can be output with the parameters ERROR and STATUS.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: new job is not effective since the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	2	Negative acknowledgement from the partner SFB/FB. The function cannot be executed.
1	3	R_ID is unknown on the connection specified by the ID or the receive block has not yet been called.
1	4	Error in the send area pointer SD_1 involving the data length or the data type or the value 0 was transferred with LEN.
1	5	Reset request was executed.
1	6	Partner SFB/FB is in the DISABLED state (EN_R has the value 0). Also check the input parameters of the BRCV block for consistency with the BSEND block.
1	7	Partner SFB/FB is in the wrong state. The receive block was not called again after the last data transmission.
1	8	Access to remote object in the user memory was rejected: The target area for the corresponding SFB/FB 13 "BRCV" is too small. The corresponding SFB/FB 13 "BRCV" reports ERROR = 1, STATUS = 4.
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	12	When the SFB was called • An instance DB was specified that does not belong to SFB 12 • No instance DB was specified, but rather a global DB. • No instance DB found (loading a new instance DB from the PG).

ERROR	STATUS (Decimal)	Explanation
1	18	R_ID already exists in the connection ID.
1	20	Insufficient memory. • H-System: SFB first called while update in progress • S7-300: - Maximum number of parallel jobs/instances exceeded - The instances were overloaded at CPU-RUN - Possible when first called.

Data Consistency

To ensure data consistency, you can only write to the currently used send area SD_i again after the current send operation is complete. This is the case when the value of the status parameter DONE changes to 1.

19.7 Receiving Segmented Data with SFB/FB 13 "BRCV"

Description

SFB/FB 13 "BRCV" receives data from a remote partner SFB/FB of the type "BSEND". After each received data segment an acknowledgement is sent to the partner SFB/FB and the LEN parameter is updated.

After it has been called and the value 1 is applied at the control input EN_R, the block is ready to receive data. An active job can be cancelled with EN_R=0.

The start address and the maximum length of the receive area is specified by RD_1. The length of the received data field is indicated in LEN.

S7-300: The parameters R_ID, ID and RD_1 are applied with every positive edge on EN_R. After a job has been completed, you can assign new values to the R_ID, ID and RD_1 parameters. For the transmission of segmented data the block must be called periodically in the user program.

S7-400: Receipt of the data from the user memory is carried out asynchronously to the processing of the user program.

The parameter R_ID must be identical at the two corresponding SFBs/FBs.

Error free reception of all the data segments is indicated by the status parameter NDR having the value 1. The received data remain unchanged until SFB/FB 13 is called again with EN_R=1.

If the block is called during asynchronous reception of data, this leads to a warning being output in the STATUS parameter; if the call is made when the value 0 is applied to control input EN_R, reception is terminated and the SFB/FB returns to its initial state.

Parameter	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter enabled to receive, signalizes that the unit is ready to receive when the input is set.
ID	INPUT	WORD	M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
R_ID	INPUT	DWORD	I, Q, M, D, L, constant	Addressing parameter R_ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication With a connection via the CP 441 to S5 or auxiliary devices, R_ID contains the address information of the remote device. For further information, refer to the description of the CP 441.
NDR	OUTPUT	BOOL	I, Q, M, D, L	NDR status parameter: 0: Job has not been started or still active 1: Job was completed successfully.
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information ERROR=1 There is an error. STATUS supplies detailed information on the type of error.
RD_1	IN_OUT	ANY	S7-300: M, D S7-400: I, Q, M, D, T, C	Pointer to the receive area. The length information specifies the maximum length of the block to be received. Only the following data types are permissible:BOOL (not allowed: Bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER. **Note:** If the ANY pointer accesses a DB, the DB must always be specified (for example: P# DB10.DBX5.0 Byte 10).
LEN	IN_OUT	WORD	I, Q, M, D, L	Length of the data already received in bytes.

Error Information

The following table contains all the error information specific to SFB/FB 13 that can be output with the parameters ERROR and STATUS.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: new job is not effective since the previous job is not yet completed.
0	17	Warning: block receiving data asynchronously.
0	25	Data being received. Ther LEN parameter shows the amount of data already received in bytes.
1	1	Communications problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	2	Function cannot be executed.
1	4	Error in the receive area pointer RD_1 regarding the data length or data type. The sent data field is longer than the receive area.
1	5	Reset request received, incomplete transfer.
1	8	Access error in the corresponding SFB/FB 12 "BSEND". After the last valid data segment has been sent, ERROR = 1 and STATUS = 4 is reported.
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	12	When the SFB was called • An instance DB was specified that does not belong to SFB 13 • No instance DB was specified, but rather a global DB. • No instance DB found (loading a new instance DB from the PG).
1	18	R_ID already exists in the connection ID.
1	20	Insufficient memory. • H-System: SFB first called while update in progress • S7-300: - Maximum number of parallel jobs/instances exceeded - The instances were overloaded at CPU-RUN - Possible when first called

Data Consistency

The data are received consistently if you remember the following point:

- Evaluate the last used receive ares (RD_1) completely before you call the block again with the value 1 at control input EN_R).

Special Case for Receiving Data (S7-400 only)

If a receiving CPU with a BRCV block ready to accept data (that is, a call with EN_R = 1 has already been made) goes into STOP mode before the corresponding send block has sent the first data segment for the job, the following will occur:

- The data in the first job after the receiving CPU has gone into STOP mode are fully entered in the receive area.
- The partner SFB "BSEND" receives a positive acknowledgement.
- Any additional BSEND jobs can no longer be accepted by a receiving CPU in STOP mode.
- As long as the CPU remains in STOP mode, both NDR and LEN have the value 0.

To prevent information about the received data from being lost, you must perform a hot restart of the receiving CPU and call SFB 13 "BRCV" with EN_R = 1.

19.8 Writing Data to a Remote CPU with SFB/FB 15 "PUT"

Description

With SFB/FB 15 "PUT," you can write data to a remote CPU.

S7-300: The data is sent on a rising edge at REQ. The parameters ID, ADDR_1 and SD_1 are transferred on each rising edge at REQ. After a job has been completed, you can assign new values to the ID, ADDR_1 und SD_1 parameters.

S7-400: The SFB is started on a rising edge at control input REQ. In the process the pointers to the areas to be written (ADDR_i) and the data (SD_i) are sent to the partner CPU.

The remote partner saves the required data under the addresses supplied with the data and returns an execution acknowledgement.

Ensure that the areas defined with the parameters ADDR_i and SD_i match in terms of number, length, and data type.

If no errors occur, this is indicated at the next SFB/FB call with the status parameter DONE having the value 1.

The write job can only be activated again after the last job is completed.

The remote CPU can be in the RUN or STOP mode.

Errors and warnings are output via ERROR and STATUS if access problems occurred while the data were being written or if the execution check results in an error.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter request, activates the data exchange at a rising edge.
ID	INPUT	WORD	M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: 0: Job not started or still running 1: Job has been executed error-free
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information. ERROR=1 There is an error. STATUS supplies detailed information on the type of error.

Parameter	Declaration	Data Type	Memory Area	Description
S7-300: ADDR_1 S7-400: ADDR_i (1≤ i ≤4)	IN_OUT	ANY	M, D	Pointers to the areas on the partner CPU in which the data is to be written.
S7-300: SD_1 S7-400: SD_i (1≤ i ≤4)	IN_OUT	ANY	S7-300: M, D S7-400 I, Q, M, D, T, C	Pointers to the areas on the local CPU which contain the data to be sent. Only the following data types are permissible: BOOL (not allowed: Bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER. **Note:** If the ANY pointer accesses a DB, the DB must always be specified (for example: P# DB10.DBX5.0 Byte 10).

Error Information

The following table contains all the error information specific to SFB/FB 15 that can be output with the parameters ERROR and STATUS.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: new job is not effective since the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	2	Negative acknowledgement from the partner device. The function cannot be executed.
1	4	Errors in the send area pointers SD_i involving the data length or the data type.
1	8	Access error on the partner CPU.
1	10	Access to the local user memory not possible (for example, access to a deleted DB)
1	12	When the SFB was called, • An instance DB was specified that does not belong to SFB 15. • No instance DB was specified, but rather a shared DB. • No instance DB found (loading a new instance DB from the PG).
1	20	Insufficient memory. • H-System: SFB first called while update in progress • S7-300: - Maximum number of parallel jobs/instances exceeded - The instances were overloaded at CPU-RUN - Possible when first called

Data Consistency for S7-300

In order to ensure data consistency, send area SD_I may not be used again for writing until the current send process has been completed. This is the case when the state parameter DONE has the value 1.

Data Consistency for S7-400:

When a send operation is activated (rising edge at REQ) the data to be sent from the send area SD_i are copied from the user program. After the block call, you can write to these areas without corrupting the current send data.

Note

The send operation is only complete when the DONE status parameter has the value 1.

19.9 Read Data from a Remote CPU with SFB/FB 14 "GET"

Description

You can read data from a remote CPU with SFB/FB 14 "GET".

S7-300: The data is read on a rising edge at REQ. The parameters ID, ADDR_1 and RD_1 are transferred on each rising edge at REQ. After a job has been completed, you can assign new values to the ID, ADDR_1 und RD_1 parameters.

S7-400: The SFB is started with a rising edge at control input REQ. In the process the relevant pointers to the areas to be read out (ADDR_i) are sent to the partner CPU.

The remote partner returns the data.

The received data are copied to the configured receive areas (RD_i) at the next SFB/FB call.

Ensure that the areas defined with the parameters ADDR_i and RD_i match in terms of length and data type.

The completion of the job is indicated by a 1 at the status parameter NDR.

The read job can only be activated again after the previous job has been completed.

The remote CPU can be in the operating state RUN or STOP.

Errors and warnings are output via ERROR and STATUS if access problems occurred while the data were being read or if the data type check results in an error.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter request, activates the data exchange at a rising edge.
ID	INPUT	WORD	IM, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
NDR	OUTPUT	BOOL	I, Q, M, D, L	NDR status parameter: 0: Job not started or still active. 1: Job successfully completed.

Parameter	Declaration	Data Type	Memory Area	Description
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information. ERROR=1 An error occurred. STATUS gives you detailed information on the type of error.
S7-300: ADDR_1 S7-400: ADDR_i (1≤ i ≤4)	IN_OUT	ANY	S7-300: M, D S7-400: I, Q, M, D, T, C	Pointers to the areas on the partner CPU that are to be read.
S7-300: RD_1 S7-400: RD_i (1≤ i ≤4)	IN_OUT	ANY	S7-300: M, D S7-400: I, Q, M, D, T, C	Pointers to the areas on the local CPU in which the read data are entered. Only the following data types are permissible: BOOL (not allowed: Bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER. **Note:** If the ANY pointer accesses a DB, the DB must always be specified (for example: P# DB10.DBX5.0 Byte 10).

Error Information

The following table contains all the error information specific to SFB/FB 14 that can be output with the parameters ERROR and STATUS.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: new job is not effective since the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (for example: cable, CPU off, CP in STOP mode) • Connection to partner not yet established
1	2	Negative acknowledgement from the partner device. The function cannot be executed.
1	4	Errors in the receive area pointers RD_i involving the data length or the data type.
1	8	Access error on the partner CPU.
1	10	Access to the local user memory not possible (for example, access to a deleted DB)
1	12	When the SFB was called, • An instance DB was specified that does not belong to SFB 14. • No instance DB was specified, but rather a shared DB. • No instance DB found (loading a new instance DB from the PG).
1	20	Insufficient memory. • H-System: SFB first called while update in progress • S7-300: - Maximum number of parallel jobs/instances exceeded - The instances were overloaded at CPU-RUN - Possible when first called

Data Consistency

The data are received in a consistent state if the following point is observed:

Evaluate the part of the receive area RD_i currently being used completely before initiating another job.

19.10 Sending Data to a Printer with SFB 16 "PRINT"

Description

SFB 16 "PRINT" sends data and a formatting instruction to a remote printer, for example, via the CP 441.

When there is a rising edge at control input REQ, the format description (FORMAT) and the data (SD_i) are sent to the printer selected with ID and PRN_NR.

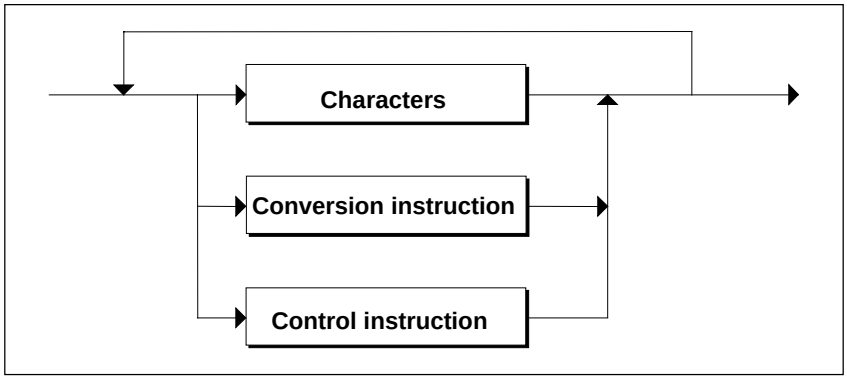
If you do not use all four send areas, you must make sure that the first area is described by the SD_1 parameter, the second area (if it exists) by the SD_2 parameter, the third area (if it exists) by SD_3.

Successful execution of the job is indicated by the DONE status parameter, errors are indicated by the ERROR and STATUS parameters.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter request, activates the data exchange at a rising edge.
ID	INPUT	WORD	M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: 0: Job not started or still running 1: Job has been executed error-free
ERROR STATE	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 and STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information ERROR=1 There is an error. STATUS supplies detailed information on the type of error.
PRN_NR	IN_OUT	BYTE	I, Q, M, D, L	Printer number
FORMAT	IN_OUT	STRING	I, Q, M, D, L	Format description
SD_i (1≤i≤4)	IN_OUT	ANY	M, D, T, C	Pointer to the "i-th" send data area. Only the following data types are permissible: BOOL (not allowed: bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER. **Note:** If the ANY pointer accesses a DB, the DB must always be specified (for example: p# DB10.DBX5.0 Byte 10).

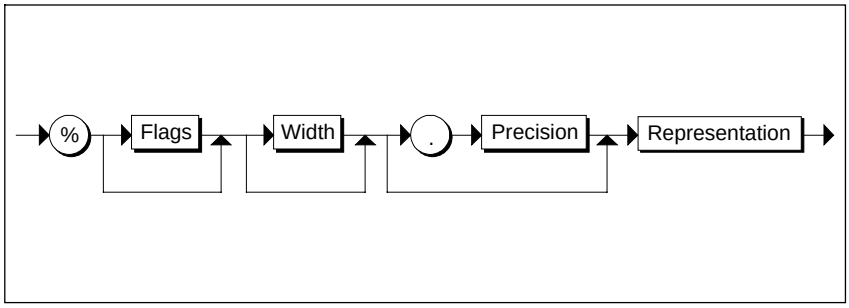
In_out Parameter FORMAT

The FORMAT character string contains printable characters and format elements. It has the following structure:



For each send area to be printed (SD_1 to SD_4) there must be one conversion instruction in FORMAT. The conversion instructions are applied to the send areas (SD_i) in the order in which they are formulated. Characters and instructions can follow each other in any order.

- Characters
  The following characters are permitted:
  - All printable characters
  - $$ (Dollar character), $' (single inverted comma),$L and $l (line feed), $P and $p (page), $R and $r (carriage return), $T and $t (tabulator)



Syntax Diagram of a Conversion Instruction

Element of a Conversion Instruction	Meaning	
Flags	• None:	right-justified output
	• -:	left-justified output
Width	• None:	output in standard representation
	• n:	exactly n characters are output. If the output is right-justified, this may be preceded by blanks, with left-justified output the blanks come after the characters.

Element of a Conversion Instruction	Meaning
Precision	The precision is only relevant for representations A, D, F and R (see following table). • None: output in standard representation • 0: no output of the decimal point or decimal places in the F and R representations • n: - with F and R: output of the decimal point and n decimal places - with A and D (date): number of digits for the year: possible values 2 and 4.
Representation	The following table contains: • The possible representations • The data types possible for each representation • The standard format for each representation(the printout is in the standard representation if no width and no precision are specified in the FORMAT parameter) and their maximum length

The following table shows possible modes of representation in the conversion instruction of the FORMAT parameter.

Representation	Possible Data Types	Example	Length	Comments
A, a	DATE	25.07.1996	10	-
	DWORD			
C, c	CHAR	K	1	-
	BYTE	M	1	
	WORD	KL	2	
	DWORD	KLMN	4	
	ARRAY of CHAR	KLMNOP	Number of characters	
	ARRAY of BYTE			
D, d	DATE	1996-07-25	10	-
	DWORD			
F, f	REAL	0.345678	8	-
	DWORD			
H, h	All data types incl. ARRAY of BYTE	Depending on data type	Depending on data type	Hexadecimal representation
I, i	INT	- 32 768	max. 6	-
	WORD	- 2 147 483 648	max. 11	

Representation	Possible Data Types	Example	Length	Comments
N, n	WORD	Text output	-	The corresponding send area SD_i contains a reference (number) to a text to be printed. The text is on the module (for example, CP 441) that creates a printable string. If no text is found under the specified number, ****** is output.
R, r	REAL	0.12E-04	8	-
	DWORD			
S, s	STRING	Text output		-
T, t	TIME	2d_3h_10m_5s_ 250ms	max. 21	If an error occurs, ****** is output.
	DWORD			
U, u	BYTE	255	max. 3	-
	WORD	65 535	max. 5	
	DWORD	4 294 967 295	max. 10	
X, x	BOOL	1	1	-
	BYTE	101 ..	8	
	WORD	101 ..	16	
	DWORD	101 ..	32	
Z, z	TIME_OF_DAY (TOD)	15:38:59.874	12	-

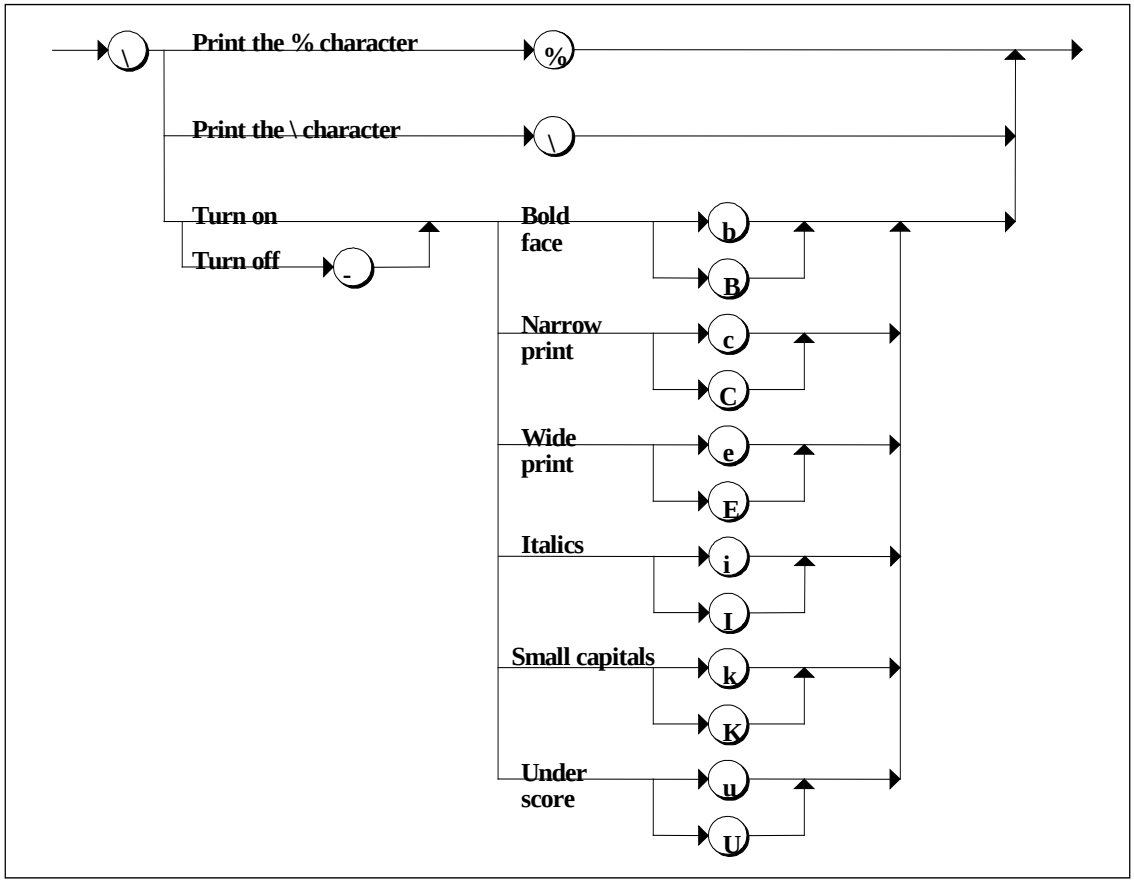
At the points in this table at which a maximum length is specified for the representation, the actual length can of course be shorter.

Note

With the data types C and S, the following points depend on the printer being used:

- which characters can be printed
- what the printer prints for non-printable characters, unless the printer driver has a conversion table for these characters.

- Control Instruction
  Using the control instruction you can do the following:
  - Print the characters % and \
  - Change the printer settings.



Syntax Diagram of the Control Instruction

If you attempt to disable, for example, a font that is not enabled or execute a function that the printer does not recognize, the control instruction is ignored. The following table contains the errors that may occur with the FORMAT in/out parameter.

Error	Printer Output
Conversion instruction cannot be executed	* characters are output according to the (maximum) length of the default representation or the specified width.
Specified width too small	In the representations A, C, D, N, S, T, and Z, as many characters are printed as specified by the selected width. With all other representations, * characters are printed across the specified width.

Error	Printer Output
Too many conversion instructions	The conversion instructions for which there is no send area pointer SD_i are ignored.
Too few conversion instructions	Send areas for which there is no conversion instruction are not printed out.
Undefined or unsupported conversion instructions	****** is printed out.
Incomplete conversion instruction	****** is printed out.
Undefined or unsupported control instructions	Control instructions that do not comply with the Syntax shown in the figure above are ignored.

Error Information

The following table contains all the error information specific to SFB 16 "PRINT" that can be printed out using the ERROR and STATUS parameters.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: New job not active because the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communication problems, for example • Connection description not loaded (local or remote) • Connection interrupted (for example, cable, CPU off, CP in STOP mode)
1	2	Negative acknowledgment from printer. The function cannot be executed.
1	3	PRN_NR is unknown on the communication link specified by the ID.
1	4	Error in the FORMAT in/out parameter or in the send area pointers SD_i in terms of the data length or data type.
1	6	The remote printer is OFFLINE.
1	7	The remote printer is not in the correct status (for example, paper out).
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	13	Error in the FORMAT in/out parameter
1	20	Insufficient memory H-System: SFB called while update in progress

Number of Transferable Data

The amount of data that can be transferred to a remote printer must not exceed a maximum length.

This maximum data length is calculated as follows:

maxleng = 420 - format

Format is the current length of the FORMAT parameter in bytes. The data to be printed can be distributed on one or more send areas.

19.11 Initiating a Warm or Cold Restart on a Remote Device with SFB 19 "START"

Description

If there is a rising edge at control input REQ, SFB 19 "START," this activates a warm or cold restart on the remote device addressed by the ID. If the remote system is a fault-tolerant system, the effect of the restart request depends on the parameter PI_NAME: the start request is valid either for exactly one CPU or for all CPUs of this system. The following conditions must be met if the remote device is a CPU:

- The CPU must be in the STOP mode.
- The key switch of the CPU must be set to "RUN" or "RUN-P."

Once the warm or cold restart is completed, the device changes to the RUN mode and sends a positive execution acknowledgement. When the positive acknowledgement is evaluated, the status parameter DONE is set to 1. If any errors occur, they are indicated by the status parameters ERROR and STATUS.

A further warm or cold restart can only be activated in the same remote device after the last complete restart is completed.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter request, activates the SFB at a rising edge
ID	INPUT	WORD	I, Q, M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: 0: Job not started or still running 1: Job has been executed error-free
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 and STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information ERROR=1 There is an error. STATUS supplies detailed information on the type of error.

Parameter	Declaration	Data Type	Memory Area	Description
PI_NAME	IN_OUT	ANY	I, Q, M, D, T, C	Pointer to the memory area in which the name of the program (ASCII code) to be started is located. This name must not contain more than 32 characters. With an S7 PLC, it must be P_PROGRAM. With an H system, the following names are possible: • P_PROGRAM (the start job is valid for all CPUs in the H system.) • P_PROG_0 (the start job is valid for the CPU in Rack 0 in the H system.) • P_PROG_1 (the start job is valid for the CPU in Rack 1 in the H system.)
ARG	IN_OUT	ANY	I, Q, M, D, T, C	Execution argument. • If you do not assign a value to ARG, a warm restart is run on the remote device. • If you assign the value "C," a cold restart is run on the remote device (if the remote device is capable of this type of startup).
IO_STATE	IN_OUT	BYTE	I, Q, M, D, L	Not currently relevant. Do not assign a value to this parameter if your communication partner is an S7 programmable controller.

Error Information

The following table contains all the error information specific to SFB 19 that can be output with the parameters ERROR and STATUS.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: new job is not effective since the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (for example, cable, CPU off, CP in STOP mode)
1	2	Negative acknowledgement from the partner device. The function cannot be executed.
1	3	The program name entered for PI_NAME is unknown.
1	4	Error in the pointers PI_NAME or ARG involving the data length or the data type.
1	7	No complete restart possible on the partner device.
1	10	Access to the local user memory not possible (for example, access to a deleted DB)

ERROR	STATUS (Decimal)	Explanation
1	12	When the SFB was called, • An instance DB was specified that does not belong to SFB 19 • No instance DB was specified, but rather a shared DB. • No instance DB found (loading a new instance DB from the PG).
1	20	Insufficient memory. H-System: SFB called while update in progress

19.12 Changing a Remote Device to the STOP State with SFB 20 "STOP"

Description

If there is a rising edge at control input REQ, SFB 20 "STOP," this activates a change to the STOP mode on the remote device addressed by the ID. The mode change is possible when the device is in the RUN, HOLD or STARTUP modes.

If the remote system is a fault-tolerant system, the effect of the restart request depends on the parameter PI_NAME: the start request is valid either for exactly one CPU or for all CPUs of this system.

Successful execution of the job is indicated by the status parameter DONE having the value 1. If any errors occur, they are indicated in the status parameters ERROR and STATUS.

The mode change can only be started again in the same remote device when the previous SFB 20 call has been completed.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter request, activates the SFB at a rising edge
ID	INPUT	WORD	I, Q, M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: 0: Job not started or still running 1: Job has been executed error-free
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 and STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information. ERROR=1 There is an error. STATUS supplies detailed information on the type of error.

Parameter	Declaration	Data Type	Memory Area	Description
PI_NAME	IN_OUT	ANY	I, Q, M, D	Pointer to the memory area in which the name of the program (ASCII code) to be started is located. This name must not contain more than 32 characters. With an S7 PLC, it must be P_PROGRAM. With an H system, the following names are possible: • P_PROGRAM (the start job is valid for all CPUs in the H system.) • P_PROG_0 (the start job is valid for the CPU in Rack 0 in the H system.) • P_PROG_1 (the start job is valid for the CPU in Rack 1 in the H system.)
IO_STATE	IN_OUT	BYTE	I, Q, M, D, L	Not currently relevant. Do not assign a value to this parameter if your communication partner is an S7 programmable controller.

Error Information

The following table contains all the error information specific to SFB 20 that can be output with the parameters ERROR and STATUS.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: new job is not effective since the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems, for example • Connection description not loaded (local or remote) • Connection interrupted (for example, cable, CPU off, CP in STOP mode)
1	2	Negative acknowledgement from the partner device. The function cannot be executed.
1	3	The program name entered for PI_NAME is unknown.
1	4	Error in the pointer PI_NAME involving the data length or the data type.
1	7	The partner device is already in the STOP state.
1	10	Access to the local user memory not possible (for example, access to a deleted DB)
1	12	When the SFB was called, • An instance DB was specified that does not belong to SFB 20 • No instance DB was specified, but rather a shared DB. • No instance DB found (loading a new instance DB from the PG).
1	20	Insufficient memory. H-System: SFB called while update in progress

19.13 Initiating a Hot Restart on a Remote Device with SFB 21 "RESUME"

Description

If there is a rising edge at control input REQ, SFB 21 "RESUME" activates a hot restart on the remote device selected with the ID.
The following conditions must be met if the remote device is a CPU:

- The CPU must be in the STOP mode.
- The key switch of the CPU must be set to "RUN" or "RUN-P."
- When you created the configuration with STEP 7, you allowed for a manual hot restart.
- There must be no condition preventing a hot restart.

Once the hot restart has been completed, the device changes to the RUN mode and sends a positive execution acknowledgement. When the positive acknowledgement is evaluated, the status parameter DONE is set to 1. Any errors that occurred are indicated in the status parameters ERROR and STATUS.

A restart can only be activated again in the same remote device after the previous hot restart has been completed.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter request, activates the SFB at a rising edge
ID	INPUT	WORD	I, Q, M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: 0: Job not started or still running 1: Job has been executed error-free
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 and STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information ERROR=1 There is an error. STATUS supplies detailed information on the type of error.

Parameter	Declaration	Data Type	Memory Area	Description
PI_NAME	IN_OUT	ANY	I, Q, M, D	Pointer to memory area in which the name of the program (ASCII code) to be started is located. This name must not contain more than 32 characters. With an S7 PLC, it must be P_PROGRAM.
ARG	IN_OUT	ANY	I, Q, M, D, T, C	Execution argument. Not currently relevant. Do not assign a value to this parameter if your communication partner is an S7 programmable controller.
IO_STATE	IN_OUT	BYTE	I, Q, M, D, L	Not currently relevant. Do not assign a value to this parameter if your communication partner is an S7 programmable controller.

Error Information

The following table contains all the error information specific to SFB 21 that can be output with the parameters ERROR and STATUS.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: new job is not effective since the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	• Communications problems, for example, connection description not loaded (local or remote) • Connection interrupted (for example, cable, CPU off, CP in STOP mode)
1	2	Negative acknowledgement from the partner device. The function cannot be executed.
1	3	The program name entered in PI_NAME is unknown.
1	4	Error in the pointers PI_NAME or ARG involving the data length or the data type.
1	7	Hot restart not possible
1	10	Access to the local user memory not possible (for example, access to a deleted DB)
1	12	When the SFB was called, • An instance DB was specified that does not belong to SFB 21 • No instance DB was specified, but rather a shared DB. • No instance DB found (loading a new instance DB from the PG).
1	20	Insufficient memory. H-System: SFB called while update in progress

19.14 Querying the Status of a Remote Partner with SFB 22 "STATUS"

Description

Using SFB 22 "STATUS," you can query the status of a remote communications partner.

If there is a rising edge at control input REQ, a job is sent to the remote partner. The reply is evaluated to determine whether problems have occurred. If no errors occurred, the received status is copied to the variables PHYS, LOG and LOCAL with the next SFB call. The completion of this job is indicated by the status parameter NDR having the value 1.

You can only query the status of the same communications partner again after the last query is completed.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter request, activates the SFB at a rising edge
ID	INPUT	WORD	I, Q, M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
NDR	OUTPUT	BOOL	I, Q, M, D, L	NDR status parameter: 0: Job not started or still running 1: Job has been executed error-free
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 and STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information. ERROR=1 There is an error. STATUS supplies detailed information on the type of error.
PHYS	IN_OUT	ANY	I, Q, M, D	Physical status (minimum length: one byte). Possible values: • 10H functioning • 13H service required
LOG	IN_OUT	ANY	I, Q, M, D	Logical status (minimum length: one byte). Possible value: • 00H status change permitted
LOCAL	IN_OUT	ANY	I, Q, M, D	Status if the partner device is an S7 CPU (minimum length: two bytes)

In/Out Parameter LOCAL

If the communications partner is an S7 CPU, the in/out parameter LOCAL contains its current status. The first byte is reserved, the second byte contains an ID for the status.

Operating Mode	Corresponding Identifier
STOP	00H
Warm restart	01H
RUN	02H
Hot restart	03H
HOLD	04H
Cold restart	06H
RUN_R	09H
LINK-UP	0BH
UPDATE	0CH

Error Information

The following table contains all the error information specific to SFB 22 that can be output with the parameters ERROR and STATUS.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: new job is not effective since the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems, for example • Connection description not loaded (local or remote) • Connection interrupted (for example, cable, CPU off, CP in STOP mode)
1	2	Negative acknowledgement from the partner device. The function cannot be executed.
1	4	Error in PHYS, LOG or LOCAL involving the data length or data type.
1	8	Access to a remote object was rejected.
1	10	Access to a local user memory not possible (for example, access to a deleted DB).
1	12	When the SFB was called, • An instance DB was specified that does not belong to SFB 22 • No instance DB was specified, but rather a shared DB. • No instance DB found (loading a new instance DB from the PG).
1	20	Insufficient memory. H-System: SFB called while update in progress

19.15 Receiving the Status of a Remote Device with SFB 23 "USTATUS"

Description

SFB 23 "USTATUS" receives the device status of a remote communication partner. The partner sends its status unsolicited when a change occurs if this is configured in STEP 7.

If the value 1 is applied to the control input EN_R when the CFB is called and there is a frame from the partner, the status information is entered in the variables PHYS, LOG and LOCAL the next time the SFB is called. Completion of this job is indicated by the status parameter NDR having the value 1.

The transfer of the operating status messages must be enabled on the connection used by USTATUS.

Note

You can only use one instance of SFB 23 per connection.

Parameter	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L	The enabled to receive control parameter signalizes that the unit is ready to receive when the input is set.
ID	INPUT	WORD	I, Q, M, D, constant	Addressing parameter ID, refer to Common Parameters of the SFBs/FBs and SFC/FC for the S7 communication
NDR	OUTPUT	BOOL	I, Q, M, D, L	NDR status parameter: 0: Job not started or still running 1: Job has been executed error-free
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 and STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information. ERROR=1 There is an error. STATUS supplies detailed information on the type of error.

Parameter	Declaration	Data Type	Memory Area	Description
PHYS	IN_OUT	ANY	I, Q, M, D	Physical status (minimum length: one byte). Possible values: • 10H functional • 13H service required
LOG	IN_OUT	ANY	I, Q, M, D	Logical status (minimum length: one byte) Possible value: • 00H status change permitted
LOCAL	IN_OUT	ANY	I, Q, M, D	Status if the partner device is an S7 CPU (minimum length: one byte)

In/Out Parameter LOCAL

If the communications partner is an S7 CPU, the in/out parameter LOCAL contains its current status. The first byte is reserved, the second byte contains an ID for the status.

Operating Mode	Corresponding Identifier
STOP	00H
Warm restart	01H
RUN	02H
Hot restart	03H
HOLD	04H
Cold restart	06H
RUN (H system status: redundant)	09H
LINK-UP	0BH
UPDATE	0CH

Error Information

The following table contains all the error information specific to SFB 23 that can be output with the parameters ERROR and STATUS.

ERROR	STATUS (Decimal)	Explanation
0	9	Overrun warning: an older device status has been overwritten by a more recent device status.
0	25	Communication has started. The job is being processed.
1	1	Communications problems, for example • Connection description not loaded (local or remote) • Connection interrupted (for example, cable, CPU off, CP in STOP mode)
1	4	Error in PHYS, LOG or LOCAL involving the data length or data type.
1	10	Access to a local user memory not possible (for example, access to a deleted DB).

ERROR	STATUS (Decimal)	Explanation
1	12	When the SFB was called, • An instance DB was specified that does not belong to SFB 23 • No instance DB was specified, but rather a shared DB. • No instance DB found (loading a new instance DB from the PG).
1	18	There is already an instance for SFB 23 "USTATUS" for the connection identified by ID.
1	19	The remote CPU sends a data faster than it can be accepted in the user program by the SFB.
1	20	Insufficient memory. SFB called while update in progress

19.16 Querying the Status of the Connection Belonging to an SFB Instance with SFC 62 "CONTROL"

Description

With SFC 62 "CONTROL," you can query for S7-400 the status of a connection belonging to a local communication SFB instance.

After calling the system function with the value 1 at control input EN_R, the current status of the connection belonging to the communication SFB instance selected with I_DB is queried.

Parameter	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L, constant	The enabled to receive control parameter signalizes that the unit is ready to receive when the input is set.
I_DB	INPUT	BLOCK_DB	I, Q, M, D, L, constant	Number of the instance DB
OFFSET	INPUT	WORD	I, Q, M, D, L, constant	Number of the data record in the multiple instance DB (if no multiple instance DB exists, 0 must be entered here).
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	ERROR and STATUS state parameters, error display: ERROR=0 and STATUS has the value: 0000H: neither warning nor error <> 0000H: Warning, STATUS supplies detailed information. ERROR=1 There is an error. STATUS supplies detailed information on the type of error.
I_TYP	OUTPUT	BYTE	I, Q, M, D, L	Identifier for the block type belonging to the selected instance

Parameter	Declaration	Data Type	Memory Area	Description
I_STATE	OUTPUT	BYTE	I, Q, M, D, L	• = 0: The corresponding SFB instance has not been called since the last cold/warm restart or loading. • <> 0: The corresponding SFB instance has been called at least once since the last cold/warm restart or loading.
I_CONN	OUTPUT	BOOL	I, Q, M, D, L	Status of the corresponding connection, possible values: • 0: Connection down or not established • 1: Connection exists
I_STATUS	OUTPUT	WORD	I, Q, M, D, L	Status parameter STATUS of the queried communication SFB instance

Output Parameter I_TYP

The following table lists the different SFB types and the corresponding identifiers

SFB Type	Identifier (W#16#...)
USEND	00
URCV	01
BSEND	04
BRCV	05
GET	06
PUT	07
PRINT	08
START	0B
STOP	0C
RESUME	0D
STATUS	0E
USTATUS	0F
ALARM	15
ALARM_8	16
ALARM_8P	17
NOTIFY	18
AR_SEND	19
(no SFB exists; I_DB or OFFSET wrong)	FF

Error Information

The output parameter RET_VAL can have the following two values with SFC 62 "CONTROL":

- 0000H: no error occurred during execution of the SFC.
- 8000H: an error occurred during execution of the SFC.

Note

Even if the value 0000H is indicated in the output parameter RET_VAL, the output parameters ERROR and STATUS should be evaluated.

ERROR	STATUS (Decimal)	Explanation
1	10	Access to local user memory is not possible (for example, a memory byte was specified as the actual parameter for I_TYP and this memory byte does not exist in the CPU being used).
1	12	For the number specified with I_DB, • There is no instance DB, but rather a shared DB, • There is no DB, or the instance has been destroyed.

19.17 Querying the Connection Status with FC 62 "C_CNTRL"

Description

Query a connection status for S7-300 with FC 62 "C_CNTRL".

The current status of the communication that has been determined via ID is queried after the system function has been called with value 1 at the control input EN_R.

Parameters	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L, const.	Control parameter enabled to receive, signals ready to receive if the input is set.
ID	INPUT	WORD	M, D, const.	Addressing parameter ID, see Common parameters of SFBs/FBs and SFC/FC of the S7-communication
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
ERROR STATUS	OUTPUT OUTPUT	BOOL WORD	I, Q, M, D, L I, Q, M, D, L	Status parameter ERROR and STATUS, Error display: ERROR=0 and STATUS have the values: 0000H: Neither warning nor error <> 0000H: Warning, STATUS supplies detailed information. ERROR=1 There is an error. STATUS supplies detailed information on the type of error.
C_CONN	OUTPUT	BOOL	I, Q, M, D, L	Status of the corresponding connection. Possible values: • 0: The connection was dropped or it

Parameters	Declaration	Data Type	Memory Area	Description
				is not up. • 1: The connection is up.
C_STATUS	OUTPUT	WORD	I, Q, M, D, L	Connection status: • W#16#0000: Connection is not established • W#16#0001: Connection is being established • W#16#0002: Connection is established • W#16#000F: No data on connection status available (such as at CP startup) • W#16#00FF: Connection is not configured

Error Information

The output parameter RET_VAL can assume the following values at FC 62 "C_CNTRL":

- 0000H: No error when FC was executed.
- 8000H: Error when FC was executed.

Note

The output parameters ERROR and STATUS are to be evaluated regardless of the output parameter RET_VAL showing the value 0000H.

ERROR	STATUS (Decimal)	Description
1	10	CP access error. Another job is currently running. Repeat job later.

20 Communication SFCs for Non-Configured S7 Connections

20.1 Common Parameters of the Communication SFCs

Input Parameter REQ

The input parameter REQ (request to activate) is a level-triggered control parameter. It is used to trigger the job (the data transfer or the connection abort):

- If you call the SFC for a job that is not currently active, you trigger the job with REQ=1. If there is no connection to the communication partner when the communication SFC is called the first time, the connection is established before data transfer begins.
- If you trigger a job and it is not yet completed when you call the SFC again for the same job, REQ is not evaluated by the SFC.

Input Parameter REQ_ID (only SFC 65 and SFC 66)

The input parameter REQ_ID is used to identify your send data. It is passed by the operating system of the sending CPU to the SFC 66 "X_RCV" of the CPU of the communication partner.

You require the REQ_ID parameter on the receiving end

- If you call several SFCs 65 "X_SEND" with different parameters REQ_ID on one send CPU and transfer the data to a communication partner.
- If you use SFC 65 "X_SEND" to send data to one communication partner from several send CPUs.

By evaluating REQ_ID you can save the received data in different memory areas.

Output Parameters RET_VAL and BUSY

The communication SFCs are executed asynchronously, this means that the execution of a job extends over more than one SFC call. The output parameters RET_VAL and BUSY indicate the status of the job. See also Meaning of the Parameters REQ, RET_VAL and BUSY with Asynchronous SFCs

Input Parameter CONT

The input parameter CONT (continue) is a control parameter. Using this parameter, you decide whether or not a connection to the communication partner remains established after the job is completed.

- If you select CONT=0 at the first call, the connection is terminated again after the data transfer is completed. The connection is then available again for data exchange with a new communication partner.

  This method ensures that connection resources are only occupied when they are actually in use.

- If you select CONT=1 at the first call, the connection remains established on completion of the data transfer.

  This method is, for example, useful when you exchange data cyclically between two stations.

Note

A connection established with CONT=1 can be terminated explicitly with SFC 69 "X_ABORT" or with SFC 74 "I_ABORT."

20.2 Error Information of the Communication SFCs for Non-Configured S7 Connections

Error Information

The "real" error information for SFCs 65 to 74 as shown in the table "Specific Error Information for SFCs 65 to 74" can be classified as follows:

Error Code(W#16# ...)	Explanation
809x	Error on the CPU on which the SFC is executed
80Ax	Permanent communication error
80Bx	Error on the communication partner
80Cx	Temporary error

Specific Error Information for SFCs 65 to 74

Error Code (W#16# ...)	Explanation (General)	Explanation (for Specific SFC)
0000	Execution completed without errors.	SFC 69 "X_ABORT" and SFC 74 "I_ABORT": REQ=1, and the specified connection is not established.
		SFC 66 "X_RCV":EN_DT=1 and RD=NIL
00xy	-	SFC 66 "X_RCV" with NDA=1 and RD<>NIL: RET_VAL contains the length of the received data (with EN_DT=0) or the length of the data copied to RD (with EN_DT=1).
		SFC 67 "X_GET": RET_VAL contains the length of the

Error Code (W#16# ...)	Explanation (General)	Explanation (for Specific SFC)
		received block of data.
		SFC 72 "I_GET": RET_VAL contains the length of the received block of data.
7000	-	SFC 65 "X_SEND," SFC 67 "X_GET," SFC 68 "X_PUT," SFC 69 "X_ABORT," SFC 72 "I_GET," SFC 73 "I_PUT" and SFC 74 "I_ABORT": call with REQ = 0 (call without execution), BUSY has the value 0, no data transfer active.
		SFC 66 "X_RCV": EN_DT=0/1 and NDA=0
7001	First call with REQ=1: data transfer was triggered; BUSY has the value 1.	-
7002	Interim call (REQ irrelevant): data transfer is already active ; BUSY has the value 1.	SFC 69 "X_ABORT" and SFC 74 "I_ABORT": Interim call using REQ=1
8090	Specified destination address of the communication partner is invalid, for example: • Wrong IOID • Wrong base address exists • Wrong MPI address (> 126)	-
8092	Error in SD or RD, for example: addressing the local data area is not permitted.	SFC 65 "X_SEND," for example • illegal length for SD • SD=NIL is illegal
		SFC 66 "X_RCV," for example • More data were received than can fit in the area specified by RD. • RD is of the BOOL data type, the received data is, however, longer than a byte.
		SFC 67 "X_GET" and SFC 72 "I_GET," for example • • illegal length for RD • • the length or the data type of RD does not match the received data. • • RD=NIL is not permitted.
		SFC 68 "X_PUT" and SFC 73 "I_PUT," for example • • illegal length for SD • • SD=NIL is illegal
8095	The block is already being executed in a lower priority class.	-

Error Code (W#16# ...)	Explanation (General)	Explanation (for Specific SFC)
80A0	Error in the received acknowledgment	SFC 68 "X_PUT" and SFC 73 "I_PUT": The data type specified in the SD of the sending CPU is not supported by the communication partner.
80A1	Communication problems: SFC call after terminating an existing connection	-
80B0	Object is not obtainable, for example, DB not loaded	Possible with SFC 67 "X_GET" and SFC 68 "X_PUT" and SFC 72 "I_GET" and SFC 73 "I_PUT"
80B1	Error in the ANY pointer. The length of the data area to be sent is incorrect.	-
80B2	Hardware error: module does not exist • The configured slot is not occupied. • Actual module type does not match expected type • Distributed peripheral I/Os not available. • No entry for the module in the corresponding SDB.	Possible with SFC 67 "X_GET" and SFC 68 "X_PUT" and SFC 72 "I_GET" and SFC 73 "I_PUT"
80B3	Data may either only be read or only written, for example, write-protected DB	Possible with SFC 67 "X_GET" and SFC 68 "X_PUT" and SFC 72 "I_GET" and SFC 73 "I_PUT"
80B4	Data type error in the ANY pointer, or ARRAY of the specified type not allowed.	SFC 67 "X_GET" and SFC 68 "X_PUT" and SFC 72 "I_GET" and SFC 73 "I_PUT": The data type specified in VAR_ADDR is not supported by the communication partner.
80B5	Execution rejected due to illegal mode	Possible with SFC 65 "X_SEND"
80B6	The received acknowledgment contains an unknown error code.	-
80B7	Data type and/or length of the transferred data does not fit in the area on the partner CPU to which it should be written.	Possible with SFC 68 "X_PUT" and SFC 73 "I_PUT"
80B8	-	SFC 65 "X_SEND":

Error Code (W#16# ...)	Explanation (General)	Explanation (for Specific SFC)
		The SFC 66 "X_RCV" of the communication partner did not allow data acceptance (RD=NIL).
80B9	-	SFC 65 "X_SEND": The block of data was identified by the communication partner (SFC 66 "X_RCV" call with EN_DT=0), it has not yet been entered in the user program because the partner is in the STOP mode.
80BA	The response of the communication partner does not fit in the communication frame.	-
80C0	The specified connection is being used by another job.	-
80C1	Lack of resources on the CPU on which the SFC is executed, for example: The maximum number of different send jobs is already being executed on the module. The connection resource is in use, for example, to receive data.	-
80C2	Temporary lack of resources on the communication partner, for example: • The communication partner is currently processing the maximum number of jobs. • The required resources, memory, etc. are being used. • Not enough work memory. (compress memory).	-
80C3	Error in connection establishment, for example: • The local S7 station is not attached to the MPI subnet. • You have addressed your own station on the MPI subnet. • The communication partner is no longer	-

Error Code (W#16# ...)	Explanation (General)	Explanation (for Specific SFC)
	obtainable. • Temporary lack of resources on the communication partner	

20.3 Sending Data to a Communication Partner outside the Local S7 Station with SFC 65 "X_SEND"

Description

With SFC 65 "X_SEND," you send data to a communication partner outside the local S7 station.

The data are received on the communication partner using SFC 66 "X_RCV."

The data is sent after calling the SFC with REQ=1.

Make sure that the send area defined by the parameter SD (on the sending CPU) is smaller than or the same size as the receive area defined by the parameter RD (on the communication partner). If SD is of the BOOL data type, RD must also be BOOL.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "request to activate", refer to Common Parameters of the SFCs for S7 Basic Communication
CONT	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "continue", refer to Common Parameters of the SFCs for S7 Basic Communication
DEST_ID	INPUT	WORD	I, Q, M, D, L, constant	Address parameter "destination ID." This contains the MPI address of the communication partner. You configured this with STEP 7.
REQ_ID	INPUT	DWORD	I, Q, M, D, L, constant	Job identifier. This is used to identify the data on the communication partner.
SD	INPUT	ANY	I, Q, M, D	Reference to the send area. The following data types are allowed: BOOL, BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5_TIME, DATE_AND_TIME and arrays of these data types except for BOOL.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains the corresponding error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: Sending is not yet completed.

Parameter	Declaration	Data Type	Memory Area	Description
				BUSY=0: Sending is completed or no send function active.

Data Consistency

The data are sent in a consistent state.

Error Information

See Error Information of the Communication SFCs for Non-Configured S7 Connections

20.4 Receiving Data from a Communication Partner outside the Local S7 Station with SFC 66 "X_RCV"

Description

With SFC 66 "X_RCV," you receive the data sent by one or more communication partners outside the local S7 station using SFC 65 "X_SEND."

With SFC 66 "X_RCV,"

- You can check whether data have been sent and are waiting to be copied. The data were entered in an internal queue by the operating system.
- You can copy the oldest block of data from the queue to a selected receive area.

Parameter	Declaration	Data Type	Memory Area	Description
EN_DT	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "enable data transfer." With the value 0, you can check whether at least one block of data is waiting to be entered in the receive area. The value 1 copies the oldest block of data in the queue to the area of the work memory specified in RD.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains the corresponding error code. If no error occurs, RET_VAL contains the following: • W#16#7000 if EN_DT=0/1 and NDA=0. In this case, there is no data block in the queue. • If EN_DT=0 and NDA=1 the length of the oldest block of data entered in the queue as a positive number in bytes. • If EN_DT=1 and NDA=1 the length of

Parameter	Declaration	Data Type	Memory Area	Description
				the block of data copied to the RD receive area as a positive number in bytes.
REQ_ID	OUTPUT	DWORD	I, Q, M, D, L	Job identifier of the SFC "X_SEND" whose data are first in the queue, in other words the oldest data in the queue. If there is no block of data in the queue, REQ_ID has the value 0.
NDA	OUTPUT	BOOL	I, Q, M, D, L	Status parameter "new data arrived." NDA=0: • There is no block of data in the queue. NDA=1: • The queue contains at least one block of data. (SFC 66 call with EN_DT=0). • The oldest block of data in the queue was copied to the user program. (SFC 66 call with EN_DT=1).
RD	OUTPUT	ANY	I, Q, M, D	Reference to the received data area. The following data types are allowed: BOOL, BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5_TIME, DATE_AND_TIME and arrays of these data types except for BOOL. If you want to discard the oldest block of data in the queue, assign the value NIL to RD.

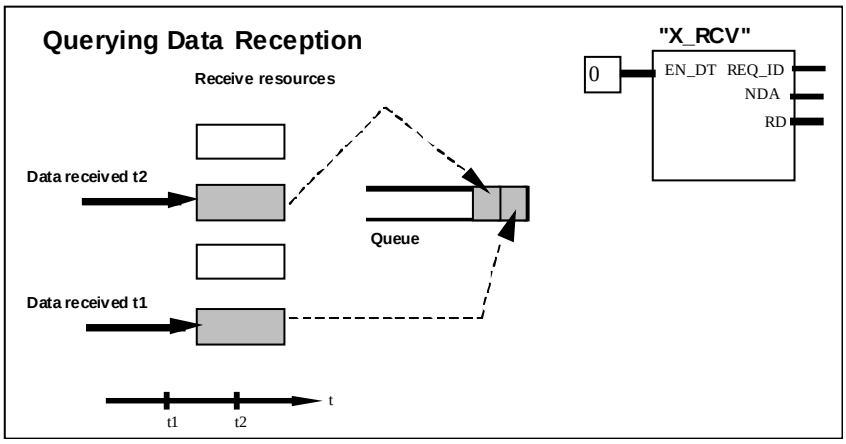
Indicating Reception of Data with EN_DT=0

As soon as data from a communication partner arrive, they are entered in the queue by the operating system in the order in which they are received.

If you want to check whether at least one block of data is in the queue, call SFC 66 with EN_DT=0 and evaluate the output parameter NDA as follows:

- NDA=0 means that the queue does not contain a block of data. REQ_ID is irrelevant, RET_VAL has the value W#16#7000.
- NDA=1 means that there is at least one block of data in the queue that can be fetched.

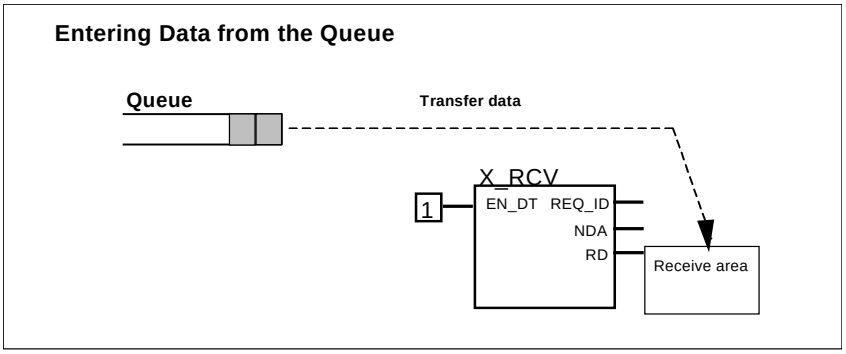
  In this case, you should also evaluate the output parameter RET_VAL and, if applicable, REQ_ID. RET_VAL contains the length of the block of data in bytes, REQ_ID contains the job identifier of the sending block. If there are several blocks of data in the queue, REQ_ID and RET_VAL belong to the oldest block of data in the queue.



Data receipt

Entering Data in the Receive Area with EN_DT=1

When you call SFC 66 "X_RCV" with EN_DT=1, the oldest block of data in the queue is copied to the area of the work memory specified by RD. RD must be larger or the same size as the send area of the corresponding SFC 65 "X_SEND" defined by the SD parameter. If the input parameter SD is of the BOOL data type, RD must also be the BOOL data type. If you want to enter the received data in different areas, you can query REQ_ID (SFC call with EN_DT = 0) and select a suitable RD in the follow-on call (with EN_DT = 1).If no error occurs when the data are copied, RET_VAL contains the length of the copied block of data in bytes and a positive acknowledgment is sent to the sender.



Data acceptance

Discarding Data

If you do not want to enter the data from the queue, assign the value NIL to RD (see ***I232I***). In this case, the sender receives a negative acknowledgment (RET_VAL of the corresponding SFC 65 "X_SEND" has the value W#1680B8). RET_VAL of the SFC 66 "X_RCV" has the value 0.

Changing to the STOP Mode

If the CPU changes to the STOP mode

- all newly arriving jobs are acknowledged negatively.
- all jobs that have arrived and are in the queue are acknowledged negatively.
  - If the STOP is followed by a warm or cold restart, the blocks of data are all discarded.
  - If the STOP is followed by a restart, (not possible on an S7-300 and an S7-400H) the block of data belonging to the oldest job is entered in the user program, if the queue was queried before the change to the STOP mode (by calling SFC 66 "X_RCV" with EN_DT=0). Otherwise it is discarded.

    All other blocks of data are discarded.

Connection Abort

If the connection is terminated a job belonging to the connection that is already in the queue is discarded.

Exception: If this job is the oldest in the queue, and you have already detected its presence by calling SFC 66 "X_RCV" with EN_DT=0, you can enter it in the receive area with EN_DT=1.

Error Information

See Error Information of the Communication SFCs for Non-Configured S7 Connections.

20.5 Writing Data to a Communication Partner outside the Local S7 Station with SFC 68 "X_PUT"

Description

With SFC 68 "X_PUT," you write data to a communication partner that is not in the same local S7 station. There is no corresponding SFC on the communication partner.

The write job is activated after calling the SFC with REQ=1. Following this, you continue to call the SFC until the acknowledgment is received with BUSY=0.

Make sure that the send area defined with the SD parameter (on the sending CPU) is the same length as the receive area defined by the VAR_ADDR parameter (on the communication partner). The data types of SD and VAR_ADDR must also match.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "request to activate," s. Common Parameters for the S7 basic comminication of the SFCs
CONT	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "continue," see Control parameters "request to activate", s. Common Parameters for the S7 basic comminication of the SFCs
DEST_ID	INPUT	WORD	I, Q, M, D, L, constant	Address parameter "destination ID." This contains the MPI address of the communication partner. You configured this with STEP 7.
VAR_ADDR	INPUT	ANY	I, Q, M, D	Reference to the area on the partner CPU to which the data will be written. You must choose a data type that is supported by the communication partner.
SD	INPUT	ANY	I, Q, M, D	Reference to the area in the local CPU that contains the data to be sent. The following data types are allowed: BOOL, BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5_TIME, DATE_AND_TIME and arrays of these data types except for BOOL. SD must have the same length as the VAR_ADDR parameter of the communication partner. The data types of SD and VAR_ADDR must also match.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains the corresponding error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: Sending is not yet completed. BUSY=0: Sending is completed or no send function active.

Changing to the STOP Mode

If the CPU changes to the STOP mode, the connection established by SFC 68 "X_PUT" is terminated. Data can no longer be sent. If the send data have already been copied to the internal buffer when the CPU changes mode, the contents of the buffer are discarded.

Communication Partner Changes to the STOP Mode

If the CPU of the communication partner changes to the STOP mode, this does not affect the data transfer with SFC 68 "X_PUT." The data can also be written with the partner in the STOP mode.

Data Consistency

The data are sent in a consistent state.

Error Information

See Error Information of the Communication SFCs for Non-Configured S7 Connections.

20.6 Reading Data from a Communication Partner outside the Local S7 Station with SFC 67 "X_GET"

Description

With SFC 67 "X_GET," you can read data from a communication partner that is not in the local S7 station. There is no corresponding SFC on the communication partner.

The read job is activated after calling the SFC with REQ=1. Following this, you continue to call the SFC until the data reception is indicated by BUSY=0. RET_VAL then contains the length of the received block of data in bytes.

Make sure that the receive area defined with the RD parameter (on the receiving CPU) is at least al long as the area to be read as defined by the VAR_ADDR parameter (on the communication partner). The data types of RD and VAR_ADDR must also match.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "request to activate," see Common Parameters of the Communication SFCs
CONT	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "continue," see Common Parameters of the Communication SFCs
DEST_ID	INPUT	WORD	I, Q, M, D, L, constant	Address parameter "destination ID." This contains the MPI address of the communication partner. You configured this with STEP 7.
VAR_ADDR	INPUT	ANY	I, Q, M, D	Reference to the area on the partner CPU from which the data will be read. You must choose a data type that is supported by the communication partner.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains the corresponding error code. If no error occurs, RET_VAL contains the length of the block of data copied to the receive area RD as a positive number of bytes.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: Receiving is not yet completed.

Parameter	Declaration	Data Type	Memory Area	Description
				BUSY=0: Receiving is completed or there is no receive job active.
RD	OUTPUT	ANY	I, Q, M, D	Reference to the receive area (receive data area). The following data types are allowed: BOOL, BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5_TIME, DATE_AND_TIME and arrays of these data types except for BOOL. The receive area RD must be at least as long as the area from which the data are read defined by the VAR_ADDR parameter on the communication partner. The data types of RD and VAR_ADDR must also match.

Changing to the STOP Mode

If the CPU changes to the STOP mode, the connection established by SFC 67 "X_GET" is terminated. Whether or not the received data located in a buffer of the operating system are lost depends on the type of restart performed:

- Following a hot restart (not on the S7-300 and the S7-400H) the data are copied to the area defined by RD.
- Following a warm or cold restart, the data are discarded.

Communication Partner Changes to the STOP Mode

If the CPU of the communication partner changes to the STOP mode, this does not affect the data transfer with SFC 67 "X_GET." The data can also be read with the partner in the STOP mode.

Data Consistency

The data are received in a consistent state.

Error Information

See Error Information of the Communication SFCs for Non-Configured S7 Connections.

20.7 Aborting an Existing Connection to a Communication Partner outside the Local S7 Station with SFC 69 "X_ABORT"

Description

With SFC 69 "X_ABORT," you terminate a connection that was established by SFCs X_SEND, X_GET or X_PUT to a communication partner that is not in the same local S7 station. If the job belonging to X_SEND, X_GET or X_PUT is completed (BUSY = 0), the connection resources used at both ends are released after SFC 69 "X_ABORT" is called. If the job belonging to X_SEND, X_GET or X_PUT is not yet completed (BUSY = 1), call the relevant SFC again with REQ = 0 and CONT = 0 after the connection has been aborted and then wait for BUSY = 0. Only then are all the connection resources released again. You can only call SFC 69 "X_ABORT" at the end where the SFCs "X_SEND," "X_PUT" or "X_GET" are located. The connection abort is activated by calling the SFC with REQ=1.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "request to activate," see Common Parameters of the Communication SFCs
DEST_ID	INPUT	WORD	I, Q, M, D, L, constant	Address parameter "destination ID." This contains the MPI address of the communication partner. You configured this with STEP 7.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains the corresponding error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: The connection abort is not yet completed. BUSY=0: the connection abort is completed.

Changing to the STOP Mode

If the CPU changes to the STOP mode, the connection abort started with SFC 69 "X_ABORT" is completed.

Communication Partner Changes to the STOP Mode

If the CPU of the communication partner changes to the STOP mode, this does not affect the connection abort with SFC 69 "X_ABORT." The connection is terminated.

Error Information

See Error Information of the Communication SFCs for Non-Configured S7 Connections.

20.8 Reading Data from a Communication Partner within the Local S7 Station with SFC 72 "I_GET"

Description

With SFC 72 "I_GET," you can read data from a communication partner in the same local S7 station. The communication partner can be in the central rack, in an expansion rack or distributed. Make sure that you assign distributed communication partners to the local CPU with STEP 7. There is no corresponding SFC on the communication partner.

The receive job is activated after calling the SFC with REQ=1. Following this, you continue to call the SFC until the data reception is indicated by BUSY=0. RET_VAL then contains the length of the received block of data in bytes.

Make sure that the receive area defined with the RD parameter (on the receiving CPU) is at least as long as the area to be read as defined by the VAR_ADDR parameter (on the communication partner). The data types of RD and VAR_ADDR must also match.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "request to activate," see Common Parameters of the SFCs of the S7 Basic Communication.
CONT	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "continue," see Common Parameters of the SFCs of the S7 Basic Communication.
IOID	INPUT	BYTE	I, Q, M, D, L, constant	Identifier of the address range on the partner module: B#16#54= Peripheral input (PI) B#16#55= Peripheral output (PQ) The identifier of a range belonging to a mixed module is the lower of the two addresses. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical address of the partner module. If it is a mixed module, specify the lower of the two addresses.
VAR_ADDR	INPUT	ANY	I, Q, M, D	Reference to the area on the partner CPU from which the data will be read. Select a data type supported by the communication partner.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains the corresponding error code. If no error occurs, RET_VAL contains the length of the block of data copied to the receive area RD as a positive number of bytes.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: Receiving is not yet completed. BUSY=0: Receiving is completed or there is

Parameter	Declaration	Data Type	Memory Area	Description
				no receive job active.
RD	OUTPUT	ANY	I, Q, M, D	Reference to the receive area (receive data area). The following data types are allowed: BOOL, BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5_TIME, DATE_AND_TIME and arrays of these data types except for BOOL. The receive area RD must be at least as long as the area from which the data are read defined by the VAR_ADDR parameter on the communication partner. The data types of RD and VAR_ADDR must also match.

Changing to the STOP Mode

If the CPU changes to the STOP mode, the connection established by SFC 72 "I_GET" is terminated. Whether or not the received data located in a buffer of the operating system are lost depends on the type of restart performed:

- Following a hot restart (not on the S7-300 and the S7-400H), the data are copied to the area defined by RD.
- Following a warm or cold restart, the data are discarded.

Communication Partner Changes to the STOP Mode

If the CPU of the communication partner changes to the STOP mode, this does not affect the data transfer with SFC 72 "I_GET." The data can also be read with the partner in the STOP mode.

Data Consistency

The data are received in a consistent state.

Error Information

See Error Information of the Communication SFCs for Non-Configured S7 Connections.

20.9 Writing Data to a Communication Partner within the Local S7 Station with SFC 73 "I_PUT"

Description

With SFC 73 "I_PUT," you write data to a communication partner that is in the same local S7 station. The communication partner can be in the central rack, in an expansion rack or distributed. Make sure that you assign distributed communication partners to the local CPU with STEP 7. There is no corresponding SFC on the communication partner.

The send job is activated after calling the SFC with signal level 1 at the REQ control input.

Make sure that the send area defined with the SD parameter (on the sending CPU) is the same length as the receive area defined by the VAR_ADDR parameter (on the communication partner). The data types of SD and VAR_ADDR must also match.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "request to activate," see Common Parameters of the SFCs.
CONT	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "continue," see Common Parameters of the SFCs.
IOID	INPUT	BYTE	I, Q, M, D, L, constant	Identifier of the address range on the partner module: B#16#54= Peripheral input (PI) B#16#55= Peripheral output (PQ) The identifier of a range belonging to a mixed module is the lower of the two addresses. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical address of the partner module. If it is a mixed module, specify the lower of the two addresses.
VAR_ADDR	INPUT	ANY	I, Q, M, D, L	Reference to the area on the communication partner to which the data will be written. Choose a data type that is supported by the communication partner.
SD	INPUT	ANY	I, Q, M, D	Reference to the area on the local CPU that contains the data to be sent. The following data types are allowed: BOOL, BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5_TIME, DATE_AND_TIME and arrays of these data types except for BOOL. SD must be the same length as the parameter VAR_ADDR of the communication partner. The data types of SD and VAR_ADDR must also match.

Parameter	Declaration	Data Type	Memory Area	Description
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains the corresponding error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: Sending is not yet completed. BUSY=0: Sending is completed or no send function active.

Changing to the STOP Mode

If the CPU changes to the STOP mode, the connection established by SFC 73 "I_PUT" is terminated. Data can no longer be sent. If the send data have already been copied to the internal buffer when the CPU changes mode, the contents of the buffer are discarded.

Communication Partner Changes to the STOP Mode

If the CPU of the communication partner changes to the STOP mode, this does not affect the data transfer with SFC 73 "I_PUT." The data can also be written with the partner in the STOP mode.

Data Consistency

The data are sent in a consistent state.

Error Information

See Error Information of the Communication SFCs for Non-Configured S7 Connections.

20.10 Aborting an Existing Connection to a Communication Partner within the Local S7 Station with SFC 74 "I_ABORT"

Description

With SFC 74 "I_ABORT," you terminate a connection that was established by SFC 72 "I_GET" or SFC 73 "I_PUT" to a communication partner in the same local S7 station. If the job belonging to I_GET or I_PUT is completed (BUSY = 0), the connection resources used at both ends are released after SFC 74 "I_ABORT" is called.

If the job belonging to I_GET or I_PUT is not yet completed (BUSY = 1), call the relevant SFC again with REQ = 0 and CONT = 0 after the connection has been aborted and then wait for BUSY = 0. Only then are all the connection resources released again.

You can only call SFC 74 "I_ABORT" at the end where the SFC "I_PUT" or "I_GET" is located (in other words at the client end).

The connection abort is activated by calling the SFC with REQ=1.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter "request to activate," see Common Parameters of the SFCs of the S7 Basic Communication.
IOID	INPUT	BYTE	I, Q, M, D, L, constant	Identifier of the address range on the partner module: B#16#54= Peripheral input (PI) B#16#55= Peripheral output (PQ) The identifier of a range belonging to a mixed module is the lower of the two addresses. If the addresses are the same, specify B#16#54.
LADDR	INPUT	WORD	I, Q, M, D, L, constant	Logical address of the partner module. If it is a mixed module, specify the lower of the two addresses.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains the corresponding error code.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: The connection abort is not yet completed. BUSY=0: the connection abort is completed.

Changing to the STOP Mode

If the CPU changes to the STOP mode, the connection abort started with SFC 74 "I_ABORT" is completed.

Communication Partner Changes to the STOP Mode

If the CPU of the communication partner changes to the STOP mode, this does not affect the connection abort with SFC 74 "I_ABORT." The connection is terminated.

Error Information

See Error Information of the Communication SFCs for Non-Configured S7 Connections.

21 Generating Block-Related Messages

21.1 Introduction to Generating Block-Related Messages with SFBs

SFBs for Generating Block-Related Messages

You can generate a block-related message by calling one of the following SFBs in your program:

- SFB 36 "NOTIFY"
- SFB 33 "ALARM"
- SFB 35 "ALARM_8P"
- SFB 34 "ALARM_8"

These SFBs have the following properties:

- Each detected edge change causes a message to be sent.
- Following execution of the block, the associated values (inputs SD_i) are read completely and assigned to the message (see "Send and Receive Parameters" in Common Parameters of the SFCs for S7 Basic Communication).
  In terms of consistency compared with high-priority classes, the following associated values are consistent:

  the simple data types (bit, byte, word and double word)

  an array of the data type byte up to a maximum length specific to the particular CPU (see **/71/**, **/101/**).

With the status parameters DONE, ERROR and STATUS, you monitor the processing status of the block (see "Status Parameters" in Common Parameters of the SFCs for S7 Basic Communication).

Note

The parameters ID and EV_ID are only evaluated the first time the block is called (the actual parameters or the predefined values from the instance). Up to 480 bytes of work memory are required to process the messages, depending on the length and number of the SD_i parameters at the first call.

Logging On Display Devices

Before SFBs for generating block-related messages can send a message when an edge change is detected, at least one display device must be logged on for block-related messages. You can check this using SFC 62 "CONTROL."

Storing Messages

To avoid messages being lost even when there is a lot of traffic on the communication system, each SFB that generates a message can buffer two messages.

If, however, messages are lost, you are informed of this by the ERROR and STATUS output parameters (ERROR = 0, STATUS = 11). The next time that a message can be sent, the logged on display devices are also informed of the loss.

Acknowledging Messages

A centralized acknowledgment concept is used. When you have acknowledged the message at a display device, the acknowledgment information is first sent to the CPU that generated the message. From here, the acknowledgment information is distributed to all stations logged on for this purpose.

You acknowledge a signal and not an individual message. If, for example, several rising edges of a signal were indicated and you acknowledge the event entering the state, all previous events with the same message number count as having been acknowledged.

Acknowledgment Display

SFB 36 "NOTIFY" does not have an acknowledgment display. You can check the output parameters ACK_UP and ACK_DN of SFB 33 "ALARM" and the output parameter ACK_STATE of SFBs 35 "ALARM_8P" and "ALARM_8." These outputs are updated when the block is called providing the control parameter EN_R has the value 1.

Disabling and Enabling Messages

In some situations, it may be useful to suppress messages, for example, when a signal is "fluttering" or when you start up your system. You can therefore disable and enable messages at the display device or in your program. Disabling/enabling applies to all stations that logged on for the particular message. A disabled message remains disabled until it is enabled again.

You are informed of disabled messages with the ERROR and STATUS output parameters (ERROR = 1, STATUS = 21).

Message Update

With a message update, you can read out the current signal and acknowledgment states at a display device. During the update, all the logged on stations continue to receive the messages for which they logged on.

Amount of Transferable Data

The data transferred with the associated values SD_i of the NOTIFY, ALARM and ALARM_8P SFBs must not exceed a maximum length. The maximum data length is calculated as follows:

maxleng = min (pdu_local, pdu_remote) - 44 - 4 * number of SD_i parameters used

Where:

- pdu_local is the maximum length of the data fields of the local CPU (SSL_ID W#16#0131, INDEX 1, variable pdu)
- pdu_remote is the maximum length of data fields of the display devices.

Example:

A CPU 414-1 is sending messages to a PG 760 (via MPI).

The associated values SD_1, SD_2 and SD_3 are used.

pdu_local = 480 bytes, pdu_remote = 480 bytes

Number of SD_i parameters used: 3

So that:

maxleng = min (480, 480) - 44 - 4 * 3 = 480 - 44 - 12 = 424

The maximum length of data that can be transferred per SFB is 424 bytes.

Work memory requirements of the SFBs for generating block-related messages

To function smoothly, the SFBs for generating block-related messages require a temporary memory area within the CPU work memory dependent on the user data (code area). The size of the occupied memory are shown in the following table.

Module Type	Memory Required in the CPU Work Memory (in Bytes)
NOTIFY	2 x (190 + Length of the associated values specified at first call of SD_1,...SD_10)
ALARM	2 x (190 + Length of the associated values specified at first call of SD_1,...SD_10)
ALARM_8P:	2 x (190 + Length of the associated values specified at first call of SD_1,...SD_10)
ALARM_8	180
AR_SEND	108

21.2 Generating Block-Related Messages without Acknowledgment with SFB 36 "NOTIFY"

Description

SFB 36 "NOTIFY" monitors a signal. It generates a message both on a rising edge (event entering state) and on a falling edge (event leaving state). You can have up to ten associated values sent with the message. The message is sent to all stations logged on for this purpose. When the SFB is first called, a message with the current signal state is sent.

The associated values are queried when the edge is detected and assigned to the message. If you acknowledge such a message at a logged on display device, all other logged on display devices are informed. The NOTIFY block is not informed of this acknowledgment.

SFB 36 "NOTIFY" can temporarily store one rising and one falling signal edge. Any further signal changes that occur are ignored. This loss of messages is indicated with the output parameters ERROR and STATUS (ERROR = 0, STATUS = 11); the logged on display devices are also informed of this loss. SFB 36 "NOTIFY" complies with the IEC 1131-5 standard.

Parameter	Declaration	Data Type	Memory Area	Description
SIG	INPUT	BOOL	I, Q, M, D, L	The signal to be monitored
ID	INPUT	WORD	I, Q, M, D, L, constant	Data channel for messages: W#16#EEEE ID is only evaluated at the first call.
EV_ID	INPUT	DWORD	I, Q, M, D, L, constant	Message number (0 not permitted)EV_ID is only evaluated at the first call. Following this, each time SFB 36 is called with the corresponding instance DB, the message number of the first call is used. When assigning the message number, use the message configuration functions. This ensures the consistency of the message numbers.
SEVERITY	INPUT	WORD	I, Q, M, D, L, constant	Weighting of the event: Possible values: 0 through 127 (value 0 means highest weighting)
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: Generation of message completed
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter
STATUS	OUTPUT	WORD	I, Q, M, D, L	STATUS status parameter
SD_i, 1≤i ≤10	IN_OUT	ANY	I, Q, M, D, T, C	i-th associated value Only the following data types are permissible: BOOL (not allowed: bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME. **Note:** If the ANY pointer accesses an DB, the

Parameter	Declaration	Data Type	Memory Area	Description
				DB must always be specified (for example: P# DB10.DBX5.0 Byte 10).

Error Information

The following table contains all the error information specific to SFB 36 that can be output with the ERROR and STATUS parameters.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: New job not active because the previous job is not yet completed.
0	22	• Error in the pointer to the associated values SD_i: - involving the data length or the data type - associated values in the user memory not accessible, for example, due to deleted DB or area length error The activated message is sent without or eventually with the possible number of associated values • The actual parameter you have selected for SEVERITY is higher than the permitted range. The activated message will be sent with SEVERITY=127.
0	25	Communication has started. The job is being processed.
1	1	Communications problems: connection aborted or no logon
1	4	At the first call: • The specified EV_ID is outside the permitted range or • The ANY pointer SD_i has a formal error • The maximum memory area that can be sent for the CPU per SFB 36 was exceeded
1	10	Access to local user memory not possible (for example, access to a deleted DB)
1	12	When the SFB was called: • an instance DB was specified which was not initialized • an instance DB that does not belong to SFB 36 was specified • a shared DB instead of an instance DB was specified
1	18	EV_ID was already being used by one of the SFBs 31 or 33 to 36.
1	20	Not enough working memory. H-System: SFB called while update in progress
1	21	The message with the specified EV_ID is disabled

21.3 Generating Block-Related Messages with Acknowledgment with SFB 33 "ALARM"

Description

SFB 33 "ALARM" monitors a signal. It generates a message both on a rising edge (event entering state) and on a falling edge (event leaving state). You can have up to ten associated values sent with the message. The message is sent to all stations logged on for this purpose.

When the SFB is first called, a message with the current signal state is sent.

The ACK_UP output is reset when there is a rising edge and message generation is complete (DONE = 1). It is set when your acknowledgment of the event entering the state has arrived from a logged on display device. The situation for the ACK_DN output is analogous: this is reset when there is a falling edge and message generation is complete (DONE = 1). It is set when your acknowledgment of the event leaving the state is received from a logged on display device. Once your acknowledgment has been received from a logged on display device, the acknowledgment information is passed on to all other stations logged on for this purpose.

One message memory with two memories areas is available per instance of SFB33 "ALARM". As long as the job that belongs to the message is not initialized because the previous message is still being processed, the memory can be overwritten again. This is the case when a further message is generated. This message loss is displayed for you in the output parameters ERROR and STATUS (ERROR = 0, STATUS = 11). The logged displaying devices also receive a corresponding message with the next message that can be transmitted.

SFB 33 "ALARM" complies with the IEC 1131-5 standard.

Parameter	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter enabled to receive that decides whether the outputs ACK_UP and ACK_DN are updated at the first block call (EN_R=1) or not (EN_R=0). If EN_R=0 SFB 33 "ALARM" operates in the same way as SFB 36 "NOTIFY." The output parameters ACK_UP and ACK_DN remain the same in this case.
SIG	INPUT	BOOL	I, Q, M, D, L	The signal to be monitored
ID	INPUT	WORD	I, Q, M, D, L, constant	Data channel for messages: W#16#EEEE ID is only evaluated at the first call.
EV_ID	INPUT	DWORD	I, Q, M, D, L, constant	Message number (0 not permitted) EV_ID is only evaluated at the first call. Following this, each time SFB 33 is called with the corresponding instance DB, the message number of the first call is used. When assigning the message

Parameter	Declaration	Data Type	Memory Area	Description
				number, use the message configuration functions. This ensures the consistency of the message numbers.
SEVERITY	INPUT	WORD	I, Q, M, D, L, constant	Weighting of the event Possible values: 0 through 127 (value 0 means highest weighting)
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: Generation of message completed
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter
STATUS	OUTPUT	WORD	I, Q, M, D, L	STATUS status parameter
ACK_DN	OUTPUT	BOOL	I, Q, M, D, L	Event leaving state was acknowledged on a display device
ACK_UP	OUTPUT	BOOL	I, Q, M, D, L	Event entering state was acknowledged on a display device
SD_i, 1≤i ≤10	IN_OUT	ANY	I, Q, M, D, T, C	i-th associated value Only the following data types are permissible: BOOL (not allowed: bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME. **Note:** If the ANY pointer accesses an DB, the DB must always be specified (for example: P# DB10.DBX5.0 Byte 10).

Error Information

The following table contains all the error information specific to SFB 33 that can be output with the ERROR and STATUS parameters.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: New job not active because the previous job is not yet completed.
0	22	• Error in the pointer to the associated values SD_i: - - Involving the data length or the data type - - Associated values in the user memory not accessible, for example, due to deleted DB or area length error - - The activated message is sent without associated values • The actual parameter you have selected for SEVERITY is higher than the permitted range. The activated message will be sent with SEVERITY=127.
0	25	Communication has started. The job is being processed.
1	1	Communications problems: connection aborted or no logon
1	4	• At the first call: - - The specified EV_ID is outside the permitted range or - - The ANY pointer SD_i has a formal error - - The maximum memory area that can be sent for the CPU per SFB 33 was exceeded

ERROR	STATUS (Decimal)	Explanation
1	10	Access to local user memory not possible (for example, access to a deleted DB)
1	12	When the SFB was called: - -An instance DB was specified that was not initialized - -An instance DB that does not belong to SFB 33 was specified - -A shared DB instead of an instance DB was specified
1	18	EV_ID was already being used by one of the SFBs 31 or 33 to 36.
1	20	Not enough working memory. H-System: Call of the FSB during update
1	21	The message with the specified EV_ID is disabled

Note

After the first block call, the ACK_UP and ACK_DN outputs have the value 1 and it is assumed that the previous value of the SIG input was 0.

21.4 Generating Block-Related Messages with Accompanying Values for Eight Signals with SFB 35 "ALARM_8P"

Description

SFB 35 "ALARM_8P" is a straight extension of SFB 33 "ALARM" allowing eight signals.

A message is generated when an edge change is detected at one or more signals (exception: a message is always sent at the first block call). All eight signals have a common message number that is broken down into eight individual messages on the display device. You can acknowledge each individual message separately or all eight individual messages at once.

You can use the ACK_STATE output parameter to incorporate the acknowledgment state of the individual messages in your program. If you disable or enable a message of an ALARM_8P block, this always affects the entire ALARM_8P block. Disabling and enabling of individual signals is not possible.

One message memory with two memories is available per instance of SFB35 "ALARM_8P". As long as the job that belongs to the message is not initialized because the previous message is still being processed, the memory can be overwritten again. This is the case when a further message is generated. This message loss is displayed for you in the output parameters ERROR and STATUS (ERROR = 0, STATUS = 11). The logged displaying devices also receive a corresponding message with the next message that can be transmitted.

Parameters	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter enabled to receive that decides whether the output ACK_STATE is updated at the block call (EN_R=1) or not (EN_R=0).
SIG_i, 1 = i = 8	INPUT	BOOL	I, Q, M, D, L	i(th) signal to be monitored
ID	INPUT	WORD	I, Q, M, D, L, constant	Data channel for messages: W#16#EEEE ID is only evaluated at the first call.
EV_ID	INPUT	DWORD	I, Q, M, D, L, constant	Message number (0 not permitted)EV_ID is only evaluated at the first call. Following this, each time SFB 35 is called with the corresponding instance DB, the message number of the first call is used. When assigning the message number, use the message configuration functions. This ensures the consistency of the message numbers.
SEVERITY	INPUT	WORD	I, Q, M, D, L, constant	Weighting of the event Possible values: 0 through 127 (value 0 means highest weighting)
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: Generation of message completed
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter
STATUS	OUTPUT	WORD	I, Q, M, D, L	STATUS status parameter
ACK_STATE	OUTPUT	WORD	I, Q, M, D, L	Bit field with the current acknowledgment status of all eight messages: • Bit 2^0 event entering state at SIG_1 was acknowledged • Bit 2^7 : event entering state at SIG_8 was acknowledged • Bit 2^8 : event leaving state at SIG_1 was acknowledged • Bit 2^{15} : event leaving state at SIG_8 was acknowledged
SD_j, $1 \leq j \leq 10$	IN_OUT	ANY	I, Q, M, D, T, C	j-th associated value The accompanying values apply for all messages. Only the following data types are permissible: BOOL (not allowed: bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME. **Note:** If the ANY pointer accesses an DB, the

Parameters	Declaration	Data Type	Memory Area	Description
				DB must always be specified (for example: P# DB10.DBX5.0 Byte 10).

Error Information

The following table contains all the error information specific to SFB 35 that can be output with the ERROR and STATUS parameters.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: New job not active because the previous job is not yet completed.
0	22	• Error in the pointer to the associated values SD_i: - involving the data length or the data type - associated values in the user memory not accessible, for example, due to deleted DB or area length error The activated message is sent without associated values • The actual parameter you have selected for SEVERITY is higher than the permitted range. The activated message will be sent with SEVERITY=127.
0	25	Communication has started. The job is being processed.
1	1	Communications problems: connection aborted or no logon
1	4	At the first call: • The specified EV_ID is outside the permitted range or • The ANY pointer SD_i has a formal error • The maximum memory area that can be sent for the CPU per SFB 35 was exceeded
1	10	Access to local user memory not possible (for example, access to a deleted DB)
1	12	When the SFB was called: • An instance DB was specified that was not initialized • An instance DB that does not belong to SFB 35 was specified • A shared DB instead of an instance DB was specified
1	18	EV_ID was already being used by one of the SFBs 31 or 33 to 36.
1	20	Not enough working memory. H-System: SFB called while update in progress
1	21	The message with the specified EV_ID is disabled.

Note

After the first block call. all the bits of the ACK_STATE output are set and it is assumed that the previous values of inputs SIG_i, $1 \leq i \leq 8$ were 0.

21.5 Generating Block-Related Messages without Accompanying Values for Eight Signals with SFB 34 "ALARM_8"

Description

SFB 34 "ALARM_8" is identical to SFB 35 "ALARM_8P" except that it does not have the associated values SD_1 through SD_10.

Parameter	Declaration	Data Type	Memory Area	Description
EN_R	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter enabled to receive that decides whether the output ACK_STATE is updated (EN_R=1) when the block is called or not (EN_R=0).
SIG_i, 1≤i≤8	INPUT	BOOL	I, Q, M, D, L	i(th) signal to be monitored
ID	INPUT	WORD	I, Q, M, D, L, constant	Data channel for messages: W#16#EEEE ID is only evaluated at the first call.
EV_ID	INPUT	DWORD	I, Q, M, D, L, constant	Message number (0 not permitted) EV_ID is only evaluated at the first call. Following this, each time SFB 34 is called with the corresponding instance DB, the message number of the first call is used. When assigning the message number, use the message configuration functions. This ensures the consistency of the message numbers.
SEVERITY	INPUT	WORD	I, Q, M, D, L, constant	Weighting of the event Possible values: 0 through 127 (value 0 means highest weighting)
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: Generation of message completed
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter
STATUS	OUTPUT	WORD	I, Q, M, D, L	STATUS status parameter
ACK_STATE	OUTPUT	WORD	I, Q, M, D, L	Bit field with the current acknowledgment status of all eight messages: • Bit 2^0 : event entering state at SIG_1 was acknowledged • Bit 2^7 : event entering state at SIG_8 was acknowledged • Bit 2^8 : event leaving state at SIG_1 was acknowledged • Bit 2^{15} : event leaving state at SIG_8 was acknowledged

Error Information

The following table contains all the error information specific to SFB 34 that can be output with the ERROR and STATUS parameters.

ERROR	STATUS (Decimal)	Explanation
0	11	Warning: New job not active because the previous job is not yet completed.
0	22	The actual parameter you have selected for SEVERITY is higher than the permitted range. The activated message is sent with SEVERITY = 127.
0	25	Communication has started. The job is being processed.
1	1	Communications problems: connection abort or no logon
1	4	At the first call, the specified EV_ID is outside the permitted range.
1	10	Access to local user memory not possible (for example, access to a deleted DB)
1	12	When the SFB was called: • An instance DB was specified that was not initialized • An instance DB that does not belong to SFB 34 was specified • A shared DB instead of an instance DB was specified
1	18	EV_ID was already being used by one of the SFBs 31 or 33 to 36.
1	20	Not enough working memory. H-System: SFB called while update in progress
1	21	The message with the specified EV_ID is disabled

Note

After the first block call. all the bits of the ACK_STATE output are set and it is assumed that the previous values of inputs SIG_i, $1 \leq i \leq 8$ were 0.

21.6 Sending Archive Data with SFB 37 "AR_SEND"

Description

SFB 37 "AR_SEND" sends archive data to operator interface systems logged on for this purpose. These systems inform the CPU of the relevant archive number in the logon message. Depending on the memory available on the CPU and the address area used, the archive data can be up to 65534 bytes long. The defaults of the operator interface system you are using must be taken into consideration in the structure of the archive data.

The sending of the data is activated by a positive edge at control input REQ after the block has been called. The start address of the archive data is specified by SD_1, the length of the data field by LEN. Data transfer is asynchronous to the execution of the user program. Successful completion of the transfer is indicated by the DONE status parameter having the value 1. A rising edge at control input R

aborts the transfer of data.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	Control parameter request
R	INPUT	BOOL	I, Q, M, D, L, constant	Control parameter reset: current job aborted
ID	INPUT	WORD	I, Q, M, D, L, constant	Data channel for messages: W#16#EEEE ID is only evaluated at the first call.
AR_ID	INPUT	DWORD	I, Q, M, D, L, constant	Archive number (0 not permitted)AR_ID is only evaluated at the first call. Following this, each time SFB 37 is called with the corresponding instance DB, the archive number from the first call is used. When assigning the archive number, use the message configuration functions. This ensures the consistency of the archive numbers.
DONE	OUTPUT	BOOL	I, Q, M, D, L	DONE status parameter: sending completed
ERROR	OUTPUT	BOOL	I, Q, M, D, L	ERROR status parameter
STATUS	OUTPUT	WORD	I, Q, M, D, L	STATUS status parameter
SD_1	IN_OUT	ANY	I, Q, M, D, T, C	Pointer to archive data. The length specification is not evaluated. Only the following data types are permissible: BOOL (not allowed: bit field), CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME The archive data have to have a PLC specific structure. **Note:** If the ANY pointer accesses an DB, the DB must always be specified (for example: P# DB10.DBX5.0 Byte 10).
LEN	IN_OUT	WORD	I, Q, M, D, L	Length of the data field to be sent in bytes

Error Information

The following table contains all the error information specific to SFB 37 that can be output with the ERROR and STATUS parameters.

ERROR	STATUS(Decimal)	Explanation
0	11	Warning: New job not active because the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems
1	2	Negative acknowledgment, function cannot be executed
1	3	There is no logon for the specified AR_ID.

ERROR	STATUS(Decimal)	Explanation
1	4	• Error in the archive data pointer SD_1 involving data length or data type. • At the first call, the specified AR_ID is outside the permitted range.
1	5	Requested reset was executed.
1	7	RESET job irrelevant because the current function was completed or not activated (block in incorrect status).
1	10	Access to local user memory not possible(for example, access to a deleted DB).
1	12	When the SFB was called: • An instance DB was specified that was not initialized • An instance DB that does not belong to SFB 37 was specified • A shared DB instead of an instance DB was specified
1	18	AR_ID was already being used by an SFB 37.
1	20	Not enough working memory. H-System: SFB called while update in progress

21.7 Disabling Block-Related, Symbol-Related and Group Status Messages with SFC 10 "DIS_MSG"

Description

With SFC 10 "DIS_MSG" (disable message) you can disable block-related messages generated with SFBs, symbol-related messages (SCAN) and group status messages. You select messages to be disabled using the input parameters MODE and MESGN. Calling SFC 10 "DIS_MSG" and successfully disabling a message is only possible when the disabling of a message is not already active with SFC 10.

Messages that are ready to be sent when SFC 10 is called but that are still in an internal buffer can no longer be disabled and are sent. A disabled message is indicated at the ERROR and STATUS outputs of the "NOTIFY", "ALARM", "ALARM_8P" and "ALARM_8" SFBs.

You start the disabling of a message by assigning the value 1 to the REQ input parameter when SFC 10 is called.

How SFC 10 Functions

Disabling is executed asynchronously, in other words it can be active throughout several SFC 10 calls:

- When it is first called (REQ =1), SFC 10 checks the input parameters and attempts to occupy the required system resources. If successful, the value W#16#7001 is entered in RET_VAL, BUSY is set and disabling the message is started.
  If unsuccessful, the error information is entered in RET_VAL and the job is terminated. BUSY must not be evaluated in this case.

- If there are further calls in the meantime, the value W#16#7002 is entered in RET_VAL (job still being executed by the CPU) and BUSY is set. Further calls do not affect the current job.
- The last time the SFB is called, the value W#16#0000 is entered in RET_VAL if no error occurred. BUSY then has the value 0. If an error occurred, the error information is entered in RET_VAL and BUSY must not be evaluated.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	REQ = 1: trigger disable
MODE	INPUT	BYTE	I, Q, M, D, L, constant	Parameter for selecting the messages to be disabled, see following table
MESGN	INPUT	DWORD	I, Q, M, D, L, constant	Message number only relevant when MODE is set to 5, 6, 7. This allows a single message to be disabled.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information, see table "Error Information"
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: disable has not yet been canceled.

MODE Input Parameter

The table below shows the permitted values for the MODE input parameter:

Value	Meaning
0	All block-related, all symbol-related and all group status messages of the CPU generated with SFBs
1	All block-related messages of the CPU generated with SFBs, in other words all messages generated by the "NOTIFY", "ALARM", "ALARM_8P" and "ALARM_8" SFBs
2	All group status messages of the CPU
3	All symbol-related messages of the CPU (SCAN)
5	Single message of the "symbol-related messages" class
6	Single message of the "block-related messages" class
7	Single message of the "group status messages" class

Error Information

Error Code (W#16#...)	Explanation
0000	Disabling was terminated without an error.
7000	REQ = 0 at first call: disabling was not activated.
7001	REQ = 1 at first call: disabling was triggered.
7002	Further call: disabling is already active.
8081	Error accessing a parameter

Error Code (W#16#...)	Explanation
8082	MODE has an illegal value.
8083	The message number is outside the permitted range of values.
8084	There is no logon for the message(s) specified with MODE and possibly MESGN.
80C3	The message(s) to be disabled in MODE and possibly MESGN, cannot be disabled at present - SFC 10 is already disabling messages.

21.8 Enabling Block-Related, Symbol-Related, and Group Status Messages with SFC 9 "EN_MSG"

Description

With SFC 9 "EN_MSG" (enable message), you can enable block-related, symbol-related and group status messages that were previously disabled. You disabled the messages either at a display device or using SFC 10 "DIS_MSG."
You specify the messages to be enabled using the MODE and MESGN input parameters. Successful enabling of messages with SFC 9 "EN_MSG" is only possible when SFC 9 is not already actively enabling messages.
You start the enabling function by assigning the value 1 to the REQ input parameter of SFC 9.

How SFC 9 Functions

Enabling is executed asynchronously, in other words it can be active throughout several SFC 9 calls:

- When it is first called (REQ =1), SFC 9 checks the input parameters and attempts to occupy the required system resources. If successful, the value W#16#7001 is entered in RET_VAL, BUSY is set and enabling the message is started.
  If unsuccessful, the error information is entered in RET_VAL and the job is terminated. BUSY must not be evaluated in this case.
- If there are further calls in the meantime, the value W#16#7002 is entered in RET_VAL (job still being executed by the CPU) and BUSY is set. Further calls do not affect the current job.
- The last time the SFB is called, the value W#16#0000 is entered in RET_VAL if no error occurred. BUSY then has the value 0. If an error occurred, the error information is entered in RET_VAL and BUSY must not be evaluated.

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L	REQ = 1: trigger enable
MODE	INPUT	BYTE	I, Q, M, D, L, constant	Parameter for selecting the messages to be enabled
MESGN	INPUT	DWORD	I, Q, M, D, L, constant	Message number only relevant when MODE is set to 5, 6, 7. This allows a single

Parameter	Declaration	Data Type	Memory Area	Description
				message to be enabled.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information, see table.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY = 1: the enable has not yet been canceled.

MODE Input Parameter

The following table shows the permitted values for the MODE input parameter.

Value	Meaning
0	All block-related, all symbol-related and all group status messages of the CPU generated with SFBs
1	All block-related messages of the CPU generated with SFBs, in other words all messages generated by the "NOTIFY," "ALARM," "ALARM_8P" and "ALARM_8" SFBs
2	All group status messages of the CPU
3	All symbol-related messages of the CPU (SCAN)
5	Single message of the "symbol-related messages" class
6	Single message of the "block-related messages" class
7	Single message of the "group status messages" class

Error Information

Error Code (W#16#...)	Explanation
0000	Disabling was terminated without an error.
7000	REQ = 0 at first call: enabling was not activated.
7001	REQ = 1 at first call: enabling was triggered.
7002	Further call: enabling is already active.
8081	Error accessing a parameter
8082	MODE has an illegal value.
8083	The message number is outside the permitted range of values.
8084	There is no logon for the message(s) specified with MODE and possibly MESGN.
80C3	The message(s) to be disabled in MODE and possibly MESGN, cannot be disabled at present - SFC 9 is already enabling messages.

21.9 Startup Behavior of the SFBs for Generating Block-Related Messages

Warm Restart and Cold Restart

During a warm or cold restart, the SFBs for generating block-related messages are set to the NO_INIT status. The actual parameters stored in the instance DBs are unchanged.

Hot Restart

During a hot restart, the SFBs for generating block-related messages behave like user function blocks that are capable of resuming execution. They continue from the point of interruption.

Memory Reset

A memory reset always causes the termination of all connections so that no station is logged on for messages. The user program is deleted. If you have inserted a FLASH card, the program sections relevant to execution are loaded on the CPU again from the card and the CPU executes a warm or cold restart (implicitly this is always a cold restart, since all user data are initialized after clearing memory).

21.10 How the SFBs for Generating Block-Related Messages React to Problems

Connection Breakdown

The connections assigned to the SFB instances are monitored for breakdown. If a connection breaks down, the stations involved are removed from the internal CPU list of stations logged on for block-related messages. Any messages pending for these stations are deleted.

If other stations are still logged on following a connection breakdown, they continue to receive messages. The SFBs only stop sending messages when there are no more connections to any logged on stations. The ERROR and STATUS output parameters indicate this situation (ERROR = 1, STATUS = 1).

Error Interface to the User Program

If an error occurs during the execution of an SFB for generating block-related messages, the SFB changes to the ERROR or ERROR_E status. At the same time, the ERROR output parameter is set to 1 and the STATUS output parameter has the corresponding error identifier. You can evaluate this error information in your program.

Examples of possible errors:

- Sending not possible due to lack of resources
- Error accessing one of the signals to be monitored.

21.11 Introduction to Generating Block-Related Messages with SFCs

SFCs for Generating Block-Related Messages

You can generate a block-related message with the following SFCs:

- SFC 17 "ALARM_SQ"
- SFC 18 "ALARM_S"
- SFC 107 "ALARM_DQ"
- SFC 108 "ALARM_D"

These SFCs have the following properties:

- The messages sent by SFC 17 "ALARM_SQ" and SFC 107 "ALARM_DQ" when the signal state is 1 can be acknowledged at a logged on display device. The messages of SFC 18 "ALARM_S" and SFC 108 "ALARM_D" are always implicitly acknowledged. It is not a detected edge change that generates a message but rather each SFC call. For more detailed information refer to Generating Acknowledgeable Block-Related Messages with SFC 17 "ALARM_SQ" and Permanently Acknowledged Block-Related Messages with SFC 18 "ALARM_S
- "ALARM_S" and the section Generating Acknowledgeable Block-Related Messages with the SFCs 17 "ALARM_DQ" and 108 "ALARM_D".
- Following execution of the block, the associated value SD_1 is read completely and assigned to the message. In terms of consistency compared with high-priority classes, the following associated values are consistent:
  - the simple data types (bit, byte, word, and double word)
  - an array of the data type byte up to a maximum length specific to the particular CPU (see /71/, /101/).

SFC 19 "ALARM_SC"

Using SFC 19 "ALARM_SC" you can query the following:

- The acknowledgment status of the last "entering state message" and the signal state at the last SFC 17/SFC 107 call, or
- The signal state at the last SFC 18/SFC 108 call.

Logging On Display Devices

The SFCs for generating block-related messages only send a message when they are called if at least one display device has logged on for block-related messages.

Message Storage

To avoid messages being lost when there is a lot of traffic on the communications system, the SFCs 17, 18, 107 and 108 can both buffer two messages.
If, however, messages are lost, you are informed in RET_VAL. The logged on display devices are informed of this the next time a message can be sent.

Message Acknowledgment with the SFCs 17 "ALARM_SQ" and 107 "ALARM_DQ"

If you have acknowledged an "entering event message" at a display device, this acknowledgment information is first sent to the CPU where the message originated. This then distributes the acknowledgment information to all stations logged on for this purpose.

Disabling and Enabling Messages

Block-related messages generated with SFC 17 "ALARM_SQ", SFC 18 "ALARM_S", SFC 107 "ALARM_DQ" or SFC 108 "ALARM_D" cannot be disabled and then enabled again.

Message Update

At a display device, you can use a message update to read out the current signal and acknowledgment status. During the update, all the logged on stations continue to receive the messages for which they logged on.

Changes in Your Program that contains the SFC 17/SFC 18 calls

Note

When you download a block that is already on the CPU using SFC 17/SFC 18 calls, it is possible that the previous block has sent an entering state message but that the new block does not send a corresponding leaving state message. This means that the message remains in the internal message memory of the CPU. This situation can also occur when you delete blocks with SFC 17/SFC 18. You can remove such messages from the internal message memory of the CPU by changing the CPU to STOP and then going through a warm or cold restart.

Changes in Your Program that contains the SFC 17/SFC 18 calls

Even though your program might contain SFC 107 and/or SFC 108 calls, the described program modifications may cause the messages to become resident in the internal message memory and thus permanently occupy system resources.

Contrary to system resources which were occupied by SFC 17/SFC 18 calls, you can release system resources occupied by SFC 107/SFC 108 calls without having to switch your CPU to STOP mode. This is carried out by using SFC 106 "DEL_SI", see Releasing Dynamically Occupied System Resources with SFC 106 "DEL_SI" . Before you release dynamically occupied system resources by calling SFC 106

"DEL_SI", it may be appropriate to fetch information on currently dynamically occupied system resources of your CPU, with the help of SFC 105 "READ_SI", see Reading Dynamically Occupied System Resources with SFC 105 "READ_SI".

Also refer to:

Configuring Messages

Amount of transferable data

The amount of data transferable using the accompanying value SD for SFCs ALARM_S, ALARM_SQ, ALARM_D and ALARM_DQ cannot exceed a maximum length. This data length is calculated as follows:

maxleng = min (pdu_local, pdu_remote) - 48

Definitions:

- pdu_local: the maximum length for CPU data blocks (SZL_ID W#16#0131, INDEX 1, Variable pdu)
- pdu_remote: the maximum length for display device data blocks

Example:

A CPU 414-1 sends a message to a programmingdevice PG 760 (via MPI).

pdu_local = 480 Byte, pdu_remote = 480 bytes,

Result:

maxleng = min (480, 480) - 48 = 480 - 48 = 432

The maximum transferable data length per SFC is thus 432 bytes.

21.12 Generating Acknowledgeable Block-Related Messages with SFC 17 "ALARM_SQ" and Permanently Acknowledged Block-Related Messages with SFC 18 "ALARM_S"

Note

Newly created programs should only use SFCs 107 and 108 since they provide improved options for managing system resources.

Description

Each time they are called, SFC 17 "ALARM_SQ" and SFC 18 "ALARM_S" generate a message to which you can add associated values. The message is sent

to all stations that have logged on for the message. SFC 17 and SFC 18 provide you with a simple mechanism for sending messages. You must make sure that you only call SFC 17 or SFC 18 when the value of the triggering signal SIG is inverted compared with the last call. If this is not the case, this is indicated in RET_VAL and no message is sent. The very first time that SFC 17 or SFC 18 is called, you must make sure that the SIG input has the value 1. Otherwise, RET_VAL contains error information and no message will be sent.

Note

Call SFC 17 and SFC 18 in an FB to which you have previously assigned suitable system attributes! For more detailed information about assigning system attributes to blocks, refer to **/232/** and **/233/**.

Use of system resources

When generating messages with the SFCs 17 "ALARM_SQ" and 18 "ALARM_S", the operating system uses one system resource for the duration of the signal cycle.

For SFC 18 "ALARM_S", the signal cycle lasts from the SFC call SIG=1 until another call with SIG=0. For SFC 17 "ALARM_SQ", this time period also includes the time until the incoming signal is acknowledged by one of the reported display devices, if necessary.

If, during the signal cycle, the message-generating block is overloaded or deleted, the associated system resource remains occupied until the next restart (warm restart).

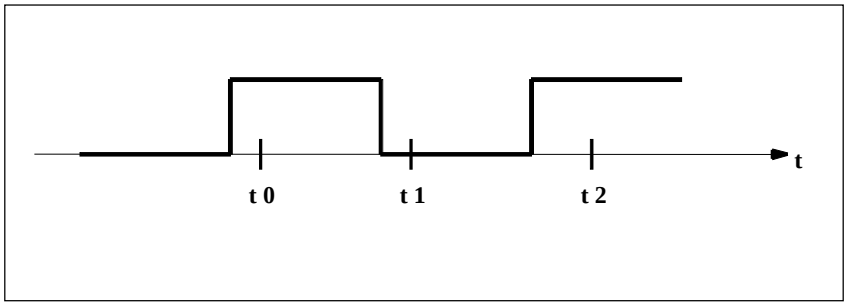
Acknowledging Messages

You can acknowledge messages sent by SFC 17 "ALARM_SQ" when the state of the monitored signal is 1. You can query the acknowledgment status of the last "entering event message" and the signal state at the last SFC call using SFC 19 "ALARM_SC." Messages you have sent with SFC 18 "ALARM_S" are always implicitly acknowledged. You can query the signal state at the last SFC 18 call using SFC 19 "ALARM_SC."

Temporary Storage of Signal States

SFC 17 "ALARM_SQ" and SFC 18 "ALARM_S" temporarilyoccupy system resources. Here, they enter among other things the last two signal states including the time stamp and associated value. If SFC 17 or SFC 18 is called at a time when the signal states of the two last "valid" SFC calls have not yet been sent (signal overflow), the current and the last signal state are discarded and an overflow ID is set in the buffer. At the next possible opportunity, the second but last signal and the overflow identifier are sent.

Example:



t0, t1 and t2 are the points at which SFC 17 or SFC 18 are called. If the signal states of t0 and t1 are not sent at the time t2, the signal states of t1 and t2 are discarded and the overflow identifier is set for the signal state of t0.

Instance Overflow

If the number of SFC 17 or SFC 18 calls is higher than the maximum amount of CPU system resources, this may result in a lack of resources (instance overflow). This is indicated both by the information in RET_VAL as well as by indications at the logged on display devices.

The maximum number of SFC 17 or SFC 18 calls depends on the particular CPU. You will find this information in **/70/** and **/101/** .

Parameter	Declaration	Data Type	Memory Area	Description
SIG	INPUT	BOOL	I, Q, M, D, L	The signal to trigger a message
ID	INPUT	WORD	I, Q, M, D, L, constant	Data channel for messages: W#16#EEEE
EV_ID	INPUT	DWORD	I, Q, M, D, L, constant	Message number (0 not permitted)When assigning the message number, use the message configuration functions. This ensures the consistency of the message numbers.
SD	INPUT	ANY	I, Q, M, D, T, C	Associated value Maximum length: 12 bytes The following data types are permitted BOOL (not permitted: bit field) BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
0001	• The associated value is longer than the maximum permitted length or • Access to the user memory is not possible (for example, access to a deleted DB). The message is sent.
0002	Warning: the last free message acknowledgment memory has been used.
8081	The specified EV_ID is outside the permitted range.
8082	Loss of messages since your CPU has no more resources for generating block-related messages by SFCs.
8083	Message loss since the same signal change already exists but could not yet be sent (signal overflow).
8084	The signal that triggered the message (SIG) had the same value at the current SFC 17 or SFC 18 call as at the last call.
8085	No logon for the specified EV_ID
8086	An SFC call for the specified EV_ID is already being executed in a lower priority class.
8087	When SFC 17 or SFC 18 were first called, the message trigger signal had the value 0.
8088	The specified EV_ID is already being used by another SFC type that is currently (still) occupying memory.

21.13 Querying the Acknowledgment Status of the Last ALARM_SQ/ALARM_DQ Entering Event Message with SFC 19 "ALARM_SC"

Description

With SFC 19 "ALARM_SC" you can query the following:

- The acknowledgment status of the last ALARM_SQ/ALARM_DQ entering state message and the status of the signal that triggered the message the last time that SFC 17 "ALARM_SQ"/SFC 107 "ALARM_DQ" was called, or
- The status of the signal that triggered the message the last time SFC 18 "ALARM_S"/SFC 108 "ALARM_D" was called.

Assuming that you assigned the message numbers during message configuration, the message or signal is referenced with a unique message number SFC 19 "ALARM_SC" accesses the temporarily occupied memory of SFC 17 or SFC 18/SFC 107/SFC 108.

Parameter	Declaration	Data Type	Memory Area	Description
EV_ID	INPUT	DWORD	I, Q, M, D, L, constant	Message number for the signal state at the last SFC call or the acknowledgment status of the last entering state message (only with SFC 17 and SFC 107!) that you want to query.
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error information
STATE	OUTPUT	BOOL	I, Q, M, D, L	State of the signal that triggered the message at the last SFC call
Q_STATE	OUTPUT	BOOL	I, Q, M, D, L	If the specified EV_ID parameter belongs to an SFC 18/SFC 108 call: 1
				If the specified EV_ID parameter belongs to an SFC 17/SFC 107 call: acknowledgment status of the last entering state message: 0: Not acknowledged 1: Acknowledged

Error Information

Error Code (W#16#...)	Explanation
0000	No error occurred.
8081	The specified EV_ID is outside the permitted range.
8082	No memory is currently occupied for this EV_ID (possible cause: the corresponding signal state was not yet 1, or the signal state has already returned to 0).

21.14 Generating Acknowledgeable and Always Acknowledged Block Related Messages with SFCs 107 "ALARM_DQ" and 108 "ALARM_D"

Description

With every call the SFCs 107 "ALARM_DQ" and 108 "ALARM_D" generate a message to which you can append an associated value. Thus, you correspond with SFCs 17 "ALARM_SQ" and 18 "ALARM_S".

When generating messages with SFCs 107 "ALARM_DQ" and 108 "ALARM_D", the operating system temporarily occupies a system resource for the duration of the signal cycle.

Der Signalzyklus dauert bei der SFC 108 "ALARM_D" vom SFC-Aufruf mit SIG=1 bis zum erneuten Aufruf mit SIG=0. Bei der SFC 107 "ALARM_DQ" kommt zu dieser Zeitspanne ggf. noch die Zeit bis zur Quittierung des kommenden Signals durch eines der angemeldeten Anzeigegeräte hinzu.

For SFC 108 "ALARM_D", the signal cycle lasts from the SFC call SIG=1 until another call with SIG=0. For SFC 107 "ALARM_DQ", this time period also includes the time until the incoming signal is acknowledged by one of the reported display devices, if necessary.

If, during the signal cycle, the message-generating block is overloaded or deleted, the associated system resource remains occupied until the next restart (warm restart).

The additional functionality of SFCs 107 "ALARM_DQ" and 108 "ALARM_D"compared to SFCs 17 and 18 is now that you can manage these occupied system resources:

- With the help of SFC 105 "READ_SI" you can fetch information related to occupied system resources.
- With SFC 106 "DEL_SI" you can release occupied system resources again. This is of special significance for permanently occupied system resources. A currently occupied system resource, for example, stays occupied until the next restart (warm restart) if you, in the course of a program change, delete an FB call that contains SFC107 or SFC108 calls . When you change the program, and reload an FB with SFC 107 or SFC 108 calls, it may happen that the SFCs 107 and 108 do not generate anymore messages.

The SFCs 107 and 108 contain one parameter more than the SFCs 17 and 18, namely the input CMP_ID. Use this input to assign the messages generated with SFCs 107 and 108 to logical areas, for example to parts of the system. If you call SFC 107/SFC 108 in an FB the obvious thing to do is to assign the number of the corresponding instance DB to CMP_ID.

Parameters	Declaration	Data type	Memory Area	Description
SIG	INPUT	BOOL	I, Q, M, D, L	The message triggering signal
ID	INPUT	WORD	I, Q, M, D, L, Const.	Data channel for messages: W#16#EEEE

Parameters	Declaration	Data type	Memory Area	Description
EV_ID	INPUT	DWORD	I, Q, M, D, L, Const.	Message number (not allowed: 0) Utilize the message configuration when you assign the message number. This ensures consistency for the message numbers.
CMP_ID	INPUT	DWORD	I, Q, M, D, L, Const.	Component identifier (not allowed: 0) ID for the partial system to which the corresponding message is assigned Recommended values: • low word: 1 to 65535 • high word: 0 You will not be confronted with any conflicts with the SIEMENS program package if you are compliant with these recommendations.
SD	INPUT	ANY	I, Q, M, D, T, C	Associated value Maximum length: 12 bytes Permitted are only data of the type BOOL (not allowed: Bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME
RET_VAL	OUTPUT	INT	E, A, M, D, L	Error Information

Error Information

Error code (W#16#...):	Explanation
0000	No error occurred.
0001	• The length of the associated value exceeds the maximum permissible length, or • Access to user memory not possible (for example, access to deleted DB) The activated message is sent.
0002	Warning: The last free message acknowledge memory was occupied.
8081	The specified EV_ID lies outside the valid range.
8082	Message loss because your CPU has no more resource for generating block related messages with SFCs.
8083	Message loss, the same signal transition is already present but could not be sent yet (signal overflow).
8084	With the current and the previous SFC 107-/SFC-108 call the message triggering signal SIG has the same value.
8085	There is no logon for the specified EV_ID.
8086	An SFC call for the specified EV_ID is already being processed in a lower priority class.
8087	At the initial call of SFC 107/SFC 108 the message triggering signal had the value 0.
8088	The specified EV_ID is already in use by another SFC type that currently (still) occupies memory space.

Error code (W#16#...):	Explanation
8089	You have assigned the value 0 to CMP_ID.

21.15 Reading Dynamically Occupied System Resources with SFC 105 "READ_SI"

How dynamically occupied system resources develop when generating messages with the SFCs 107 and 108

When messages are generated with SFCs 107 "ALARM_DQ" and 108 "ALARM_D", the operating system occupies temporarily system memory space.

For example, if you do not delete a FB that exists in the CPU with SFC 107 or SFC 108 calls it may happen that corresponding system resources stay permanently occupied. If you reload the FB with SFC 108 or SFC 108 calls, it may happen that the SFCs 107 and 108 are not processed properly anymore.

Description

With SFC 105 "READ_SI" you can read currently used system resources occupied with the SFCs 107 and 108 when messages were generated. This is done via the values of EV_ID and CMP_ID used in this place. The values are passed on to SFC 105 "READ_SI" in parameter SI_ID.

SFC 105 "READ_SI" has four possible operating modes that we explain in the table below. Set the desired operating mode via the MODE parameter.

MODE	Which of the system resources occupied by SFC 107/SFC 108 are read?
1	All (call of SFC 105 with SI_ID:=0)
2	The system resource occupied by the call of SFC 107-/SFC 108 with EV_ID:=ev_id (call of the SFC 105 with SI_ID:=ev_id)
3	The system resource occupied by the call of SFC 107-/SFC 108 with CMP_ID:=cmp_id (call of the SFC 105 with SI_ID:=ev_id)
0	Additional system resources that could not be read with the previous call in MODE=1 or MODE=3 because you have specified a target field SYS_INST that is too small

Operating principle

If you have not selected a sufficiently large SYS_INST target area when you called the SFC 105 in MODE=1 or MODE=3, it contains the content of all currently occupied system resources selected via MODE parameter.

The SFY runtime is correspondingly high if many current system resources are occupied. On high CPU load this can result in exceeding of the maximum configurable cycle monitoring time.

You can work around this runtime problem as follows: Select a relatively small SYS_INST target area RET_VAL=W#16#0001 informs you if the SFC cannot enter all system resources to be read in SYS_INST. In this case, call SFC 105 in MODE=0 and with the same SI_ID as for the previous call until the value of RET_VAL is W#16#0000.

Note

Since the operating system does not coordinate the SFC 105 calls that belong to the read job, you should execute all SFC 105 calls with the same priority class.

Structure of the Target Area SYS_INST

The target area for the fetched occupied system resource must lie within a DB. You should appropriately define the target area as a field of structures, whereby a structure is constructed as follows:

Structure element	Data type	Description
SFC_NO	WORD	no. of the SFC that occupies the system resource
LEN	BYTE	Length of the structures in bytes, incl. SFC_NO and LEN: B#16#0C
SIG_STAT	BOOL	Signal state
ACK_STAT	BOOL	Acknowledgement status of the incoming event (positive edge)
EV_ID	DWORD	Message number
CMP_ID	DWORD	Partial system ID

Parameters	Declaration	Data type	Memory Area	Description
MODE	INPUT	INT	I, Q, M, D, L, Const.	Job identifier Permissible values • 1: Read all system resources • 2. Read the system resource that was occupied with EV_ID = ev_id when SFC 107-/SFC 108 was called • 3: Read the system resources that were occupied with CMP_ID = cmp_id when SFC 107-/SFC 108 was called • 0: subsequent call
SI_ID	INPUT	DWORD	I, Q, M, D, L, Const.	ID for the system resource(s) to be read Permissible values • 0, if MODE=1 • Message number ev_id, if MODE=2 • ID cmp_id for identification of the system section, if MODE=3
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Return value (error information or job

Parameters	Declaration	Data type	Memory Area	Description
				status)
N_SI	OUTPUT	INT	I, Q, M, D, L	number of output system resources with SYS_INT
SYS_INT	OUTPUT	ANY	D	Target area for the fetched system resources.

Error Information

Error code (W#16#...):	Explanation
0000	No error occurred.
0001	Not all system resources could be read because the SYS_INT target range you have selected is too short.
8081	(only with MODE=2 or 3) You have assigned the value 0 to SI_ID.
8082	(only with MODE=1) You have assigned one of 0 different values to SI_ID.
8083	(only with MODE=0) You have assigned SI_ID a value other than at the preceding call of the SFC with MODE=1 or 3.
8084	You have assigned an illegal value to MODE.
8085	SFC 105 is already being processed in another OB.
8086	Target area SYS_INST too small for a system resource.
8087	Target area SYS_INST does not exist in a DB.

21.16 Reading Dynamically Occupied System Resources with SFC 106 "READ_SI"

How dynamically occupied system resources develop when generating messages with the SFCs 107 and 108

When messages are generated with SFCs 107 "ALARM_DQ" and 108 "ALARM_D", the operating system occupies temporarily system memory space.

For example, if you do not delete a FB that exists in the CPU with SFC 107 or SFC 108 calls it may happen that corresponding system resources stay permanently occupied. If you reload the FB with SFC 108 or SFC 108 calls, it may happen that the SFCs 107 and 108 are not processed properly anymore.

Description

With SFC 106 "DEL_SI" you can delete currently used system resources.

SFC 106 "READ_SI" has three possible operating modes explained in the table below. Set the desired operating mode via the MODE parameter.

MODE	Which of the system resources occupied by SFC 107/SFC 108 are deleted?
1	All (call of SFC 106 with SI_ID:=0)
2	The system resource occupied by the call of SFC 107-/SFC 108 with EV_ID:=ev_id (call of the SFC 106 with SI_ID:=ev_id)
3	The system resource occupied by the call of SFC 107-/SFC 108 with CMP_ID:=cmp_id (call of the SFC 106 with SI_ID:=ev_id)

Parameters	Declaration	Data type	Memory Area	Description
MODE	INPUT	INT	I, Q, M, D, L, Const.	Job identifier Permissible values • 1: delete all system resources • 2: delete the system resource that was occupied with EV_ID = ev_id when SFC 107-/SFC 108 was called • 3: delete the system resources that were occupied with CMP_ID = cmp_id when SFC 107-/SFC 108 was called
SI_ID	INPUT	DWORD	I, Q, M, D, L, Const.	ID of the system resource(s) to be deleted Permissible values • 0, if MODE=1 • Message number ev_id, if MODE=2 • ID cmp_id for identification of the system section, if MODE=3
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Error Information

Error Information

Error code (W#16#...):	Explanation
0000	No error occurred.
8081	(only with MODE=2 or 3) You have assigned the value 0 to SI_ID.
8082	(only with MODE=1) You have assigned one of 0 different values to SI_ID.
8084	You have assigned an illegal value to MODE.
8085	SFC 106 is currently being processed.
8086	Not all selected system resources could be deleted because at least one of them was being processed when SFC 106 was called.

22 IEC Timers and IEC Counters

22.1 Generating a Pulse with SFB 3 "TP"

Description

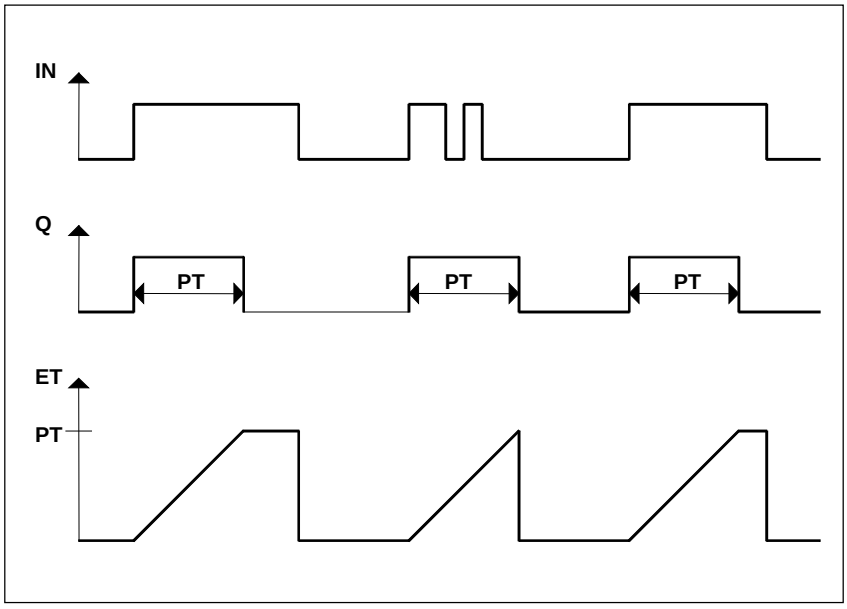
SFB 3 "TP" generates a pulse with the length PT. The timer runs only in the STARTUP and RUN modes.

A rising signal edge at input IN starts the pulse. Output Q remains set for the time PT regardless of changes in the input signal (in other words even when the IN input changes back from 0 to 1 before the time PT has expired).The ET output provides the time for which output Q has already been set. The maximum value of the ET output is the value of the PT input. Output ET is reset when input IN changes to 0, however, not before the time PT has expired.

SFB 3 "TP" complies with the IEC 1131-3 standard.

The operating system resets the instances of SFB 3 "TP" during a cold restart. If you want instances of this SFB to be initialized following a warm restart, you must call SFB 3 with PT = 0 ms in OB100. If instances of this SFB are located within another block, you can reset these instances, for example, by initializing the higher-level block.

Timing Diagram



Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	BOOL	I, Q, M, D, L, constant	Start input
PT	INPUT	TIME	I, Q, M, D, L, constant	Duration of the pulse. PT must be positive. (Note: the range of values is fixed by the TIME data type)
Q	OUTPUT	BOOL	I, Q, M, D, L	Status of the time
ET	OUTPUT	TIME	I, Q, M, D, L	Expired time

22.2 Generating an On Delay with SFB 4 "TON"

Description

SFB 4 "TON" delays a rising signal edge by the time PT. The timer runs only in the STARTUP and RUN modes.

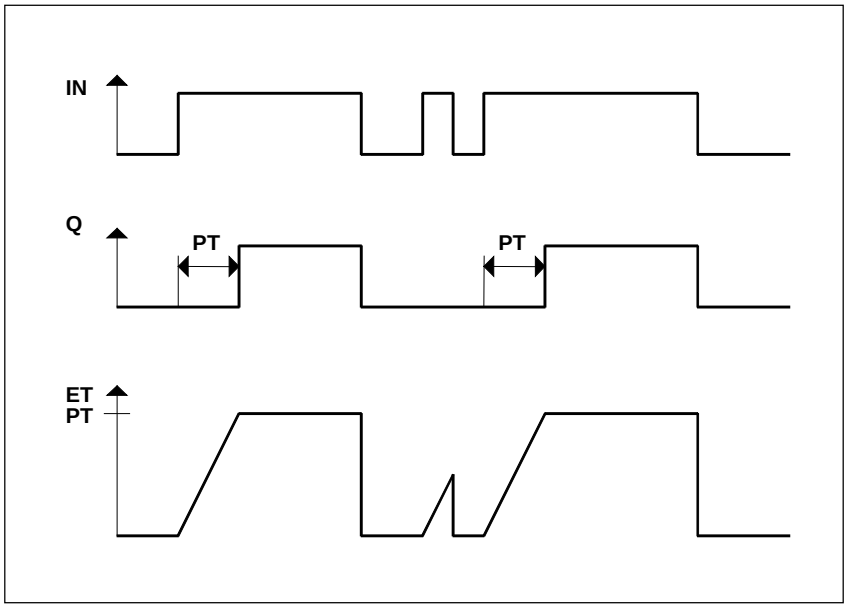
A rising edge at the IN input causes a rising edge at output Q after the time PT has expired. Q then remains set until the IN input changes to 0 again. If the IN input changes to 0 before the time PT has expired, output Q remains set to 0.

The ET output provides the time that has passed since the last rising edge at the IN input. Its maximum value is the value of the PT input. ET is reset when the IN input changes to 0.

SFB 4 "TON" complies with the IEC 1131-3 standard.

The operating system resets the instances of SFB 4 "TON" during a cold restart. If you want instances of this SFB to be initialized following a warm restart, you must call SFB 4 with PT = 0 ms in OB100. If instances of this SFB are located within another block, you can reset these instances, for example, by initializing the higher-level block.

Timing Diagram



Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	BOOL	I, Q, M, D, L, constant	Start input
PT	INPUT	TIME	I, Q, M, D, L, constant	Time by which the rising edge at the IN input is delayed. PT must be positive. (Note: the range of values is fixed by the TIME data type)
Q	OUTPUT	BOOL	I, Q, M, D, L	Status of the time
ET	OUTPUT	TIME	I, Q, M, D, L	Expired time

22.3 Generating an Off Delay with SFB 5 "TOF"

Description

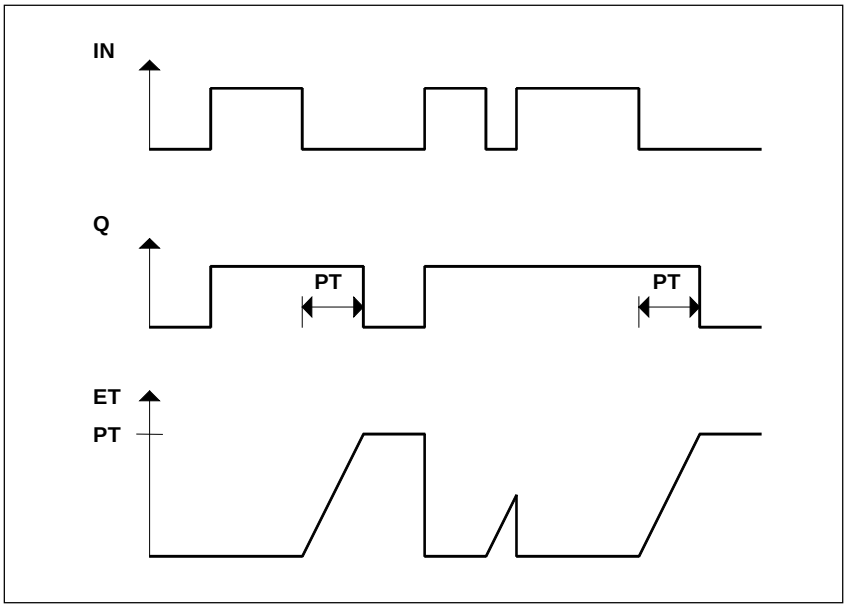
SFB 5 "TOF" delays a falling edge by the time PT. The timer runs only in the STARTUP and RUN modes.

A rising edge at the IN input causes a rising edge at output Q. A falling edge at the IN input causes a falling edge at output Q delayed by the time PT. If the IN input changes back to 1 before the time PT has expired, output Q remains set to 1. The ET output provides the time that has elapsed since the last falling edge at the IN input. Its maximum value is, however the value of the PT input. ET is reset when the IN input changes to 1.

SFB 5 "TOF" complies with the IEC 1131-3 standard.

The operating system resets the instances of SFB 5 "TOF" during a cold restart. If you want instances of this SFB to be initialized following a warm restart, you must call SFB 5 with PT = 0 ms in OB100. If instances of this SFB are located within another block, you can reset these instances, for example, by initializing the higher-level block.

Timing Diagram



Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	BOOL	I, Q, M, D, L, constant	Start input
PT	INPUT	TIME	I, Q, M, D, L, constant	Time by which the falling edge at the IN input is delayed. PT must be positive. (Note: the range of values is fixed by the TIME data type)
Q	OUTPUT	BOOL	I, Q, M, D, L	Status of the time
ET	OUTPUT	TIME	I, Q, M, D, L	Expired time

22.4 Counting Up with SFB 0 "CTU"

Description

You can count up with SFB 0 "CTU." The counter is incremented by 1 by a rising edge at the CU input (compared with the last SFB call). If the counted value reaches the upper limit of 32767, it is no longer incremented. Each subsequent rising edge at the CU input no longer has an effect.

Signal level 1 at the R input resets the counter to the value 0 regardless of the value currently at the CU input.

The Q output indicates whether the current counted value is greater or equal to the preset value PV.

SFB 0 "CTU" complies with the IEC 1131-3 standard.

The operating system resets the instances of SFB 0 "CTU" during a cold restart. If you want instances of this SFB to be initialized following a warm restart, you must call SFB 0 with R = 1 in OB100. If instances of this SFB are located within another block, you can reset these instances, for example, by initializing the higher-level block.

Parameter	Declaration	Data Type	Memory Area	Description
CU	INPUT	BOOL	I, Q, M, D, L, constant	Counter input
R	INPUT	BOOL	I, Q, M, D, L, constant	Reset input R is dominant over CU.
PV	INPUT	INT	I, Q, M, D, L, constant	Preset value. Refer to parameter Q for the effect of PV.
Q	OUTPUT	BOOL	I, Q, M, D, L	Status of the counter: Q has the following value • 1, if CV $\geq$ PV • 0 otherwise
CV	OUTPUT	INT	I, Q, M, D, L	Current count value (possible value: 0 to 32 767)

22.5 Counting Down with SFB 1 "CTD"

Description

You can count down with SFB 1 "CTD." The counter decrements at a rising edge on the CD input (compared with the last SFB call). If the count value reaches the lower limit of -32768, it no longer decrements. Any further rising edge at the CD input then has no further effect.

Signal level 1 at the LOAD input sets the counter to the preset value PV regardless of the value at the CD input.

The Q output indicates whether the current counted value is less than or equal to 0.

SFB 1 "CTD" complies with the IEC 1131-3 standard.

The operating system resets instances of SFB 1 "CTD" during a cold restart. If you want instances of this SFB to be initialized following a warm restart, you must call SFB 1 with LOAD = 1 and PV = required initial value for CV in OB100. If instances of this SFB are located within another block, you can reset these instances, for example, by initializing the higher-level block.

Parameter	Declaration	Data Type	Memory Area	Description
CD	INPUT	BOOL	I, Q, M, D, L, constant	Count input
LOAD	INPUT	BOOL	I, Q, M, D, L, constant	Load input. LOAD is dominant over CD.
PV	INPUT	INT	I, Q, M, D, L, constant	Preset value. The counter is preset to PV when the signal level at the LOAD input is 1.
Q	OUTPUT	BOOL	I, Q, M, D, L	Status of the counter: Q has the following value: • 1, if CV $\leq$ 0 • 0 otherwise
CV	OUTPUT	INT	I, Q, M, D, L	Current count value(possible values: -32 768 to 32 767)

22.6 Counting Up and Counting Down with SFB 2 "CTUD"

Description

You can count up and down with SFB 2 "CTUD." The count value is changed by a rising edge, compared with the last SFB call as follows:

- At input CU it is incremented by 1
- At input CD it decrements by 1.

If the count value reaches the limits, the counter reacts as follows:

- The lower limit of -32768, it is no longer decrements
- The upper limit of 32767, it is no longer incremented.

If there is a rising edge at both input CU and input CD in one cycle, the counter retains its current value. This reaction does not comply with the standard IEC 1131-3. In the standard, the CU input is dominant if both signals are active at the same time. This change has been proposed to the IEC.

A signal level 1 at the LOAD input presets the counter to the value PV regardless of the values at the CU and CD inputs.

The signal level 1 at the R input resets the counter to the value 0 regardless of the values at the CU, CD and LOAD inputs. The QU output indicates whether the current count value is greater than or equal to the preset value PV; the QD output indicates whether the value is less than or equal to 0.

The operating system resets SFB 2 "CTUD" during a cold restart. If you want SFB 2 "CTUD" to be initialized following a warm restart, you must call SFB 2 in OB100 as follows:

- With R = 1 when using the block to count up
- With R = 0 and LOAD = 1 and PV = required initial value for CV when using the block to count down
- If instances of this SFB are located within another block, you can reset these instances, for example, by initializing the higher-level block.

Parameter	Declaration	Data Type	Memory Area	Description
CU	INPUT	BOOL	I, Q, M, D, L, constant	Count up input.
CD	INPUT	BOOL	I, Q, M, D, L, constant	Count down input
R	INPUT	BOOL	I, Q, M, D, L, constant	Reset input. R is dominant over LOAD.
LOAD	INPUT	BOOL	I, Q, M, D, L, constant	Load input. LOAD is dominant over CU and CD.
PV	INPUT	INT	I, Q, M, D, L, constant	Preset value. The counter is set to the preset value PV when the signal level at the LOAD input is 1.
QU	OUTPUT	BOOL	I, Q, M, D, L	Status of the up counter: QU has the following value • 1, if CV $\geq$PV • 0 otherwise
QD	OUTPUT	BOOL	I, Q, M, D, L	Status of the down counter: QD has the following value • 1, if CV $\leq$ 0 • 0 otherwise
CV	OUTPUT	INT	I, Q, M, D, L	Current count value (possible values: -32 768 to 32 767)

23 IEC Functions

23.1 Overview

You can copy the following International Electrotechnical Commission (IEC) functions from the STEP 7 library "S7libs\Stdlib30" to your program directory.

Name	IEC Block Family	Function
FC3 D_TOD_DT	Convert	Combine DATE and TIME_OF_DAY to DT
FC6 DT_DATE	Convert	Extract the DATE from DT
FC7 DT_DAY	Convert	Extract the day of the week from DT
FC8 DT_TOD	Convert	Extract the TIME_OF_DAY from DT
FC33 S5TI_TIM	Convert	Data type conversion S5TIME to TIME
FC40 TIM_S5TI	Convert	Data type conversion TIME to S5TIME
FC16 I_STRNG	Convert	Data type conversion INT to STRING
FC5 DI_STRNG	Convert	Data type conversion DINT to STRING
FC30 R_STRNG	Convert	Data type conversion REAL to STRING
FC38 STRNG_I	Convert	Data type conversion STRING to INT
FC37 STRNG_DI	Convert	Data type conversion STRING to DINT
FC39 STRNG_R	Convert	Data type conversion STRING to REAL
FC9 EQ_DT	DT	Compare DT for equal
FC12 GE_DT	DT	Compare DT for greater than or equal
FC14 GT_DT	DT	Compare DT for greater than
FC18 LE_DT	DT	Compare DT for smaller than or equal
FC23 LT_DT	DT	Compare DT for smaller than
FC28 NE_DT	DT	Compare DT for unequal
FC10 EQ_STRNG	String	Compare STRING for equal
FC13 GE_STRNG	String	Compare STRING for greater than or equal
FC15 GT_STRNG	String	Compare STRING for greater than
FC19 LE_STRNG	String	Compare STRING for smaller than or equal
FC24 LT_STRNG	String	Compare STRING for smaller than
FC29 NE_STRNG	String	Compare STRING for unequal
FC21 LEN	String	Length of a STRING variable
FC20 LEFT	String	Left part of a STRING variable
FC32 RIGHT	String	Right part of a STRING variable
FC26 MID	String	Middle part of a STRING variable
FC2 CONCAT	String	Combine two STRING variables
FC17 INSERT	String	Insert in a STRING variable

Name	IEC Block Family	Function
FC4 DELETE	String	Delete in a STRING variable
FC31 REPLACE	String	Replace in a STRING variable
FC11 FIND	String	Find in a STRING variable
FC1 AD_DT_TM	Floating Point Math	Add duration to a time
FC35 SB_DT_TM	Floating Point Math	Subtract duration from a time
FC34 SB_DT_DT	Floating Point Math	Subtract two time values
FC22 LIMIT	Floating Point Math	Limit
FC25 MAX	Floating Point Math	Select maximum
FC27 MIN	Floating Point Math	Select minimum
FC36 SEL	Floating Point Math	Binary selection

For more information on the IEC conform communication blocks, refer to the communication functions in the context help for the system function blocks (SFBs/SFCs) (see: *Differences between the Blocks of the S7 Communication and the S7 Basic Communication*)

23.2 Technical Data of the IEC Functions

Memory Requirements

The following table shows how much work memory and how much load memory is required for each of the International Electrotechnical Commission (IEC) functions, and the number of bytes of local data required for each IEC function.

FC No.	Name	Size (No. of Bytes) in Work Memory	Load Memory	Local Data (Bytes)
FC3	D_TOD_DT	634	810	12
FC6	DT_DATE	340	466	10
FC7	DT_DAY	346	472	10
FC8	DT_TOD	114	210	6
FC33	S5TI_TIM	94	208	2
FC40	TIM_S5TI	104	208	6
FC16	I_STRNG	226	340	10
FC5	DI_STRNG	314	440	18
FC30	R_STRNG	528	684	28
FC38	STRNG_I	292	420	12
FC37	STRNG_DI	310	442	12
FC39	STRNG_R	828	1038	30
FC9	EQ_DT	96	194	2
FC12	GE_DT	174	288	4
FC14	GT_DT	192	310	4
FC18	LE_DT	168	280	4
FC23	LT_DT	192	310	4
FC28	NE_DT	96	194	2
FC10	EQ_STRNG	114	220	4
FC13	GE_STRNG	162	282	8
FC15	GT_STRNG	158	278	8
FC19	LE_STRNG	162	282	8
FC24	LT_STRNG	158	278	8
FC29	NE_STRNG	150	266	8
FC21	LEN	38	132	2
FC20	LEFT	200	320	8
FC32	RIGHT	230	350	8
FC26	MID	302	390	8
FC2	CONCAT	358	452	14
FC17	INSERT	488	644	20
FC4	DELETE	376	512	8
FC31	REPLACE	562	726	20
FC11	FIND	236	360	14

FC No.	Name	Size (No. of Bytes) in Work Memory	Load Memory	Local Data (Bytes)
FC1	AD_DT_TM	1350	1590	22
FC35	SB_DT_TM	1356	1596	22
FC34	SB_DT_DT	992	1178	30
FC22	LIMIT	426	600	12
FC25	MAX	374	532	8
FC27	MIN	374	532	8
FC36	SEL	374	560	8

23.3 Date and Time as Complex Data Types

Actual Parameters for DATE_AND_TIME

The DATE_AND_TIME data type falls into the category of complex data types, along with ARRAY, STRING, and STRUCT. The permissible memory areas for complex data types are the data block (D) and local data (L) areas.

Because DATE_AND_TIME is a complex data type, when you use DATE_AND_TIME as a formal parameter in a statement, you can provide the actual parameter only in one of the following forms:

- A block-specific symbol from the variable declaration table for a specific block
- A symbolic name for a data block, such as "DB_sys_info.System_Time", made up of the following parts:
  - A name defined in the symbol table for the number of the data block (for example, "DB_sys_info" for DB5)
  - A name defined within the data block for the DATE_AND_TIME element (for example, "System_Time" for a variable of data type DATE_AND_TIME contained in DB5)

You cannot pass constants as actual parameters to formal parameters of the complex data types, including DATE_AND_TIME. Also, you cannot pass absolute addresses as actual parameters to DATE_AND_TIME.

23.4 Time-of-Day Functions

Description FC1 AD_DT_TM

The function FC1 adds a duration (format TIME) to a time (format DT) and provides a new time (format DT) as the result. The time (parameter T) must be in the range from DT#1990-01-01-00:00:00.000 to DT#2089-12-31-23:59:59.999. The function does not run an input check. If the result of the addition is not within the valid range, the result is limited to the corresponding value and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
T	INPUT	DATE_AND_TIME	D, L	Time in format DT
D	INPUT	TIME	I, Q, M, D, L, Const.	Duration in format TIME
RET_VAL	OUTPUT	DATE_AND_TIME	D, L	Sum in format DT

You can assign only a symbolically defined variable for the input parameter T and the output parameter.

Description FC3 D_TOD_DT

The function FC3 combines the data formats DATE and TIME_OF_DAY (TOD) together and converts these formats to the data type format DATE_AND_TIME (DT). The input value IN1 must be between the limits DATE#1990-01-01 and DATE#2089-12-31. (This value is not checked.) The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
IN1	INPUT	DATE	I, Q, M, D, L, Const.	Input variable in format DATE
IN2	INPUT	TIME_OF_DAY	I, Q, M, D, L, Const.	Input variable in format TOD
RET_VAL	OUTPUT	DATE_AND_TIME	D, L	Return value in format DT

You can assign only a symbolically defined variable for the return value.

Description FC6 DT_DATE

The function FC6 extracts the data type format DATE from the format DATE_AND_TIME. The DATE value must be between the limits DATE#1990-1-1 and DATE#2089-12-31. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
RET_VAL	OUTPUT	DATE	I, Q, M, D, L	Return value in format DATE

You can assign only a symbolically defined variable for this input.

Description FC7 DT_DAY

The function FC7 extracts the day of the week from the format DATE_AND_TIME. The day of the week is available in the data type format INTEGER:

- 1 = Sunday
- 2 = Monday
- 3 = Tuesday
- 4 = Wednesday
- 5 = Thursday
- 6 = Friday
- 7 = Saturday

The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Return value in format INT

You can assign only a symbolically defined variable for this input.

Description FC8 DT_TOD

The function FC8 extracts the data type format TIME_OF_DAY from the format DATE_AND_TIME. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
RET_VAL	OUTPUT	TIME_OF_DAY	I, Q, M, D, L	Return value in format TOD

You can assign only a symbolically defined variable for this input.

Description FC33 S5TI_TIM

The function FC33 converts the data type format S5TIME to the format TIME. If the result of the conversion is outside the TIME range, the result is limited to the corresponding value and the binary result (BR) bit of the status word is set to "0".

Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	S5TIME	I, Q, M, D, L, Const.	Input variable in format S5TIME
RET_VAL	OUTPUT	TIME	I, Q, M, D, L	Return value in format TIME

Description FC34 SB_DT_DT

The function FC34 subtracts two time values (format DT) and provides a duration (format TIME) as the result. The times must be in the range from DT#1990-01-01-00:00:00.000 to DT#2089-12-31-23:59:59.999. The function does not run an input check. If the first time (parameter T1) is greater (more recent) than the second (parameter DT2), the result is positive; if the first time is smaller (less recent) than the second, the result is negative. If the result of the subtraction is outside the TIME range, the result is limited to the corresponding value and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
DT1	INPUT	DATE_AND_TIME	D, L	First time in format DT
DT2	INPUT	DATE_AND_TIME	D, L	Second time in format DT
RET_VAL	OUTPUT	TIME	I, Q, M, D, L	Difference in format TIME

You can assign only a symbolically defined variable for the input parameters.

Description FC35 SB_DT_TM

The function FC35 subtracts a duration (format TIME) from a time (format DT) and provides a new time (format DT) as the result. The time (parameter T) must be between DT#1990-01-01-00:00:00.000 and DT#2089-12-31-23:59:59.999. The function does not run an input check. If the result of the subtraction is not within the valid range, the result is limited to the corresponding value and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
T	INPUT	DATE_AND_TIME	D, L	Time in format DT
D	INPUT	TIME	I, Q, M, D, L, Const.	Duration in format TIME
RET_VAL	OUTPUT	DATE_AND_TIME	D, L	Difference in format DT

You can assign only a symbolically defined variable for the input parameter T and the output parameter.

23.5 Comparing DATE_AND_TIME Variables

Description FC9 EQ_DT

The function FC9 compares the contents of two variables in the data type format DATE_AND_TIME to find out if they are equal and outputs the result of the comparison as a return value. The return value has the signal state “1" if the time at parameter DT1 is the same as the time at parameter DT2. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
DT1	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
DT2	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC12 GE_DT

The function FC12 compares the contents of two variables in the data type format DATE_AND_TIME to find out if one is greater or equal to the other and outputs the result of the comparison as a return value. The return value has the signal state “1" if the time at parameter DT1 is greater (more recent) than the time at parameter DT2 or if both times are the same. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
DT1	INPUT	DATE_AND_TIME	D, L	Input variable in format DT

DT2	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC14 GT_DT

The function FC14 compares the contents of two variables in the data type format DATE_AND_TIME to find out if one is greater than the other and outputs the result of the comparison as a return value. The return value has the signal state “1" if the time at parameter DT1 is greater (more recent) than the time at parameter DT2. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
DT1	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
DT2	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC18 LE_DT

The function FC18 compares the contents of two variables in the data type format DATE_AND_TIME to find out if one is smaller than or equal to the other and outputs the result of the comparison as a return value. The return value has the signal state “1" if the time at parameter DT1 is smaller (less recent) than the time at parameter DT2 or if both times are the same. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
DT1	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
DT2	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC23 LT_DT

The function FC23 compares the contents of two variables in the data type format DATE_AND_TIME to find out if one is smaller than the other and outputs the result of the comparison as a return value. The return value has the signal state “1" if the time at parameter DT1 is smaller (less recent) than the time at parameter DT2. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
DT1	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
DT2	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC28 NE_DT

The function FC28 compares the contents of two variables in the data type format DATE_AND_TIME to find out if they are unequal and outputs the result of the comparison as a return value. The return value has the signal state “1" if the time at parameter DT1 is not equal to the time at parameter DT2. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
DT1	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
DT2	INPUT	DATE_AND_TIME	D, L	Input variable in format DT
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

23.6 Comparing STRING Variables

Description FC10 EQ_STRNG

The function FC10 compares the contents of two variables in the data type format STRING to find out if they are equal and outputs the result of the comparison as a return value. The return value has the signal state “1" if the string at parameter S1 is the same as the string at parameter S2.

The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
S1	INPUT	STRING	D, L	Input variable in format STRING
S2	INPUT	STRING	D, L	Input variable in format STRING
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC13 GE_STRNG

The function FC13 compares the contents of two variables in the data type format STRING to find out if the first is greater than or equal to the other and outputs the result of the comparison as a return value. The return value has the signal state “1" if the string at parameter S1 is greater than or equal to the string at parameter S2.

The characters are compared by their ASCII code (for example, 'a' is greater than 'A'), starting from the left. The first character to be different decides the result of the comparison. If the first characters are the same, the longer string is greater.

The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
S1	INPUT	STRING	D, L	Input variable in format STRING
S2	INPUT	STRING	D, L	Input variable in format STRING
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC15 GT_STRNG

The function FC15 compares the contents of two variables in the data type format STRING to find out if the first is greater than the other and outputs the result of the comparison as a return value. The return value has the signal state “1" if the string at parameter S1 is greater than the string at parameter S2.

The characters are compared by their ASCII code (for example, 'a' is greater than 'A'), starting from the left. The first character to be different decides the result of the comparison. If the first characters are the same, the longer string is greater.

The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
S1	INPUT	STRING	D, L	Input variable in format STRING
S2	INPUT	STRING	D, L	Input variable in format STRING
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC19 LE_STRNG

The function FC19 compares the contents of two variables in the data type format STRING to find out if the first is smaller than or equal to the other and outputs the result of the comparison as a return value. The return value has the signal state “1" if the string at parameter S1 is smaller than or equal to the string at parameter S2.

The characters are compared by their ASCII code (for example, 'a' is smaller than 'A'), starting from the left. The first character to be different decides the result of the comparison. If the first characters are the same, the shorter string is smaller.

The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
S1	INPUT	STRING	D, L	Input variable in format STRING
S2	INPUT	STRING	D, L	Input variable in format STRING
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC24 LT_STRNG

The function FC24 compares the contents of two variables in the data type format STRING to find out if the first is smaller than the other and outputs the result of the comparison as a return value. The return value has the signal state “1" if the string at parameter S1 is smaller than the string at parameter S2.

The characters are compared by their ASCII code (for example, 'a' is smaller than 'A'), starting from the left. The first character to be different decides the result of the comparison. If the left part of the longer character string and the shorter character string are the same, the shorter string is smaller.

The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
S1	INPUT	STRING	D, L	Input variable in format STRING
S2	INPUT	STRING	D, L	Input variable in format STRING
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

Description FC29 NE_STRNG

The function FC29 compares the contents of two variables in the data type format STRING to find out if they are unequal and outputs the result of the comparison as a return value. The return value has the signal state “1" if the string at parameter S1 is not equal to the string at parameter S2.

The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
S1	INPUT	STRING	D, L	Input variable in format STRING
S2	INPUT	STRING	D, L	Input variable in format STRING
RET_VAL	OUTPUT	BOOL	I, Q, M, D, L	Result of comparison

You can assign only a symbolically defined variable for the input parameters.

23.7 Editing Number Values

Description FC22 LIMIT

The function FC22 limits the number value of a variable to limit values which can have parameters assigned. Variables of the data types INT, DINT, and REAL are permitted as input values. All variables with parameters assigned must be of the same data type. The variable type recognized by the ANY pointer. The lower limit value (parameter MN) must be smaller than/equal to the upper limit value (parameter MX).

The output value remains unchanged and the binary result (BR) bit of the status word is set to “0" if any of the following are true:

- A variable with parameters assigned has an invalid data type
- All variables with parameters assigned do not have the same data type
- The lower limit value is greater than the upper limit value
- A REAL variable does not represent a valid floating-point number.

Parameter	Declaration	Data Type	Memory Area	Description
MN	INPUT	ANY	I, Q, M, D, L	Lower limit
IN	INPUT	ANY	I, Q, M, D, L	Input variable
MX	INPUT	ANY	I, Q, M, D, L	Upper limit
RET_VAL	OUTPUT	ANY	I, Q, M, D, L	Limited output variable

Description FC25 MAX

The function FC25 selects the largest of three numerical variable values. Variables of the data types INT, DINT, and REAL are permitted as input values. All variables with parameters assigned must be of the same data type. The variable type recognized by the ANY pointer.

The output value remains unchanged and the binary result (BR) bit of the status word is set to “0" if any of the following are true:

- A variable with parameters assigned has an invalid data type
- All variables with parameters assigned do not have the same data type
- A REAL variable does not represent a valid floating-point number.

Parameter	Declaration	Data Type	Memory Area	Description
IN1	INPUT	ANY	I, Q, M, D, L	First input value
IN2	INPUT	ANY	I, Q, M, D, L	Second input value
IN3	INPUT	ANY	I, Q, M, D, L	Third input value
RET_VAL	OUTPUT	ANY	I, Q, M, D, L	Largest of the input values

23.8 Example in STL

```
CALL FC 25
   IN1               := P#M 10.0 DINT 1
   IN2               := MD20
   IN3               := P#DB1.DBX 0.0 DINT 1
   RET_VAL           := P#M 40.0 DINT 1
=                    M 0.0
```

Note:
The admitted data types INT, DINT and REAL must be entered in the ANY pointer. Such paramters as “MD20“ are also admitted, but you must define the corresponding data type of “MD20“ in “Symbol“.

Description FC27 MIN

The function FC27 selects the smallest of three numerical variable values. Variables of the data types INT, DINT, and REAL are permitted as input values. All variables with parameters assigned must be of the same data type. The variable type is recognized by the ANY pointer.

The output value remains unchanged and the binary result (BR) bit of the status word is set to “0" if any of the following are true:

- A variable with parameters assigned has an invalid data type
- All variables with parameters assigned do not have the same data type
- A REAL variable does not represent a valid floating-point number.

Parameter	Declaration	Data Type	Memory Area	Description
IN1	INPUT	ANY	I, Q, M, D, L	First input value
IN2	INPUT	ANY	I, Q, M, D, L	Second input value
IN3	INPUT	ANY	I, Q, M, D, L	Third input value
RET_VAL	OUTPUT	ANY	I, Q, M, D, L	Smallest of the input values

23.9 Example in STL

```
CALL FC 27
   IN1               := P#M 10.0 DINT 1
   IN2               := MD20
   IN3               := P#DB1.DBX 0.0 DINT 1
   RET_VAL           := P#M 40.0 DINT 1
=                    M 0.0
```

Note:
The admitted data types INT, DINT and REAL must be entered in the ANY pointer. Such paramters as “MD20“ are also admitted, but you must define the corresponding data type of “MD20“ in “Symbol“.

Description FC36 SEL

The function FC36 selects one of two variable values depending on a switch (parameter G). Variables with all data types which correspond to the data width bit, byte, word, and double word (not data types DT and STRING) are permitted as input values at the parameters IN0 and IN1. Both input variables and the output variable must be of the same data type.

The output value remains unchanged and the binary result (BR) bit of the status word is set to “0" if any of the following are true:

- A variable with parameters assigned has an invalid data type
- All variables with parameters assigned do not have the same data type
- A REAL variable does not represent a valid floating-point number.

Parameter	Declaration	Data Type	Memory Area	Description
G	INPUT	BOOL	I, Q, M, D, L	Selection switch
IN0	INPUT	ANY	I, Q, M, D, L	First input value
IN1	INPUT	ANY	I, Q, M, D, L	Second input value
RET_VAL	OUTPUT	ANY	I, Q, M, D, L	Selected input value

23.10 Editing STRING Variables

Description FC2 CONCAT

The function FC2 concatenates two STRING variables together to form one string. If the resulting string is longer than the variable given at the output parameter, the result string is limited to the maximum set length and the binary result (BR) bit of the status word set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
IN1	INPUT	STRING	D, L	Input variable in format STRING
IN2	INPUT	STRING	D, L	Input variable in format STRING
RET_VAL	OUTPUT	STRING	D, L	Combined string

You can assign only a symbolically defined variable for the parameters.

Description FC4 DELETE

The function FC4 deletes a number of characters (L) from the character at position P (inclusive) in a string. If L and/or P are equal to zero or if P is greater than the current length of the input string, the input string is returned. If the sum of L and P is greater than the input string, the string is deleted up to the end. If L and/or P are negative, a blank string is returned and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	STRING	D, L	STRING variable to be deleted in
L	INPUT	INT	I, Q, M, D, L, Const.	Number of characters to be deleted
P	INPUT	INT	I, Q, M, D, L, Const.	Position of first character to be deleted
RET_VAL	OUTPUT	STRING	D, L	Result string

You can assign only a symbolically defined variable for the input parameter IN and the output parameter.

Description FC11 FIND

The function FC11 provides the position of the second string (IN2) within the first string (IN1). The search starts on the left; the first occurrence of the string is reported. If the second string is not found in the first, zero is returned. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
IN1	INPUT	STRING	D, L	STRING variable to be searched in
IN2	INPUT	STRING	D, L	STRING variable to be found
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Position of the string found

You can assign only a symbolically defined variable for the input parameters IN1 and IN2.

Description FC17 INSERT

The function FC17 inserts a string at parameter IN2 into the string at parameter IN1 after the character at position P. If P equals zero, the second string is inserted before the first string. If P is greater than the current length of the first string, the second string is appended to the first, If P is negative, a blank string is output and the binary result (BR) bit of the status word is set to “0". The binary result bit is also set to “0" if the resulting string is longer than the variable given at the output parameter; in this case the result string is limited to the maximum set length.

Parameter	Declaration	Data Type	Memory Area	Description
IN1	INPUT	STRING	D, L	STRING variable to be inserted into
IN2	INPUT	STRING	D, L	STRING variable to be inserted
P	INPUT	INT	I, Q, M, D, L, Const.	Insert position
RET_VAL	OUTPUT	STRING	D, L	Result string

You can assign only a symbolically defined variable for the input parameters IN1 and IN2 and the output parameter.

Description FC20 LEFT

The function FC20 provides the first L characters of a string (where L stands for a number). If L is greater than the current length of the STRING variables, the input value is returned. With L = 0 and with a blank string as the input value, a blank string is returned. If L is negative, a blank string is returned and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	STRING	D, L	Input variable in format STRING
L	INPUT	INT	I, Q, M, D, L, Const.	Length of the left character sting
RET_VAL	OUTPUT	STRING	D, L	Output variable in format STRING

You can assign only a symbolically defined variable for the parameter IN and the return value.

Description FC21 LEN

A STRING variable contains two lengths: the maximum length (this is given in square brackets when the variables are being defined) and the current length (this is the number of currently valid characters). The current length must be smaller than or equal to the maximum length. The number of bytes occupied by a string is 2 greater than the maximum length.

The function FC21 outputs the current length of a string (number of valid characters) as a return value. A blank string (' ') has the length zero. The maximum length is 254. The function does not report any errors.

Parameter	Declaration	Data Type	Memory Area	Description
S	INPUT	STRING	D, L	Input variable in format STRING
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Number of valid characters

You can assign only a symbolically defined variable for the input parameter.

Description FC26 MID

The function FC26 provides the middle part of a string (L characters from the character P inclusive). If the sum of L and P exceeds the current length of the STRING variables, a string is returned from the character P to the end of the input value. In all other cases (P is outside the current length, P and/or L are equal to zero or negative), a blank string is returned and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	STRING	D, L	Input variable in format STRING
L	INPUT	INT	I, Q, M, D, L, Const.	Length of the middle character sting
P	INPUT	INT	I, Q, M, D, L, Const.	Position of first character
RET_VAL	OUTPUT	STRING	D, L	Output variable in format STRING

You can assign only a symbolically defined variable for the parameter IN and the return value.

Description FC31 REPLACE

The function FC31 replaces a number of characters (L) of the first string (IN1) from the character at position P (inclusive) with the second string (IN2). If L is equal to zero, the first string is returned. If P is equal to zero or one, the string is replaced from the first character (inclusive). If P is outside the first string, the second string is appended to the first string. If L and/or P is negative, a blank string is returned and the binary result (BR) bit of the status word is set to “0". The binary result bit is also set to “0" if the resulting string is longer than the variable given at the output parameter; in this case the result string is limited to the maximum set length.

Parameter	Declaration	Data Type	Memory Area	Description
IN1	INPUT	STRING	D, L	STRING variable to be replaced in
IN2	INPUT	STRING	D, L	STRING variable to be inserted
L	INPUT	INT	I, Q, M, D, L, Const.	Number of characters to be replaced
P	INPUT	INT	I, Q, M, D, L, Const.	Position of first character to be replaced
RET_VAL	OUTPUT	STRING	D, L	Result string

You can assign only a symbolically defined variable for the input parameters IN1 and IN2 and the output parameter.

Description FC32 RIGHT

The function FC32 provides the last L characters of a string (where L stands for a number). If L is greater than the current length of the STRING variables, the input value is returned. With L = 0 and with a blank string as the input value, a blank string is returned. If L is negative, a blank string is returned and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	STRING	D, L,	Input variable in format STRING
L	INPUT	INT	I, Q, M, D, L, Const.	Length of the right character sting
RET_VAL	OUTPUT	STRING	D, L	Output variable in format STRING

You can assign only a symbolically defined variable for the parameter IN and the return value.

23.11 Converting Data Type Formats

Description FC5 DI_STRNG

The function FC5 converts a variable in DINT data type format to a string. The string is shown preceded by a sign. If the variable given at the return parameter is too short, no conversion takes place and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
I	INPUT	DINT	I, Q, M, D, L, Const.	Input value
RET_VAL	OUTPUT	STRING	D, L	Result string

You can assign only a symbolically defined variable for the output parameter.

Description FC16 I_STRNG

The function FC16 converts a variable in INT data type format to a string. The string is shown preceded by a sign. If the variable given at the return parameter is too short, no conversion takes place and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
I	INPUT	INT	I, Q, M, D, L, Const.	Input value
RET_VAL	OUTPUT	STRING	D, L	Result string

You can assign only a symbolically defined variable for the output parameter.

Description FC30 R_STRNG

The function FC30 converts a variable in REAL data type format to a string. The string is shown with 14 digits:

±v.nnnnnnnE±xx	±	Sign
	v	1 digit before the decimal point
	n	7 digits after the decimal point
	x	2 exponential digits

If the variable given at the return parameter is too short or if no valid floating-point number is given at parameter IN, no conversion takes place and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
IN	INPUT	REAL	I, Q, M, D, L, Const.	Input value
RET_VAL	OUTPUT	STRING	D, L	Result string

You can assign only a symbolically defined variable for the output parameter.

Description FC37 STRNG_DI

The function FC37 converts a string to a variable in DINT data type format. The first character in the string may be a sign or a number, the characters which then follow must be numbers. If the length of the string is equal to zero or greater than 11, or if invalid characters are found in the string, no conversion takes place and the binary result (BR) bit of the status word is set to “0". If the result of the conversion is outside the DINT range, the result is limited to the corresponding value and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
S	INPUT	STRING	D, L	Input string
RET_VAL	OUTPUT	DINT	I, Q, M, D, L	Result

You can assign only a symbolically defined variable for the input parameter.

Description FC38 STRNG_I

The function FC38 converts a string to a variable in INT data type format. The first character in the string may be a sign or a number, the characters which then follow must be numbers. If the length of the string is equal to zero or greater than 6, or if invalid characters are found in the string, no conversion takes place and the binary result (BR) bit of the status word is set to “0". If the result of the conversion is outside the INT range, the result is limited to the corresponding value and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
S	INPUT	STRING	D, L	Input string
RET_VAL	OUTPUT	INT	I, Q, M, D, L	Result

You can assign only a symbolically defined variable for the input parameter.

Description FC39 STRNG_R

The function FC39 converts a string to a variable in REAL data type format. The string must have the following format:

±v.nnnnnnnE±xx	±	Sign
	v	1 digit before the decimal point
	n	7 digits after the decimal point
	x	2 exponential digits

If the length of the string is smaller than 14, or if it is not structured as shown above, no conversion takes place and the binary result (BR) bit of the status word is set to “0". If the result of the conversion is outside the REAL range, the result is limited to the corresponding value and the binary result (BR) bit of the status word is set to “0".

Parameter	Declaration	Data Type	Memory Area	Description
S	INPUT	STRING	D, L	Input string
RET_VAL	OUTPUT	REAL	I, Q, M, D, L	Result

You can assign only a symbolically defined variable for the input parameter.

24 SFBs for Integrated Control

24.1 Continuous Control with SFB 41/FB 41 "CONT_C"

Introduction

SFB/FB "CONT_C" is used on SIMATIC S7 programmable logic controllers to control technical processes with continuous input and output variables. During parameter assignment, you can activate or deactivate sub-functions of the PID controller to adapt the controller to the process. You can assign this easily by using the parameter assignment tool (Menu path: **Start > Simatic > Step7 > Assign PID Control parameters**). The online electronic manual is found under **Start > Simatic > Step7 > Assign PID Control English**.

Application

You can use the controller as a PID fixed setpoint controller or in multi-loop controls as a cascade, blending or ratio controller. The functions of the controller are based on the PID control algorithm of the sampling controller with an analog signal, if necessary extended by including a pulse generator stage to generate pulse duration modulated output signals for two or three step controllers with proportional actuators.

Description

Apart from the functions in the setpoint and process value branches, the SFB/FB implements a complete PID controller with continuous manipulated variable output and the option of influencing the manipulated value manually. In the following, you will find a detailed description of the sub-functions:

Setpoint Branch

The setpoint is entered in floating-point format at the **SP_INT** input.

Process Variable Branch

The process variable can be input in the peripheral (I/O) or floating-point format. The CRP_IN function converts the PV_PER peripheral value to a floating-point format of -100 to +100 % according to the following formula:

$$\text{Output of CPR\_IN} = \text{PV\_PER} * \frac{10}{2764}$$

The PV_NORM function normalizes the output of CRP_IN according to the following formula:

Output of PV_NORM = (output of CPR_IN) * PV_FAC + PV_OFF

PV_FAC has a default of 1 and PV_OFF a default of 0.

Error Signal

The difference between the setpoint and process variable is the error signal. To suppress a small constant oscillation due to the manipulated variable quantization (for example, in pulse duration modulation with PULSEGEN), a dead band is applied to the error signal (DEADBAND). If DEADB_W = 0, the dead band is switched off.

PID Algorithm

The PID algorithm operates as a position algorithm. The proportional, integral (INT), and derivative (DIF) actions are connected in parallel and can be activated or deactivated individually. This allows P, PI, PD, and PID controllers to be configured. Pure I and D controllers are also possible.

Manual Value

It is possible to switch over between a manual and an automatic mode. In the manual mode, the manipulated variable is corrected to a manually selected value. The integrator (INT) is set internally to LMN - LMN_P - DISV and the derivative unit (DIF) to 0 and matched internally. This means that a switchover to the automatic mode does not cause any sudden change in the manipulated value.

Manipulated Value

The manipulated value can be limited to a selected value using the LMNLIMIT function. Signaling bits indicate when a limit is exceeded by the input variable. The LMN_NORM function normalizes the output of LMNLIMIT according to the following formula:

LMN = (output of LMNLIMIT) * LMN_FAC + LMN_OFF

LMN_FAC has the default 1 and LMN_OFF the default 0.

The manipulated value is also available in the peripheral format. The CPR_OUT function converts the floating-point value LMN to a peripheral value according to the following formula:

$$\text{LMN\_PER} = \text{LMN} * \frac{2764}{10}$$

Feed Forward Control

A disturbance variable can be fed forward at the DISV input.

Initialization

SFB 41 "CONT_C" has an initialization routine that is run through when the input parameter COM_RST = TRUE is set.

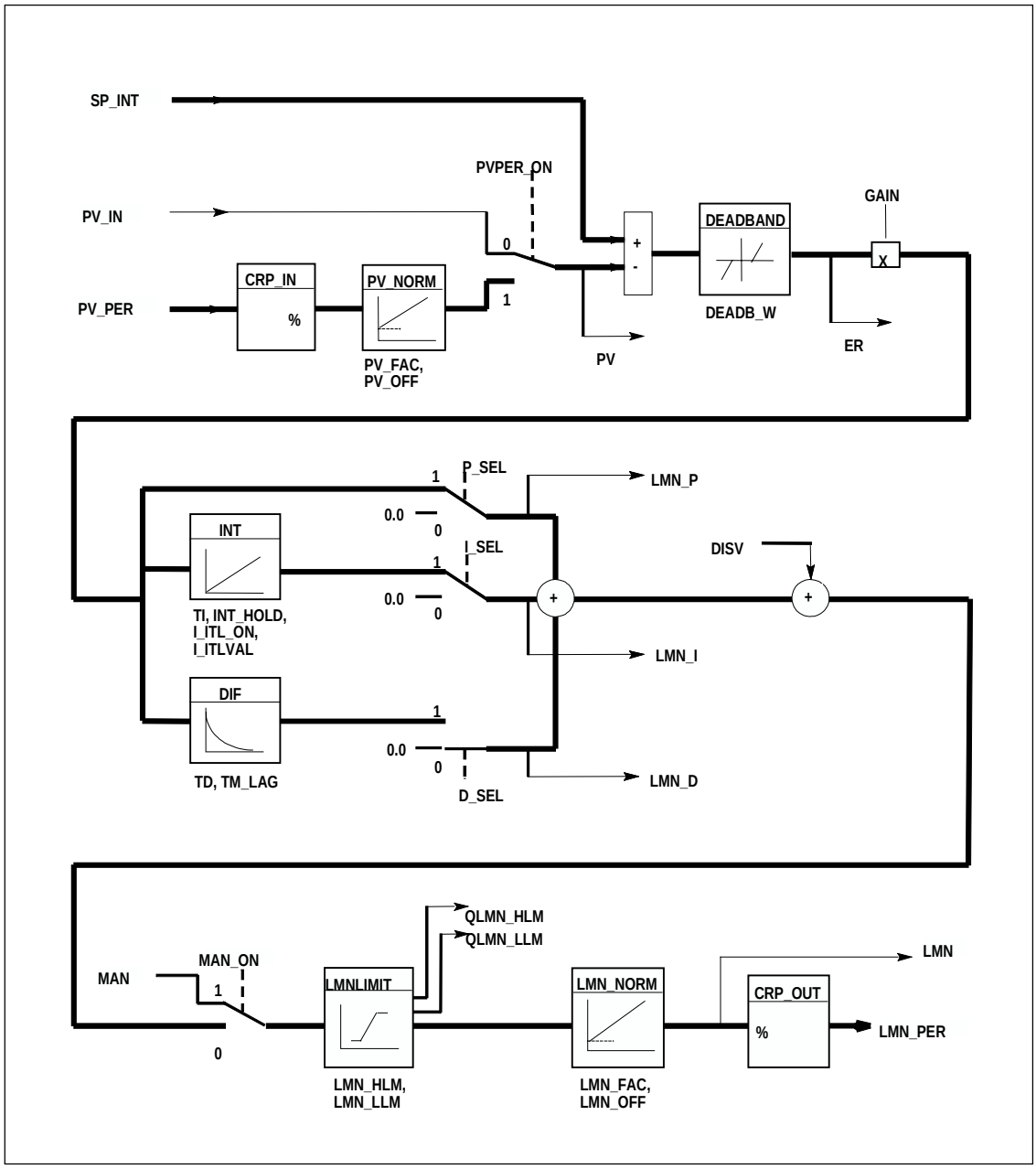
During initialization, the integrator is set internally to the initialization value I_ITVAL. When it is called in a cyclic interrupt priority class, it then continues to work starting at this value.

All other outputs are set to their default values.

Error Information

The error output parameter RET_VAL is not used.

CONT_C Block Diagram



Input Parameters

The following table contains the description of the input parameters for SFB 41/FB 41 "CONT_C."

Parameter	Data Type	Range of Values	Default	Description
COM_RST	BOOL		FALSE	COMPLETE RESTART The block has an initialization routine that is

Parameter	Data Type	Range of Values	Default	Description
				processed when the input COM_RST is set.
MAN_ON	BOOL		TRUE	MANUAL VALUE ON If the input "manual value on" is set, the control loop is interrupted. A manual value is set as the manipulated value.
PVPER_ON	BOOL		FALSE	PROCESS VARIABLE PERIPHERAL ON If the process variable is read from the I/Os, the input PV_PER must be connected to the I/Os and the input "process variable peripheral on" must be set.
P_SEL	BOOL		TRUE	PROPORTIONAL ACTION ON The PID actions can be activated or deactivated individually in the PID algorithm. The P action is on when the input "proportional action on" is set.
I_SEL	BOOL		TRUE	INTEGRAL ACTION ON The PID actions can be activated or deactivated individually in the PID algorithm. The I action is on when the input "integral action on" is set.
INT_HOLD	BOOL		FALSE	INTEGRAL ACTION HOLD The output of the integrator can be "frozen" by setting the input "integral action hold."
I_ITL_ON	BOOL		FALSE	INITIALIZATION OF THE INTEGRAL ACTION ON The output of the integrator can be connected to the input I_ITL_VAL by setting the input "initialization of the integral action on."
D_SEL	BOOL		FALSE	DERIVATIVE ACTION ON The PID actions can be activated or deactivated individually in the PID algorithm. The D action is on when the input "derivative action on" is set.
CYCLE	TIME	>= 1 ms	T#1s	SAMPLING TIME The time between the block calls must be constant. The "sampling time" input specifies the time between block calls.
SP_INT	REAL	-100.0 to +100.0 (%) or phys. value 1)	0.0	INTERNAL SETPOINT The "internal setpoint" input is used to specify a setpoint.
PV_IN	REAL	-100.0 to +100.0 (%) or phys. Value 1)	0.0	PROCESS VARIABLE IN An initialization value can be set at the "process variable in" input or an external process variable in floating point format can be connected.
PV_PER	WORD		W#16#0000	PROCESS VARIABLE PERIPHERAL The process variable in the I/O format is connected to the controller at the "process variable peripheral" input.
MAN	REAL	-100.0 to +100.0 (%) or phys. Value	0.0	MANUAL VALUE The "manual value" input is used to set a

Parameter	Data Type	Range of Values	Default	Description
		2)		manual value using the operator interface functions.
GAIN	REAL		2.0	PROPORTIONAL GAIN The "proportional value" input specifies the controller gain.
TI	TIME	>= CYCLE	T#20s	RESET TIME The "reset time" input determines the time response of the integrator.
TD	TIME	>= CYCLE	T#10s	DERIVATIVE TIME The "derivative time" input determines the time response of the derivative unit.
TM_LAG	TIME	>= CYCLE/2	T#2s	TIME LAG OF THE DERIVATIVE ACTION The algorithm of the D action includes a time lag that can be assigned at the "time lag of the derivative action" input.
DEADB_W	REAL	>= 0.0 (%) or phys. Value 1)	0.0	DEAD BAND WIDTH A dead band is applied to the error. The "dead band width" input determines the size of the dead band.
LMN_HLM	REAL	LMN_LLM ...100.0 (%) or phys. Value 2)	100.0	MANIPULATED VALUE HIGH LIMIT The manipulated value is always limited by an upper and lower limit. The "manipulated value high limit" input specifies the upper limit.
LMN_LLM	REAL	-100.0... LMN_HLM (%) or phys. Value 2)	0.0	MANIPULATED VALUE LOW LIMIT The manipulated value is always limited by an upper and lower limit. The "manipulated value low limit" input specifies the lower limit.
PV_FAC	REAL		1.0	PROCESS VARIABLE FACTOR The "process variable factor" input is multiplied by the process variable. The input is used to adapt the process variable range.
PV_OFF	REAL		0.0	PROCESS VARIABLE OFFSET The "process variable offset" input is added to the process variable. The input is used to adapt the process variable range.
LMN_FAC	REAL		1.0	MANIPULATED VALUE FACTOR The "manipulated value factor" input is multiplied by the manipulated value. The input is used to adapt the manipulated value range.
LMN_OFF	REAL		0.0	MANIPULATED VALUE OFFSET The "manipulated value offset" is added to the manipulated value. The input is used to adapt the manipulated value range.
I_ITLVAL	REAL	-100.0 to +100.0 (%) or phys. Value 2)	0.0	INITIALIZATION VALUE OF THE INTEGRAL ACTION The output of the integrator can be set at input I_ITL_ON. The initialization value is applied to the input "initialization value of

Parameter	Data Type	Range of Values	Default	Description
				the integral action."
DISV	REAL	-100.0 to +100.0 (%) or phys. Value 2)	0.0	DISTURBANCE VARIABLE For feed forward control, the disturbance variable is connected to input "disturbance variable."

1) Parameters in the setpoint and process variable branches with the same unit
2) Parameters in the manipulated value branch with the same unit

Output Parameters

The following table contains the description of the output parameters for SFB 41/FB41 "CONT_C."

Parameter	Data Type	Range of Values	Default	Description
LMN	REAL		0.0	MANIPULATED VALUE The effective manipulated value is output in floating point format at the "manipulated value" output.
LMN_PER	WORD		W#16#0000	MANIPULATED VALUE PERIPHERAL The manipulated value in the I/O format is connected to the controller at the "manipulated value peripheral" output.
QLMN_HLM	BOOL		FALSE	HIGH LIMIT OF MANIPULATED VALUE REACHED The manipulated value is always limited to an upper and lower limit. The output "high limit of manipulated value reached" indicates that the upper limit has been exceeded.
QLMN_LLM	BOOL		FALSE	LOW LIMIT OF MANIPULATED VALUE REACHED The manipulated value is always limited to an upper and lower limit. The output "low limit of manipulated value reached" indicates that the lower limit has been exceeded.
LMN_P	REAL		0.0	PROPORTIONAL COMPONENT The "proportional component" output contains the proportional component of the manipulated variable.
LMN_I	REAL		0.0	INTEGRAL COMPONENT The "integral component" output contains the integral component of the manipulated value.
LMN_D	REAL		0.0	DERIVATIVE COMPONENT The "derivative component" output contains the derivative component of the manipulated value.

Parameter	Data Type	Range of Values	Default	Description
PV	REAL		0.0	PROCESS VARIABLE The effective process variable is output at the "process variable" output.
ER	REAL		0.0	ERROR SIGNAL The effective error is output at the "error signal" output.

24.2 Step Control with SFB 42/FB 42 "CONT_S"

Introduction

SFB/FB "CONT_S" is used on SIMATIC S7 programmable logic controllers to control technical processes with digital manipulated value output signals for integrating actuators. During parameter assignment, you can activate or deactivate sub-functions of the PI step controller to adapt the controller to the process. You can easily do this by using the parameter assignment tool (Menu path: **Start > Simatic > Step7 > Assign PID Control parameters**). The online electronic manual is found under **Start > Simatic > Step7 > Assign PID Control English**.

Application

You can use the controller as a PI fixed setpoint controller or in secondary control loops in cascade, blending or ratio controllers, however not as the primary controller. The functions of the controller are based on the PI control algorithm of the sampling controller supplemented by the functions for generating the binary output signal from the analog actuating signal.

The following applies starting with **FB V1.5 or V1.1.0 of CPU 314 IFM**:

With TI = T#0ms, the integral component of the controller can be disabled, thus allowing the block to be used as a proportional controller.

Since the controller works without any position feedback signal, the internally calculated manipulated variable will not exactly match the signal control element position. An adjustment is made if the manipulated variable (ER * GAIN) is negative. The controller then sets the output QLMNDN (manipulated value signal low) until LMNR_LS (lower limit of the position feedback signal) is set.

The controller can also be used as a secondary actuator in a controller cascade. The setpoint input SP_INT is used to assign the control element position. In this case the actual value input and the parameter TI (integration time) must be set to zero. An application example is temperature regulation by means of heat output control using pulse-pause control and cooling capacity control by means of a valve. In this case, to close the valve completely, the manipulated variable (ER * GAIN) should have a negative setting.

Description

Apart from the functions in the process value branch, the SFB implements a complete PI controller with a digital manipulated value output and the option of influencing the manipulated value manually. The step controller operates without a position feedback signal.
In the following you will find the description of the partial functions:

Setpoint Branch

The setpoint is entered in floating-point format at the **SP_INT** input.

Process Variable Branch

The process variable can be input in the peripheral (I/O) or floating-point format. The CRP_IN function converts the PV_PER peripheral value to a floating-point format of -100 to +100 % according to the following formula:

$$\text{Output of CPR\_IN} = \text{PV\_PER} * \frac{100}{27648}$$

The PV_NORM function normalizes the output of CRP_IN according to the following formula:

Output of PV_NORM = (output of CPR_IN) * PV_FAC + PV_OFF

PV_FAC has a default of 1 and PV_OFF a default of 0.

Error Signal

The difference between the setpoint and process variable is the error signal. To suppress a small constant oscillation due to the manipulated variable quantization (for example, due to a limited resolution of the manipulated value by the actuator valve), a dead band is applied to the error signal (DEADBAND). If DEADB_W = 0, the dead band is switched off.

PI Step Algorithm

The SFB/FB operates without a position feedback signal. The I action of the PI algorithm and the assumed position feedback signal are calculated in **one** integrator (INT) and compared with the remaining P action as a feedback value. The difference is applied to a three-step element (THREE_ST) and a pulse generator (PULSEOUT) that creates the pulses for the actuator. The switching frequency of the controller can be reduced by adapting the threshold on of the three-step element.

Feed Forward Control

A disturbance variable can be fed forward at the **DISV** input.

Initialization

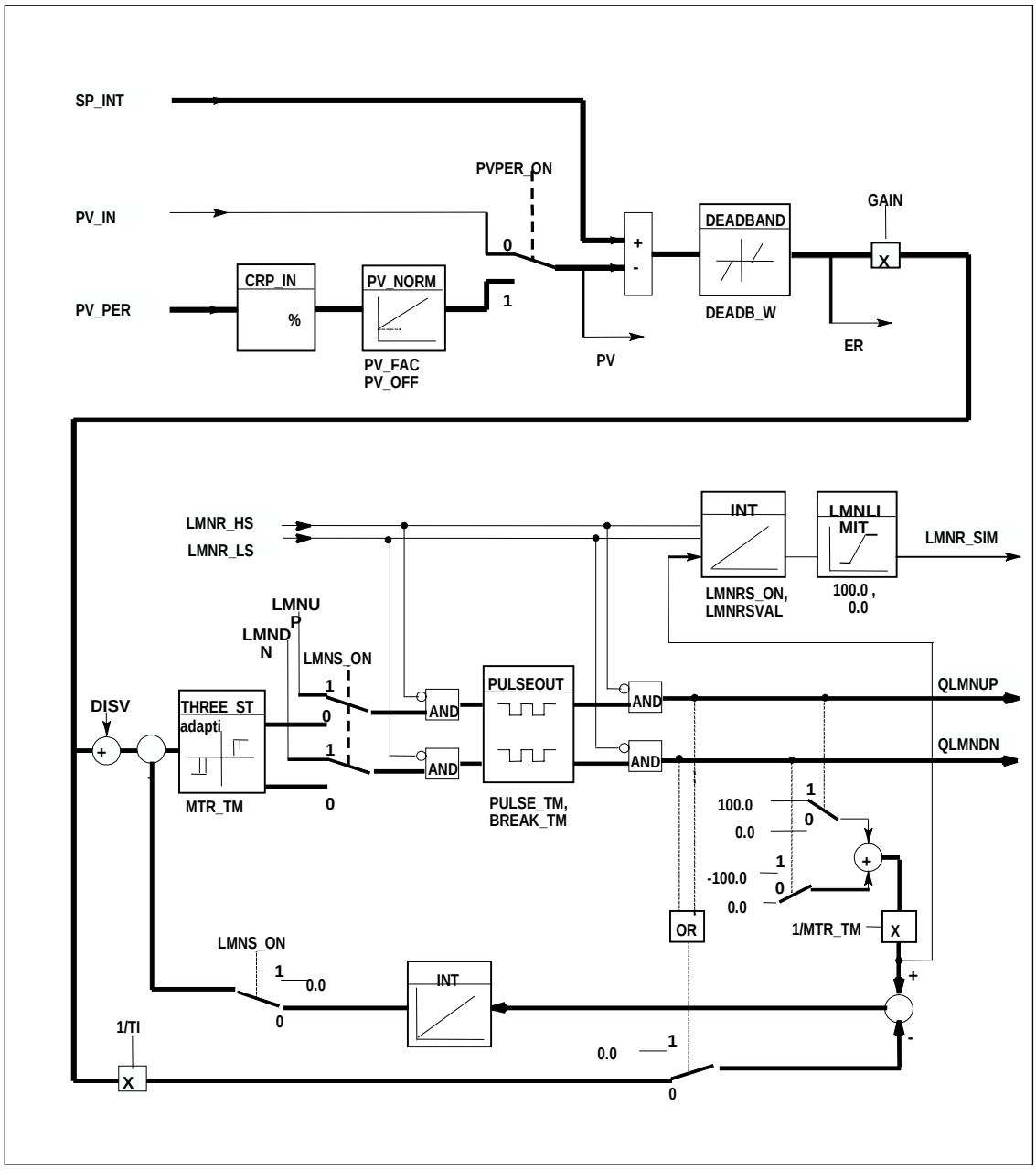
SFB/FB42 "CONT_S" has an initialization routine that is run through when the input parameter COM_RST = TRUE is set.

All other outputs are set to their default values.

Error Information

The error output parameter RET_VAL is not used.

Block Diagram



Input Parameters

The following table contains the description of the input parameters for SFB 42/FB 42 "CONT_S."

Parameter	Data Type	Values	Default	Description
COM_RST	BOOL		FALSE	COMPLETE RESTART The block has an initialization routine that is processed when the input COM_RST is set.
LMNR_HS	BOOL		FALSE	HIGH LIMIT OF POSITION FEEDBACK SIGNAL The "actuator at upper limit stop" signal is connected to the "high limit of position feedback signal" input. LMNR_HS=TRUE means the actuator is at upper limit stop.
LMNR_LS	BOOL		FALSE	LOW LIMIT OF POSITION FEEDBACK SIGNAL The "actuator at lower limit stop" signal is connected to the "low limit of position feedback signal" input. LMNR_LS=TRUE means the actuator is at lower limit stop.
LMNS_ON	BOOL		TRUE	MANUAL ACTUATING SIGNALS ON The actuating signal processing is switched to manual at the "manual actuating signals on" input.
LMNUP	BOOL		FALSE	ACTUATING SIGNALS UP With manual actuating value signals, the output signal QLMNUP is set at the input "actuating signals up."
LMNDN	BOOL		FALSE	ACTUATING SIGNALS DOWN With manual actuating value signals, the output signal QLMNDN is set at the input "actuating signals down."
PVPER_ON	BOOL		FALSE	PROCESS VARIABLE PERIPHERAL ON If the process variable is read in from the I/Os, the input PV_PER must be connected to the I/Os and the input "process variable peripheral on" must be set.
CYCLE	TIME	>= 1ms	T#1s	SAMPLING TIME The time between the block calls must be constant. The "sampling time" input specifies the time between block calls.
SP_INT	REAL	-100.0 ... +100.0 (%) or phys. value 1)	0.0	INTERNAL SETPOINT The "internal setpoint" input is used to specify a setpoint.
PV_IN	REAL	-100.0 ... +100.0 (%) or phys. value 1)	0.0	PROCESS VARIABLE IN An initialization value can be set at the "process variable in" input or an external process variable in floating point format can be connected.
PV_PER	WORD		W#16#0000	PROCESS VARIABLE PERIPHERAL The process variable in the I/O format is connected to the controller at the "process variable peripheral" input.

Parameter	Data Type	Values	Default	Description
GAIN	REAL		2.0	PROPORTIONAL GAIN The "proportional gain" input sets the controller gain.
TI	TIME	>= CYCLE	T#20s	RESET TIME The "reset time" input determines the time response of the integrator.
DEADB_W	REAL	0.0...100.0 (%) or phys. value 1)	1.0	DEAD BAND WIDTH A dead band is applied to the error. The "dead band width" input determines the size of the dead band.
PV_FAC	REAL		1.0	PROCESS VARIABLE FACTOR The "process variable factor" input is multiplied by the process variable. The input is used to adapt the process variable range.
PV_OFF	REAL		0.0	PROCESS VARIABLE OFFSET The "process variable offset" input is added to the process variable. The input is used to adapt the process variable range.
PULSE_TM	TIME	>= CYCLE	T#3s	MINIMUM PULSE TIME A minimum pulse duration can be assigned with the parameter "minimum pulse time."
BREAK_TM	TIME	>= CYCLE	T#3s	MINIMUM BREAK TIME A minimum break duration can be assigned with the parameter "minimum break time."
MTR_TM	TIME	>= CYCLE	T#30s	MOTOR ACTUATING TIME The time required by the actuator to move from limit stop to limit stop is entered at the "motor actuating time" parameter.
DISV	REAL	-100.0...100.0 (%) or phys. value 2)	0.0	DISTURBANCE VARIABLE For feed forward control, the disturbance variable is connected to input "disturbance variable."

1) Parameters in the setpoint and process variable branches with the same unit
2) Parameters in the manipulated value branch with the same unit

Output Parameters

The following table contains the description of the output parameters for SFB 42/FB 42 "CONT_S."

Parameter	Data Type	Values	Default	Description
QLMNUP	BOOL		FALSE	ACTUATING SIGNAL UP If the output "actuating signal up" is set, the actuating valve is opened.
QLMNDN	BOOL		FALSE	ACTUATING SIGNAL DOWN If the output "actuating signal down" is set, the

Parameter	Data Type	Values	Default	Description
				actuating valve is opened.
PV	REAL		0.0	PROCESS VARIABLE The effective process variable is output at the "process variable" output.
ER	REAL		0.0	ERROR SIGNAL The effective error is output at the "error signal" output.

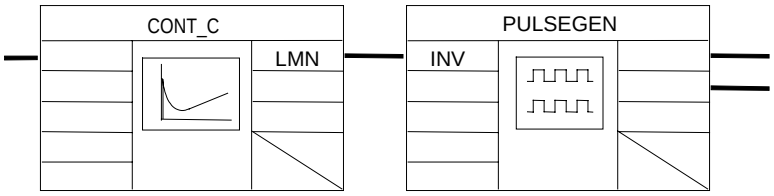
24.3 Pulse Generation with SFB 43/FB 43 "PULSEGEN"

Introduction

SFB 43 "PULSEGEN" is used to structure a PID controller with pulse output for proportional actuators. The electronic manual is found under **Start > Simatic > S7 Manuals > PID Control English.**

Application

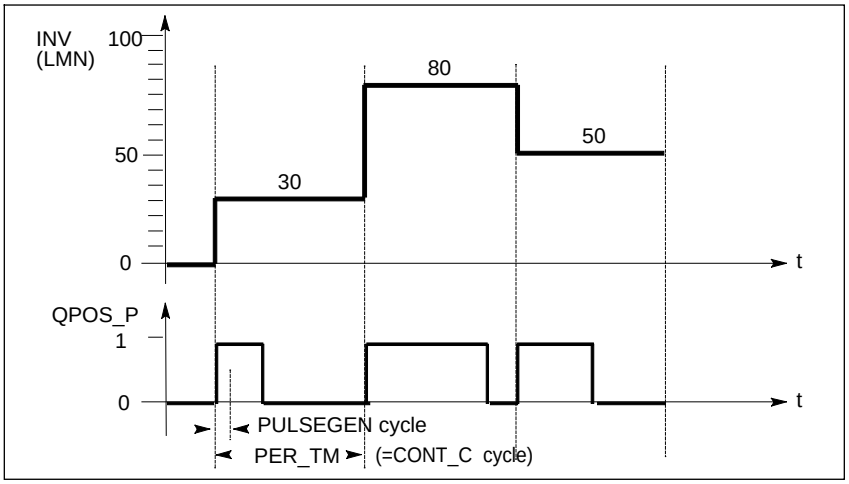
Using SFB/FB "PULSEGEN," PID two or three step controllers with pulse duration modulation can be configured. The function is normally used in conjunction with the continuous controller "CONT_C."



Description

The PULSEGEN function transforms the input variable INV (= manipulated value of the PID controller) by modulating the pulse duration into a pulse train with a constant period, corresponding to the cycle time at which the input variable is updated and which must be assigned in PER_TM.

The duration of a pulse per period is proportional to the input variable. The cycle assigned to PER_TM is not identical to the processing cycle of the SFB/FB "PULSEGEN." The PER_TM cycle is made up of several processing cycles of SFB/FB "PULSEGEN," whereby the number of SFB/FB "PULSEGEN" calls per PER_TM cycle is the yardstick for the accuracy of the pulse duration modulation.

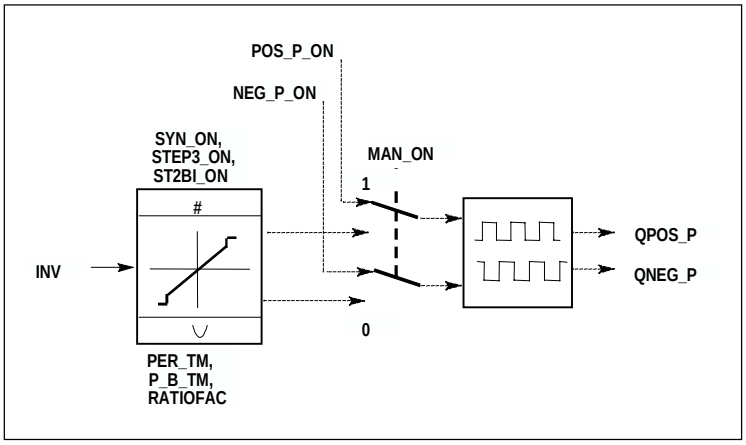


Pulse Duration Modulation

An input variable of 30% and 10 SFB/FB "PULSEGEN" calls per PER_TM means the following:

- "One" at the QPOS output for the first three calls of SFB/FB "PULSEGEN" (30% of 10 calls)
- "Zero" at the QPOS output for seven further calls of SFB/FB "PULSEGEN" (70% of 10 calls)

Block Diagram



Accuracy of the Manipulated Value

With a "sampling ratio" of 1:10 (CONT_C calls to PULSEGEN calls) the accuracy of the manipulated value in this example is restricted to 10%, in other words, set input values INV can only be simulated by a pulse duration at the QPOS output in steps of 10 %.

The accuracy is increased as the number of SFB/FB "PULSEGEN" calls per CONT_C call is increased.

If PULSEGEN is called, for example, 100 times more often than CONT_C, a resolution of 1 % of the manipulated value range is achieved.

Note

The call frequency must be programmed by the user.

Automatic Synchronization

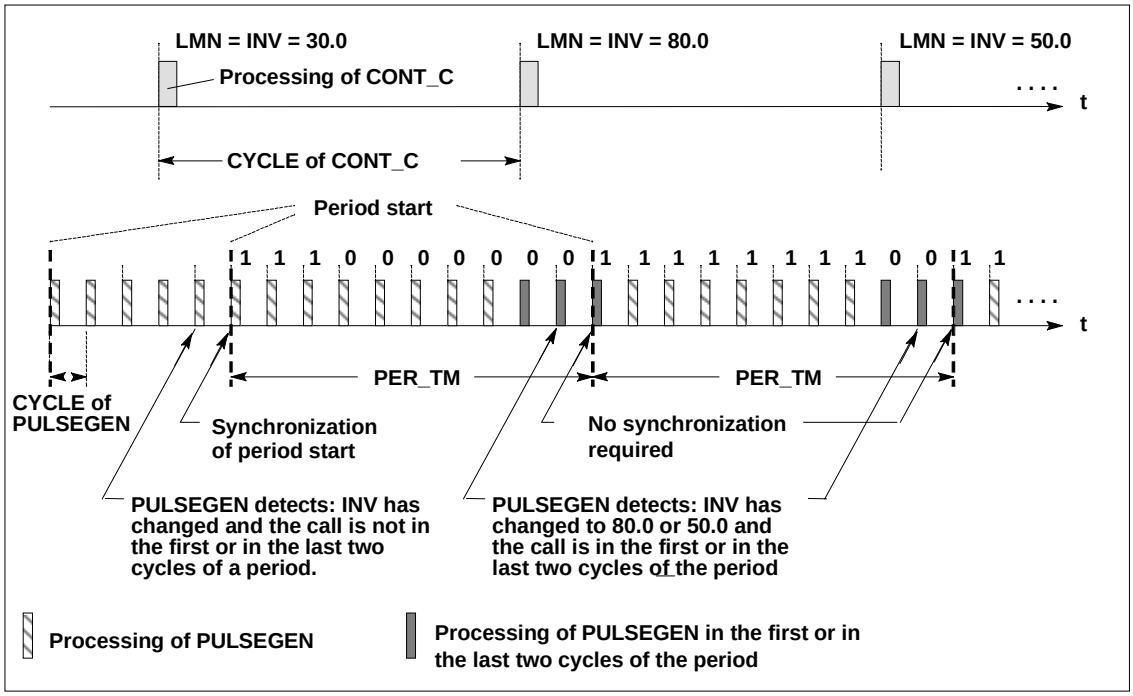
It is possible to synchronize the pulse output with the block that updates the input variable INV (for example, CONT_C). This ensures that a change in the input variable is output as quickly as possible as a pulse.

The pulse generator evaluates the input value INV at intervals corresponding to the period PER_TM and converts the value into a pulse signal of corresponding length.

Since, however, INV is usually calculated in a slower cyclic interrupt class, the pulse generator should start the conversion of the discrete value into a pulse signal as soon as possible after the updating of INV.

To allow this, the block can synchronize the start of the period using the following procedure:

If INV changes and if the block call is not in the first or last two call cycles of a period, the synchronization is performed. The pulse duration is recalculated and in the next cycle is output with a new period.



The automatic synchronization can be disabled at the "SYN_ON" input (= FALSE).

Note

With the beginning of a new period, the old value of INV (in other words, of LMN) is simulated in the pulse signal more or less accurately following the synchronization.

Modes

Depending on the parameters assigned to the pulse generator, PID controllers with a three-step output or with a bipolar or unipolar two-step output can be configured. The following table illustrates the setting of the switch combinations for the possible modes.

		Switch	
Mode	**MAN_ON**	**STEP3_ON**	**ST2BI_ON**
Three-step control	FALSE	TRUE	Any
Two-step control with bipolar control range (-100 % to +100 %)	FALSE	FALSE	TRUE
Two-step control with unipolar control range (0 % ... 100 %)	FALSE	FALSE	FALSE
Manual mode	TRUE	Any	Any

Three-Step Control

In the "three-step control" mode, the actuating signal can adopt three states. The values of the binary output signals QPOS_P and QNEG_P are assigned to the statuses of the actuator.
The table shows the example of a temperature control:

		Actuator	
Output Signals	**Heat**	**Off**	**Cool**
QPOS_P	TRUE	FALSE	FALSE
QNEG_P	FALSE	FALSE	TRUE

Based on the input variable, a characteristic curve is used to calculate a pulse duration. The form of the characteristic curve is defined by the minimum pulse or minimum break time and the ratio factor.

The normal value for the ratio factor is 1.

The "doglegs" in the curves are caused by the minimum pulse or minimum break times.

Minimum Pulse or Minimum Break Time

A correctly assigned minimum pulse or minimum break time P_B_TM can prevent short on/off times that reduce the working life of switching elements and actuators.

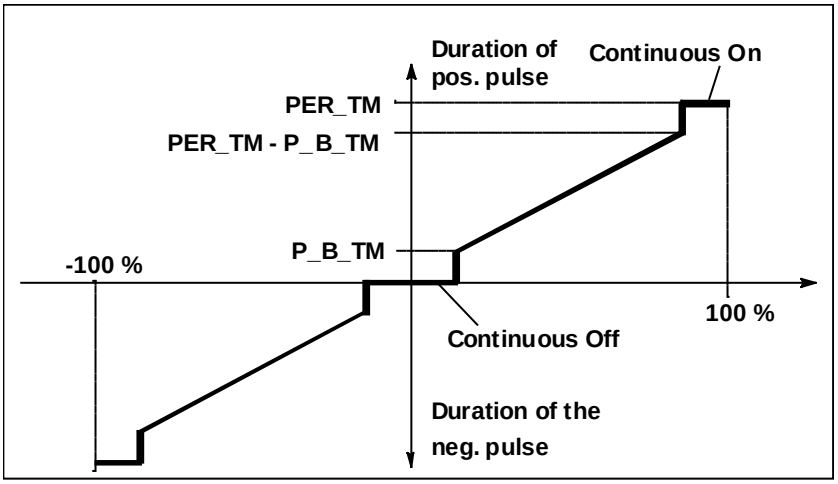
Note

Small absolute values at the input variable LMN that could otherwise generate a pulse duration shorter than P_B_TM are suppressed. Large input values that would generate a pulse duration longer than (PER_TM - P_B_TM) are set to 100 % or -100 %.

The duration of the positive or negative pulses is calculated from the input variable (in %) multiplied by the period time.

$$\text{Period Time} = \frac{\text{INV}}{100} * \text{PER\_TM}$$

The following figure shows a symmetrical curve of a three-step controller (ratio factor = 1).



Three-Step Control Asymmetrical

Using the ratio factor RATIOFAC, the ratio of the duration of positive to negative pulses can be changed. In a thermal process, for example, this would allow different system time constants for heating and cooling.

The ratio factor also influences the minimum pulse or minimum break time. A ratio factor < 1 means that the threshold value for negative pulses is multiplied by the ratio factor.

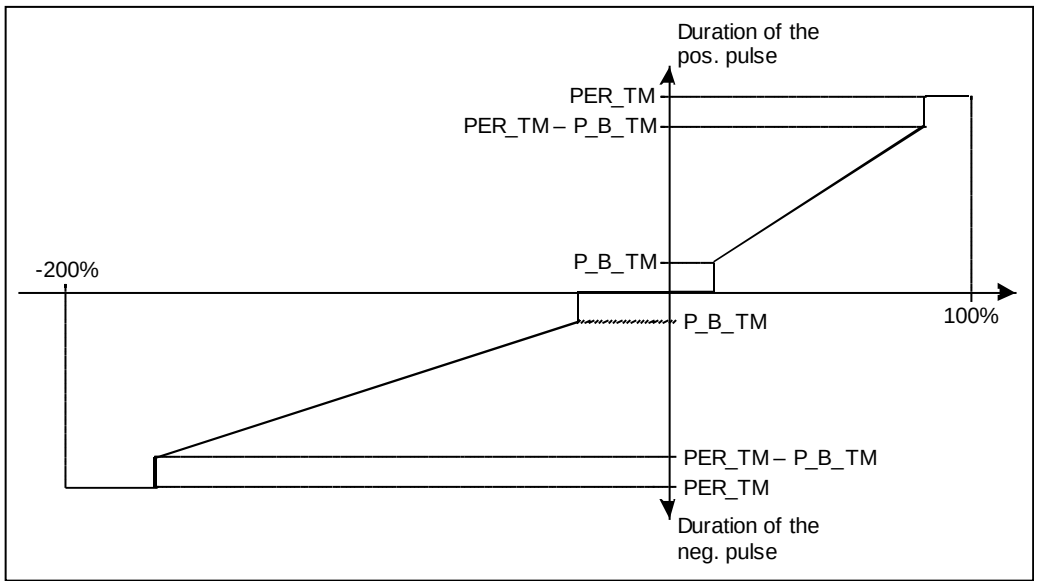
Ratio Factor < 1

The pulse duration at the negative pulse output calculated from the input variable multiplied by the period time is reduced by the ratio factor.

$$\text{Duration of the positive pulse} = \frac{\text{INV}}{100} * \text{PER\_TM}$$

$$\text{Duration of the negative pulse} = \frac{\text{INV}}{100} * \text{PER\_TM} \_ * \text{RATIOFAC}$$

The following figure shows the asymmetric curve of the three-step controller (ratio factor = 0.5):



Ratio Factor > 1

The pulse duration at the positive pulse output calculated from the input variable multiplied by the period time is reduced by the ratio factor.

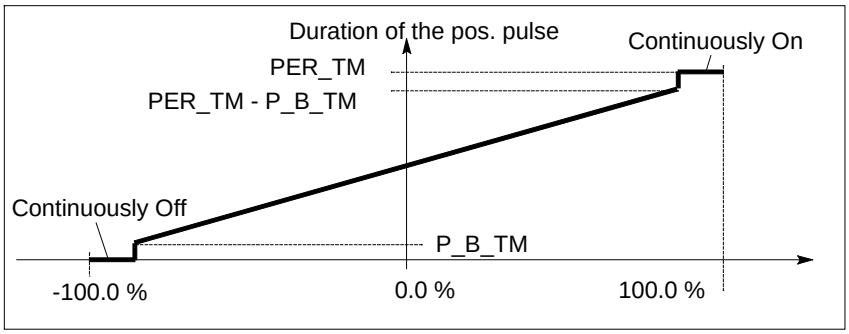
$$\text{Duration of the negative pulse} = \frac{INV}{100} * PER\_TM$$

$$\text{Duration of the positive pulse} \quad \frac{INV}{100} * \frac{PER\_T}{RATIOFAC}$$

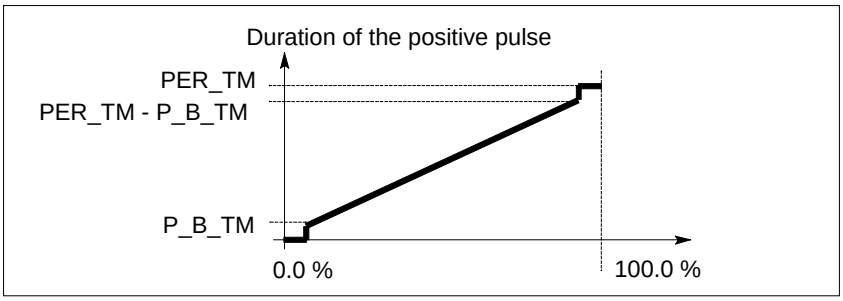
Two-Step Control

In two-step control, only the positive pulse output QPOS_P of PULSEGEN is connected to the on/off actuator. Depending on the manipulated value range being used, the two-step controller has a bipolar or a unipolar manipulated value range.

Two-Step Control with Bipolar Manipulated Variable Range (-100% to 100%)



Two-Step Control with Unipolar Manipulated Variable Range (0% to 100%)



The negated output signal is available at QNEG_P if the connection of the two-step controller in the control loop requires a logically inverted binary signal for the actuating pulses.

	Actuator	
Pulse	**On**	**Off**
QPOS_P	TRUE	FALSE
QNEG_P	FALSE	TRUE

Manual Mode in Two/Three-Step Control

In the manual mode (MAN_ON = TRUE), the binary outputs of the three-step or two-step controller can be set using the signals POS_P_ON and NEG_P_ON regardless of INV.

	POS_P_ON	**NEG_P_ON**	**QPOS_P**	**QNEG_P**
Three-step control	FALSE	FALSE	FALSE	FALSE
	TRUE	FALSE	TRUE	FALSE

	POS_P_ON	NEG_P_ON	QPOS_P	QNEG_P
	FALSE	TRUE	FALSE	TRUE
	TRUE	TRUE	FALSE	FALSE
Two-step control	FALSE	Any	FALSE	TRUE
	TRUE	Any	TRUE	FALSE

Initialization

SFB/FB "PULSGEN" has an initialization routine that is run through when the input parameter COM_RST = TRUE is set.

All the signal outputs are set to 0.

Error Information

The error output parameter RET_VAL is not used.

Input Parameters

Parameter	Data Type	Range of Values	Default	Description
INV	REAL	-100.0...100.0 (%)	0.0	INPUT VARIABLE An analog manipulated value is connected to the input parameter "input variable."
PER_TM	TIME	>=20*CYCLE	T#1s	PERIOD TIME The constant period of pulse duration modulation is input with the "period time" input parameter. This corresponds to the sampling time of the controller. The ratio between the sampling time of the pulse generator and the sampling time of the controller determines the accuracy of the pulse duration modulation.
P_B_TM	TIME	>= CYCLE	T#0ms	MINIMUM PULSE/BREAK TIME A minimum pulse or minimum break time can be assigned at the input parameters "minimum pulse or minimum break time."
RATIOFAC	REAL	0.1 ...10.0	1.0	RATIO FACTOR The input parameter "ratio factor" can be used to change the ratio of the duration of negative to positive pulses. In a thermal process, this would, for example, allow different time constants for heating and cooling to be compensated (for example, in a process with electrical heating and water cooling).
STEP3_ON	BOOL		TRUE	THREE STEP CONTROL ON The "three-step control on" input parameter activates this mode. In three-step control, both output signals are active.

Parameter	Data Type	Range of Values	Default	Description
ST2BI_ON	BOOL		FALSE	TWO STEP CONTROL FOR BIPOLAR MANIPULATED VALUE RANGE ON With the input parameter "two-step control for bipolar manipulated value range on" you can select between the modes "two-step control for bipolar manipulated value" and "two-step control for unipolar manipulated value range." The parameter STEP3_ON = FALSE must be set.
MAN_ON	BOOL		FALSE	MANUAL MODE ON By setting the input parameter "manual mode on," the output signals can be set manually.
POS_P_ON	BOOL		FALSE	POSITIVE PULSE ON In the manual mode with three-step control, the output signal QPOS_P can be set at the input parameter "positive pulse on." In the manual mode with two-step control, QNEG_P is always set inversely to QPOS_P.
NEG_P_ON	BOOL		FALSE	NEGATIVE PULSE ON In the manual mode with three-step control, the output signal QNEG_P can be set at the input parameter "negative pulse on." In the manual mode with two-step control, QNEG_P is always set inversely to QPOS_P.
SYN_ON	BOOL		TRUE	SYNCHRONIZATION ON By setting the input parameter "synchronization on," it is possible to synchronize automatically with the block that updates the input variable INV. This ensures that a changing input variable is output as quickly as possible as a pulse.
COM_RST	BOOL		FALSE	COMPLETE RESTART The block has an initialization routine that is processed when the COM_RST input is set
CYCLE	TIME	>= 1ms	T#10ms	SAMPLING TIME The time between block calls must be constant. The "sampling time" input specifies the time between block calls.

Note

The values of the input parameters are not limited in the block. There is no parameter check.

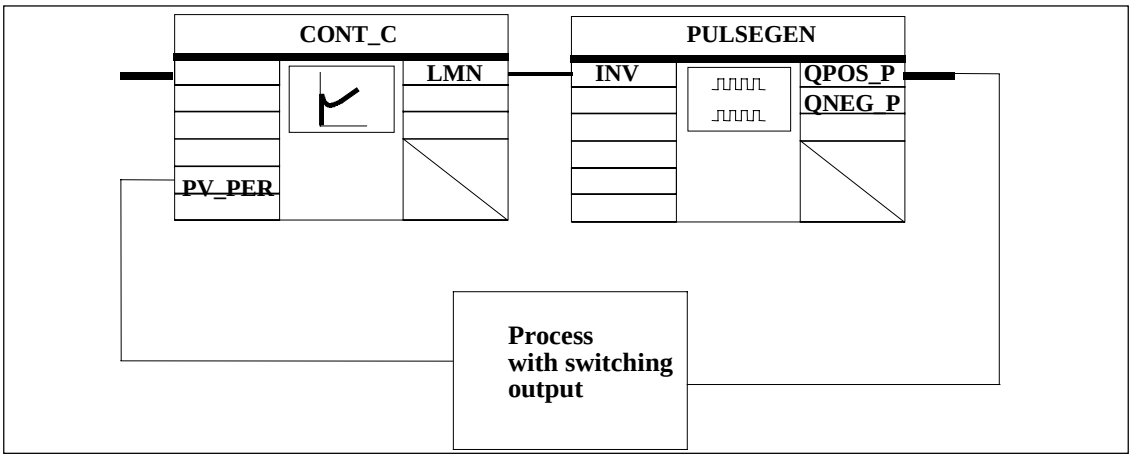
Output Parameters

Parameter	Data Type	Values	Default	Description
QPOS_P	BOOL		FALSE	OUTPUT POSITIVE PULSE The output parameter "output positive pulse" is set when a pulse is to be output. In three-step control, this is always the positive pulse. In two-step control, QNEG_P is always set inversely to QPOS_P.
QNEG_P	BOOL		FALSE	OUTPUT NEGATIVE PULSE The output parameter "output negative pulse" is set when a pulse is to be output. In three-step control, this is always the negative pulse. In two-step control, QNEG_P is always set inversely to QPOS_P.

24.4 Example of the PULSEGEN Block

Control Loop

With the continuous controller CONT_C and the pulse generator PULSEGEN, you can implement a fixed setpoint controller with a switching output for proportional actuators. The following figure shows the signal flow of the control loop.



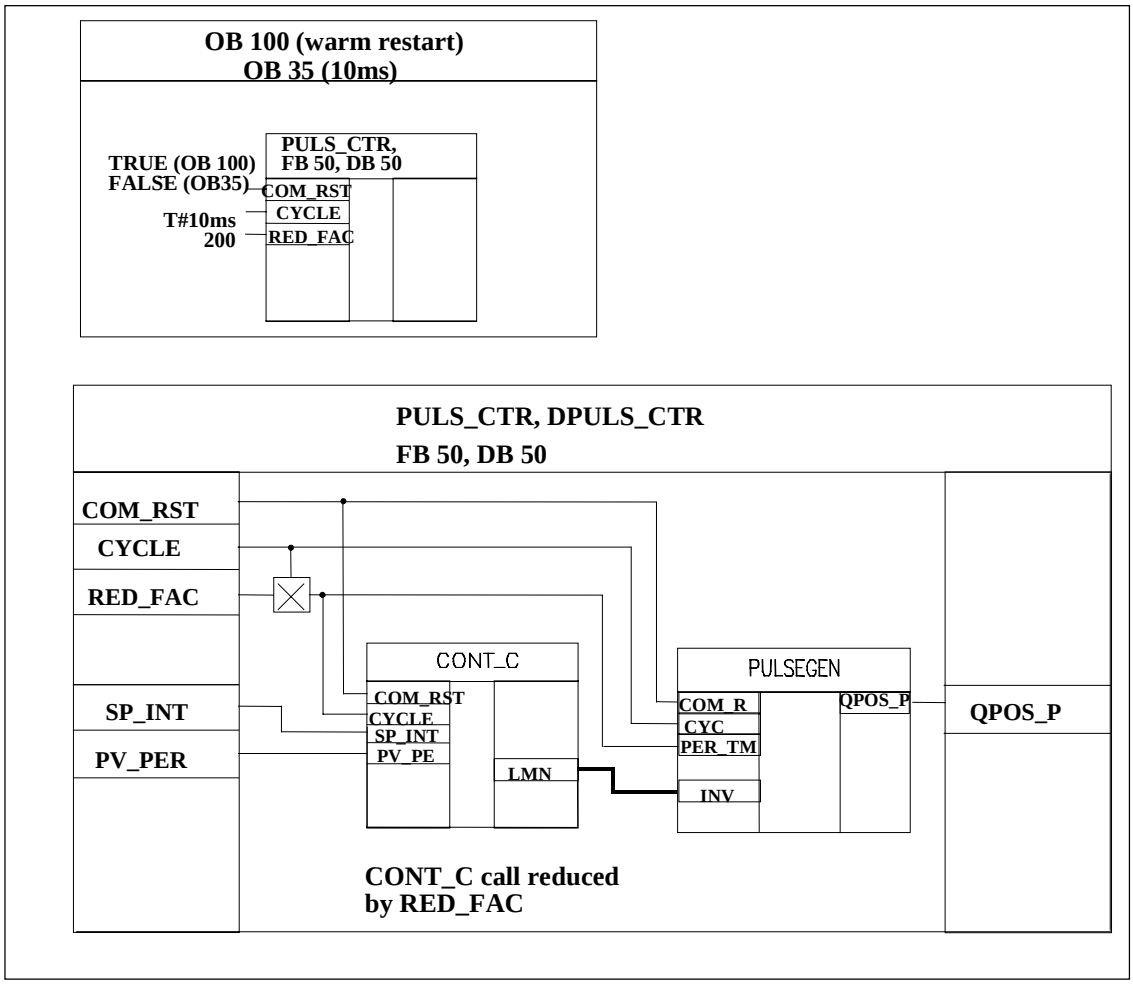
The continuous controller CONT_C forms the manipulated value LMN that is converted by the pulse generator PULSEGEN into pulse/break signals QPOS_P or QNEG_P.

Calling the Block and Connecting it

The fixed setpoint controller with switching output for proportional actuators PULS_CTR consists of the blocks CONT_C and PULSEGEN. The block call is

implemented so that CONT_C is called every 2 seconds (=CYCLE*RED_FAC) and PULSEGEN every 10 ms (=CYCLE). The cycle time of OB35 is set to 10 ms. The interconnection can be seen in the following figure.

During a warm restart, the block PULS_CTR is called in OB100 and the input COM_RST is set to TRUE.



STL Program for FB PULS_CTR

Address	Declaration	Name	Type	Comment
0.0	in	SP_INT	REAL	Setpoint
4.0	in	PV_PER	WORD	Process variable peripheral
6.0	in	RED_FAC	INT	Call reduction factor
8.0	in	COM_RST	BOOL	Complete restart
10.0	in	CYCLE	TIME	Sampling time
14.0	out	QPOS_P	BOOL	Actuating signal

Address	Declaration	Name	Type	Comment
16.0	stat	DI_CONT_C	FB-CONT_C	Counter
142.0	stat	DI_PULSEGEN	FB-PULSEGEN	Counter
176.0	stat	SCount	INT	Counter
0.0	temp	TCycCtr	TIME	Controller sampling time

STL			Description
	A	#COM_RST	//Initialization routine
	JCN	M001	
	L	0	
	T	#sCount	
M001: L	#CYCLE		//Calculate controller sampling time
	L	#RED_FAC	
	*D		
	T	#tCycCtr	
	L	#sCount	//Decrement counter and compare with zero
	L	1	
	-I		
	T	#sCount	
	L	0	
	<=I		
	JCN	M002	//Conditional block call and set counter
	CALL	#DI_CONT_C	
	COM_RST	:=#COM_RST	
	CYCLE	:=#tCycCtr	
	SP_INT	:=#SP_INT	
	PV_PER	:=#PV_PER	
	L	#RED_FAC	
	T	#sCount	
M002:	L	#DI_CONT_C.LMN	
	T	#DI_PULSEGEN.INV	
	CALL	#DI_PULSEGEN	
	PER_TM	:=#tCycCtr	
	COM_RST	:=#COM_RST	
	CYCLE	:=#CYCLE	
	QPOS_P	:=#QPOS_P	
	BE		

25 SFBs for Compact CPUs

25.1 Positioning With Analog Output Using SFB 44 "Analog"

Description

To control the positioning functions via the user program, use **SFB ANALOG (SFB 44)**.

A fixed assigned analog output controls the power stage with a voltage (**voltage signal)** of ±10 V or with a current (**current signal**) of ±20 mA.

- After the acceleration phase (**RAM_UP)** the drive approaches the target with the speed (**$V_{Setpoint}$**).
- At the **braking point**, that is calculated by the CPU, the deceleration (**RAMP_DN)** up to the change-over point is initialized.
- Once the **change-over point** is reached, run is continued with creep speed **(V_{Creep})**.
- The drive is switched off at the **cut-off point**.
- The change-over point and the cut-off point are determined for every Step Approach in the parameter values **change-over difference** and **cut-off difference** you have specified. The change-over difference and cut-off difference can be determined differently for the forward motion (in plus direction) and for the reverse motion (in minus direction).
- The run is completed (**WORKING** = FALSE) when the cut-off point is reached. A new run can then be started.
- The specified target is reached (**POS_RCD** = TRUE) when the actual position value has reached the **target range**. If the actual position value drifts off without a new run having been started the "Position reached" signal is not reset again.

When the change-over difference is smaller than the cut-off difference, the drive is slowed down as of the braking point to the speed setpoint 0.

Basic Parameters:

Here we describe the SFB parameters which apply to all operating modes. The parameters specific to the operating mode are described with the individual operating modes.

Parameters:

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
LADDR	INPUT	WORD	0	CPU specific	W#16#0310	The I/O address of your submodule, specified by you in "HW Config". If the E and A addresses are not equal, the lower one of the two must be specified.
CHANNEL	INPUT	INT	2	0	0	Channel number
STOP	INPUT	BOOL	4.4	TRUE/FALSE	FALSE	Stop run With STOP = TRUE you can stop/interrupt the run prematurely.
ERR_A	INPUT	BOOL	4.5	TRUE/FALSE	FALSE	Collect acknowledgment for external error External errors are cleared acknowledged with ERR_A = TRUE.
SPEED	INPUT	DINT	12	Creep speed up to 1,000,000 pulses/s No higher than the maximum speed declared in the parameter	1000	The axis is accelerated to the speed "$V_{Setpoint}$". It is not possible to change the speed during the run.
WORKING	OUTPUT	BOOL	16.0	TRUE/FALSE	FALSE	Run is in progress
ACT_POS	OUTPUT	DINT	18	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value
MODE_OUT	OUTPUT	INT	22	0, 1, 3, 4, 5	0	Active/configured operating mode

ERR	OUTPUT	WORD	24	Every bit "0" or "1":	0	External error: Bit 2: Zero point monitoring Bit 11: Travel range monitoring (always 1) Bit 12: operating range monitoring Bit 13: actual value monitoring Bit 14: Target home monitoring Bit 15: target range monitoring The remaining bits are reserved
ST_ENBLD	OUTPUT	BOOL	26.0	TRUE/ FALSE	TRUE	The CPU sets start enable if all the following conditions apply: • no STOP pending (STOP = FALSE) • no external error pending (ERR = 0) • drive enable is set (DRV_EN = TRUE • no positioning run active (WORKING = FALSE)
ERROR	OUTPUT	BOOL	26.1	TRUE/FALSE	FALSE	Error when starting /resuming a run
STATUS	OUTPUT	WORD	28.0	W#16#0000 to W#16#FFFF	W#16#0000	Error number

Parameters not assigned to the block (Statistical local data):

Parameter	Decla-ration	Data type	Address (Instance DB)	Range of values	Default	Description
ACCEL	STATIC	DINT	30	1 to 100,000 pulses/s^2	100	Acceleration Change during run not possible.
DECEL	STATIC	DINT	34	1 to 100,000 pulses/s^2	100	Deceleration Change during run not possible.
CHGDIFF_P	STATIC	DINT	38	0 to +10^8 pulses	1000	Changeover difference plus: "Changeover difference plus" defines the change-over point from which the drive continues its forward run with creep speed.

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
CUTOFF-DIFF_P	STATIC	DINT	42	0 to $+10^8$ pulses	100	Cut-off difference plus: "Cut-off difference plus" defines the cut-off point at which the drive forward run at creep speed is switched off.
CHGDIFF_M	STATIC	DINT	46	0 to $+10^8$ pulses	1000	Changeover difference minus: "Changeover difference minus" defines the changeover point from which the drive continues with a reverse run at creep speed.
CUTOFF-DIFF_P	STATIC	DINT	50	0 to $+10^8$ pulses	100	Cut-off difference minus: "Cut-off difference plus" defines the cut-off point at which the drive reverse run at creep speed is switched off.
PARA	STATIC	BOOL	54.0	TRUE/ FALSE	FALSE	Parameters have been assigned to the axis
DIR	STATIC	BOOL	54.1	TRUE/ FALSE	FALSE	Actual/last sense of direction FALSE = forward (in plus direction) TRUE = reverse (in minus direction)
CUTOFF	STATIC	BOOL	54.2	TRUE/ FALSE	FALSE	Drive in cut-off range (as off the cut-off point to the start of the next run)
CHGOVER	STATIC	BOOL	54.3	TRUE/ FALSE	FALSE	Drive in change-over range (between reaching creep speed and the start of the next run)
RAMP_DN	STATIC	BOOL	54.4	TRUE/ FALSE	FALSE	The drive is slowed down (from braking point to changeover point)
RAMP_UP	STATIC	BOOL	54.5	TRUE/ FALSE	FALSE	The drive is accelerated (from start until it reaches the speed SPEED ($V_{Setpoint}$))
DIST_TO_GO	STATIC	DINT	56	$-5x10^8$ to $+5x10^8$ pulses	0	Actual distance to go
LAST_TRG	STATIC	DINT	60	$-5x10^8$ to $+5x10^8$ pulses	0	Last/current target • Absolute Step Approach: At run start LST_TRG = current absolute target (TARGET). • Relative Step Approach: At run start LST_TRG = LAST_TRG is the specified +/- distance of the previous run (TARGET).

Parameters for "Jog" Mode

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
DRV_EN	INPUT	BOOL	4.0	TRUE/ FALSE	FALSE	Drive enable
DIR_P	INPUT	BOOL	4.2	TRUE/ FALSE	FALSE	Jogging in plus direction (positive edge)
DIR_M	INPUT	BOOL	4.3	TRUE/ FALSE	FALSE	Jogging in minus direction (positive edge)
MODE_IN	INPUT	INT	6	0, 1, 3, 4, 5	1	Operating mode, 1 = jogging
WORKING	OUTPUT	BOOL	16.0	TRUE/ FALSE	FALSE	Run is in progress
ACT_POS	OUTPUT	DINT	18	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value
MODE_OUT	OUTPUT	INT	22	0, 1, 3, 4, 5	0	Active/configured operating mode

Parameters for "Reference run" Mode

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
DRV_EN	INPUT	BOOL	4.0	TRUE/ FALSE	FALSE	Drive enable
DIR_P	INPUT	BOOL	4.2	TRUE/ FALSE	FALSE	Reference run in plus direction (positive edge)
DIR_M	INPUT	BOOL	4.3	TRUE/ FALSE	FALSE	Reference run in minus direction (positive edge)
MODE_IN	INPUT	INT	6	0, 1, 3, 4, 5	1	Operating mode, 3 = "Reference run"
WORKING	OUTPUT	BOOL	16.0	TRUE/ FALSE	FALSE	Run is in progress
SYNC	OUTPUT	BOOL	16.3	TRUE/ FALSE	FALSE	SYNC = TRUE: Axis is synchronized
ACT_POS	OUTPUT	DINT	18	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value
MODE_OUT	OUTPUT	INT	22	0, 1, 3, 4, 5	0	Active/configured operating mode

Parameters for "Relative Step Approach" Mode

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
DRV_EN	INPUT	BOOL	4.0	TRUE/ FALSE	FALSE	Drive enable
DIR_P	INPUT	BOOL	4.2	TRUE/ FALSE	FALSE	Run in plus direction (positive

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
						edge)
DIR_M	INPUT	BOOL	4.3	TRUE/ FALSE	FALSE	Run in minus direction (positive edge)
MODE_IN	INPUT	INT	6	0, 1, 3, 4, 5	1	Operating mode, 4 = Relative Step Approach
TARGET	INPUT	DINT	8	0 to 10^9 pulses	1000	Distance in pulses (only positive values allowed)
WORKING	OUTPUT	BOOL	16.0	TRUE/ FALSE	FALSE	Run is in progress
POS_RCD	OUTPUT	BOOL	16.1	TRUE/ FALSE	FALSE	Position reached
ACT_POS	OUTPUT	DINT	18	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value
MODE_OUT	OUTPUT	INT	22	0, 1, 3, 4, 5	0	Active/configured operating mode

Parameters for "Absolute Step Approach"

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
DRV_EN	INPUT	BOOL	4.0	TRUE/ FALSE	FALSE	Drive enable
START	INPUT	BOOL	4.1	TRUE/ FALSE	FALSE	Start run (positive edge)
DIR_P	INPUT	BOOL	4.2	TRUE/ FALSE	FALSE	Run in plus direction (positive edge)
DIR_M	INPUT	BOOL	4.3	TRUE/ FALSE	FALSE	Run in minus direction (positive edge)
MODE_IN	INPUT	INT	6	0, 1, 3, 4, 5	1	Operating mode, 5 = Absolute Step Approach
TARGET	INPUT	DINT	8	Linear axis $-5x10^8$ to $+5x10^8$ Rotary axis: 0 to rotary axis end -1	1000	Target in pulses
WORKING	OUTPUT	BOOL	16.0	TRUE/ FALSE	FALSE	Run is in progress
POS_RCD	OUTPUT	BOOL	16.1	TRUE/ FALSE	FALSE	Position reached
ACT_POS	OUTPUT	DINT	18	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value
MODE_OUT	OUTPUT	INT	22	0, 1, 3, 4, 5	0	Active/configured operating mode

Parameters For The Job "Set Reference Point"

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
SYNC	OUTPUT	BOOL	16.3	TRUE/ FALSE	FALSE	Axis is synchronized

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
JOB_REQ	STATIC	BOOL	76.0	TRUE/ FALSE	FALSE	Job initialization (positive edge)
JOB_DONE	STATIC	BOOL	76.1	TRUE/ FALSE	TRUE	New job can be started
JOB_ERR	STATIC	BOOL	76.2	TRUE/ FALSE	FALSE	Faulty job
JOB_ID	STATIC	INT	78	1, 2	0	Job, 1 = "Set Reference Point"
JOB_STAT	STATIC	WORD	80	W#16#0000 to W#16#FFFF	W#16#0000	Job error number
JOB_VAL	STATIC	DINT	82	$-5x10^8$ to $+5x10^8$ pulses	0	Job parameter for reference point coordinates

Parameters for the Job "Clear Remaining Distance"

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
JOB_REQ	STATIC	BOOL	76.0	TRUE/ FALSE	FALSE	Job initialization (positive edge)
JOB_DONE	STATIC	BOOL	76.1	TRUE/ FALSE	TRUE	New job can be started
JOB_ERR	STATIC	BOOL	76.2	TRUE/ FALSE	FALSE	Faulty job
JOB_ID	STATIC	INT	78	1, 2	0	Job, 2 = "Clear Remaining Distance"
JOB_STAT	STATIC	WORD	80	W#16#0000 to W#16#FFFF	W#16#0000	Job error number
JOB_VAL	STATIC	DINT	82	-	0	Any setting

Parameters for "Length Measurement" Operation

This operation is started at the positive edge on the digital input. There are no specific input parameters.

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
MSR_DONE	OUTPUT	BOOL	16.2	TRUE/ FALSE	FALSE	Length measurement completed

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
BEG_VAL	STATIC	DINT	64	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value Start length measurement
END_VAL	STATIC	DINT	68	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value Length measurement done
LEN_VAL	STATIC	DINT	72	0 to 10^9 pulses	0	Measured Length

Error Information

Operating mode error (ERROR = TRUE)

The output parameter ERROR is set TRUE if an error is detected. The parameter **STATUS** shows the cause of the error.

Event class Error code	Explanation
W#16#2002	Wrong SFB, use SFB 44
W#16#2004	Wrong channel number (CHANNEL). Set channel "0"
W#16#3001	Run job rejected because of job error in the same SFB call. Correct the respective JOB parameters
W#16#3002	A change of MODE_IN is not permitted while the drive is in operation. Wait for the end of the current positioning run.
W#16#3003	Unknown operating mode (MODE_IN). Permitted is 1 (jog), 3 (reference run), 4 (Relative Step Approach) and 5 (absolute Step Approach).
W#16#3004	Only one start request is allowed the same time. Valid start requests are DIR_P, DIR_M or START.
W#16#3005	START is only allowed in operating mode "Absolute Step Approach". Start the run with

Event class Error code	Explanation
	DIR_P or DIR_M
W#16#3006	DIR_P or DIR_M is not allowed for linear axis and in operating mode "Absolute Step Approach". Start the run with START
W#16#3007	Axis not synchronized. "Absolute Step Approach" is only possible synchronized axis.
W#16#3008	Clear working range. Return run to working position is only allowed in jog mode.
W#16#3101	No start enable because the axis is not parameterized. Parameterize the "Positioning" submodule via HWConfig
W#16#3102	Start not enabled because the drive is not enabled. Set "Enable Drive" on the SFB (DRV_EN=TRUE)
W#16#3103	Start not enabled because STOP is set. Clear the STOP on the SFB (STOP=FALSE)
W#16#3104	Start not enabled because the drive is currently performing a positioning run (WORKING=TRUE). Wait for the end of the current positioning run.
W#16#3105	Start not enabled because at least one pending error has not been cleared. First, eliminate and clear all external errors and the restart the run.
W#16#3202	Wrong speed setpoint in SPEED. The speed setpoint is out of the permitted range of the creep speed of up to 1000000 pulses/s, though not higher than the parameterized maximum speed.
W#16#3203	The acceleration setpoint in ACCEL is out of the range of 1 to 100,000 pulses/s^2.
W#16#3204	The deceleration setpoint in DECEL is out of the range of 1 to 100,000 pulses/s^2.
W#16#3206	The speed setpoint in SPEED must be higher than / equal to the parameterized referencing frequency.
W#16#3301	Changeover/cut-off difference is too high. Set a maximum changeover/cut-off difference of 10^8
W#16#3304	Cut-off difference too low. The cut-off difference must be at least half the size of the target range.
W#16#3305	Changeover difference too low. The changeover difference must be at least half the size of the target range.
W#16#3401	Target setting out of working range. For a linear axis and Step Approach the target setting must be within the range of the software limit switches (inclusive).
W#16#3402	Wrong target setting. For a rotary axis the target setting must be greater than 0 and lower than the rotary axis end value.
W#16#3403	Wrong distance setting. The travel distance setpoint for the Relative Step Approach must be positive.
W#16#3404	Wrong distance setting. The result, the absolute target coordinate, must be greater than $-5x10^8$.
W#16#3405	Wrong distance setting. The result, the absolute target coordinate, must be lower than $5x10^8$.
W#16#3406	Wrong distance setting. The result, the absolute target coordinate, must lie within the working range (+/- half of the target range)
W#16#3501	Travel distance too long. Target coordinate + actual remaining distance must be greater than / equal to $-5x10^8$
W#16#3502	Travel distance too long. Target coordinate + actual remaining distance must be smaller than / equal to $5x10^8$
W#16#3503	Travel distance too short. The travel distance in plus direction must be greater than the specified cut-off difference in plus direction

Event class Error code	Explanation
W#16#3504	Travel distance too short. The travel distance in minus direction must be greater than the specified cut-off difference in minus direction
W#16#3505	Travel distance too short or the limit switch in plus direction is already actuated. The last approachable target in plus direction (working range or travel distance limit) is too close to the actual position
W#16#3506	Travel distance too short or the limit switch in minus direction is already actuated. The last approachable target in minus direction (working range or travel distance limit) is too close to the actual position

Job Error (JOB_ERR = TRUE)

The output parameter JOB_ERROR is set TRUE if an error is detected. The parameter JOB_STAT shows the cause of the error.

Event class Error code	Explanation
W#16#4001	Axis not parameterized. Parameterize the "Positioning" submodule via HWConfig
W#16#4002	Job not possible because positioning is still in progress. Wait until WORKING = FALSE, then repeat the job.
W#16#4004	Unknown job. Check the job ID and then repeat the job.
W#16#4101	For a linear axis the reference point coordinate must not be out of range of the working range limits.
W#16#4102	For a linear axis the set reference point coordinate + actual remaining distance must be greater than / equal to $-5x10^8$.
W#16#4103	For a linear axis the set reference point coordinate + actual remaining distance must be smaller than / equal to $5x10^8$.
W#16#4104	For a linear axis the set reference point coordinate + actual remaining difference to the starting point must be greater than / equal to $-5x10^8$.
W#16#4105	For a linear axis the set reference point coordinate + actual remaining difference to the starting point must be smaller than / equal to $-5x10^8$.
W#16#4106	For a rotary axis the reference point coordinate must not be lower than 0 and greater than / equal to the rotary axis end.

External Error (ERR)

The technical circuit monitors the run, travel distance and the connected peripheral devices. Prerequisite is that you have switched on monitoring in the "Drive", "Axis" and "Encoder" parameter screen forms.

An external fault is reported when the monitoring unit is triggered. External errors can occur independent on the started operations. You must always clear external errors with ERR_A = TRUE.

A set bit in the SFB parameter ERR (WORD) the external errors.

Monitoring	Error code	Bit in ERR-WORD
Zero pulse (zero mark)	W#16#0004	2
Travel distance	W#16#0800	11
Working range	W#16#1000	12
Actual value	W#16#2000	13
Target position	W#16#4000	14
Target range	W#16#8000	15

System Error

A system error is indicated with BIE = FALSE. A system error is caused by errors while reading/writing the instance DB or by a multiple call of the SFB.

25.2 Positioning with Digital Output Using SFB 46 "DIGITAL"

Description

Use **SFB DIGITAL (SFB 46)** to control the positioning functions via user program.

The four 24-V digital outputs are assigned fixed to drive. They control the power stage. Dependent on the control mode configuration, the digital outputs control the direction and speed stages (rapid/creep speed).

The distance is measured via an asymmetrical 24-V incremental transducer with two phases offset at 90 degrees.

- First, the target is approached with the speed (**V_{Rapid}**).
- At the **changeover point** the speed is toggled to creep speed **(V_{Creep})**.
- The drive is switched off at the **switch-off point**.
- The switch-over point and the switch-off point are determined for every Step Approach by the parameter values you have declared for **changeover difference** and **cut-off difference**. The changeover difference and cut-off difference can be determined differently for the forward motion (in plus direction) and for the reverse motion (in minus direction).
- The run is completed (**WORKING** = FALSE) when the cut-off point is reached. A new run can then be started.
- The specified target is reached (**POS_RCD** = TRUE) when the actual position value has reached the **home target**. If the actual position value drifts off without a new run having been started the signal "Position reached" is not reset again.

Basic Parameters:

Here we describe the SFB parameters. They apply to all operating modes. The parameters specific to the operating mode are described with the individual operating modes.

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
LADDR	INPUT	WORD	0	CPU specific	W#16#0310	The I/O address of your submodule, specified by you in "HW Config". If the E and A addresses are not equal, the lower one of the two must be specified.
CHANNEL	INPUT	INT	2	0	0	Channel number:
STOP	INPUT	BOOL	4.4	TRUE/ FALSE	FALSE	Stop run With STOP = TRUE you can stop/interrupt the run prematurely.
ERR_A	INPUT	BOOL	4.5	TRUE/ FALSE	FALSE	Collect acknowledgment for

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
						external error External errors are cleared acknowledged with ERR_A = TRUE.
SPEED	INPUT	BOOL	12.0	TRUE/ FALSE	FALSE	Two speed stages for Fast/Creep mode TRUE = Rapid mode FALSE = Creep mode
WORKING	OUTPUT	BOOL	14.0	TRUE/ FALSE	FALSE	Run is in progress
ACT_POS	OUTPUT	DINT	16	$-5x10^8$ to $5x10^8$ pulses	0	Actual position value
MODE_OUT	OUTPUT	INT	20	0, 1, 3, 4, 5	0	Active/configured operating mode
ERR	OUTPUT	WORD	22	Every bit "0" or "1":	0	External error: Bit2: Zero mark monitoring Bit11: Travel range monitoring (always 1) Bit12: Operating range monitoring Bit12: Actual value monitoring Bit12: Target position monitoring Bit15: Target position monitoring. The remaining bits are reserved
ST_ENBLD	OUTPUT	BOOL	24.0	TRUE/ FALSE	TRUE	The CPU enables the if all the following conditions apply: • no STOP pending (STOP = FALSE) • no external error pending (ERR = 0) • drive enable is set (DRV_EN = TRUE • no positioning run active (WORKING = FALSE)
ERROR	OUTPUT	BOOL	24.1	TRUE/ FALSE	FALSE	Error when starting /resuming a run
STATUS	OUTPUT	WORD	26.0	W#16#0000 to W#16#FFFF	W#16#0000	Error number

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
CHGDIFF_P	STATIC	DINT	28	0 to $+10^8$ pulses	1000	Switch-over difference plus: The "Changeover difference plus" defines the point at which the drive continues its forward run with creep speed.
CUTOFF-DIFF_P	STATIC	DINT	32	0 to $+10^8$ pulses	100	Cut-off difference plus: The "Cut-off difference plus" defines the cut-off point at which the drive forward run with creep speed is switched off.
CHGDIFF_M	STATIC	DINT	36	0 to $+10^8$ pulses	1000	Changeover difference minus: The "Changeover difference minus" defines the point at which the drive continues its forward run with creep speed.
CUTOFF-DIFF_P	STATIC	DINT	40	0 to $+10^8$ pulses	100	Cut-off difference minus: "Cut-off difference plus" defines the point at which the drive reverse run with crawl speed is switched off.
PARA	STATIC	BOOL	44.0	TRUE/ FALSE	FALSE	Parameters have been assigned to the axis
DIR	STATIC	BOOL	44.1	TRUE/ FALSE	FALSE	Actual/last sense of direction FALSE = forward (in plus direction) TRUE = reverse (in minus direction)
CUTOFF	STATIC	BOOL	44.2	TRUE/ FALSE	FALSE	Drive in cut-off range (from cut-off position to the start of the next run)
CHGOVER	STATIC	BOOL	44.3	TRUE/ FALSE	FALSE	Drive in changeover range (from the point where it reaches creep speed to the start of the next run)

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
DIST_TO_GO	STATIC	DINT	46	$-5x10^8$ to $+5x10^8$ pulses	0	Actual remaining distance
LAST_TRG	STATIC	DINT	50	$-5x10^8$ to $+5x10^8$ pulses	0	Last/current target • Absolute Step Approach: At run start LST_TRG = current absolute target (TARGET). • Relative Step Approach: At run start LST_TRG = LAST_TRG is the specified +/- distance of the previous run (TARGET).

Parameters for "Jog" Mode

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
DRV_EN	INPUT	BOOL	4.0	TRUE/ FALSE	FALSE	Drive enable
DIR_P	INPUT	BOOL	4.2	TRUE/ FALSE	FALSE	Jogging in plus direction (positive edge)
DIR_M	INPUT	BOOL	4.3	TRUE/ FALSE	FALSE	Jogging in minus direction (positive edge)
MODE_IN	INPUT	INT	6	0, 1, 3, 4, 5	1	Operating mode, 1 = jogging
WORKING	OUTPUT	BOOL	14.0	TRUE/ FALSE	FALSE	Run is in progress
ACT_POS	OUTPUT	DINT	16	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value
MODE_OUT	OUTPUT	INT	20	0, 1, 3, 4, 5	0	Active/configured operating mode

Parameters for "Reference run" Mode

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
DRV_EN	INPUT	BOOL	4.0	TRUE/ FALSE	FALSE	Drive enable
DIR_P	INPUT	BOOL	4.2	TRUE/ FALSE	FALSE	Reference run in plus direction (positive edge)
DIR_M	INPUT	BOOL	4.3	TRUE/ FALSE	FALSE	Reference run in minus direction (positive edge)
MODE_IN	INPUT	INT	6	0, 1, 3, 4, 5	1	Operating mode, 3 = "Reference run"
WORKING	OUTPUT	BOOL	14.0	TRUE/ FALSE	FALSE	Run is in progress
SYNC	OUTPUT	BOOL	14.3	TRUE/ FALSE	FALSE	SYNC = TRUE: Axis is synchronized
ACT_POS	OUTPUT	DINT	16	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value
MODE_ OUT	OUTPUT	INT	20	0, 1, 3, 4, 5	0	Active/configured operating mode

Parameters for "Relative Step Approach" Mode

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
DRV_EN	INPUT	BOOL	4.0	TRUE/ FALSE	FALSE	Drive enable
DIR_P	INPUT	BOOL	4.2	TRUE/ FALSE	FALSE	Run in plus direction (positive edge)
DIR_M	INPUT	BOOL	4.3	TRUE/ FALSE	FALSE	Run in minus direction (positive edge)
MODE_IN	INPUT	INT	6	0, 1, 3, 4, 5	1	Operating mode, 4 = Relative Step Approach
TARGET	INPUT	DINT	8	0 to 10^9 pulses	1,000	Distance in pulses (only positive values allowed)
WORKING	OUTPUT	BOOL	14.0	TRUE/ FALSE	FALSE	Run is in progress
POS_RCD	OUTPUT	BOOL	14.1	TRUE/ FALSE	FALSE	Position reached
ACT_POS	OUTPUT	DINT	16	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value
MODE_ OUT	OUTPUT	INT	20	0, 1, 3, 4, 5	0	Active/configured operating mode

Parameters for " Absolute Step Approach "

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
DRV_EN	INPUT	BOOL	4.0	TRUE/ FALSE	FALSE	Drive enable
START	INPUT	BOOL	4.1	TRUE/ FALSE	FALSE	Start run (positive edge)
DIR_P	INPUT	BOOL	4.2	TRUE/ FALSE	FALSE	Run in plus direction (positive edge)
DIR_M	INPUT	BOOL	4.3	TRUE/ FALSE	FALSE	Run in minus direction (positive edge)
MODE_IN	INPUT	INT	6	0, 1, 3, 4, 5	1	Operating mode, 5 = Absolute Step Approach
TARGET	INPUT	DINT	8	Linear axis $-5x10^8$ to $+5x10^8$ Rotary axis: 0 to rotary axis end -1	1,000	Target in pulses
WORKING	OUTPUT	BOOL	14.0	TRUE/ FALSE	FALSE	Run is in progress
POS_RCD	OUTPUT	BOOL	14.1	TRUE/ FALSE	FALSE	Position reached
ACT_POS	OUTPUT	DINT	16	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value
MODE_ OUT	OUTPUT	INT	20	0, 1, 3, 4, 5	0	Active/configured operating mode

Parameters for The Job "Set Reference Point"

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
SYNC	OUTPUT	BOOL	14.3	TRUE/ FALSE	FALSE	Axis is synchronized

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
JOB_REQ	STATIC	BOOL	66.0	TRUE/ FALSE	FALSE	Job initialization (positive edge)
JOB_DONE	STATIC	BOOL	66.1	TRUE/ FALSE	TRUE	New job can be started
JOB_ERR	STATIC	BOOL	66.2	TRUE/ FALSE	FALSE	Faulty job
JOB_ID	STATIC	INT	68	1, 2	0	Job, 1 = "Set Reference Point"
JOB_STAT	STATIC	WORD	70	W#16#0000 to W#16#FFFF	W#16#0000	Job error number
JOB_VAL	STATIC	DINT	72	$-5x10^8$ to $+5x10^8$ pulses	0	Job parameter for reference point coordinates

Parameters for The Job "Clear Remaining Distance"

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
JOB_REQ	STATIC	BOOL	66.0	TRUE/ FALSE	FALSE	Job initialization (positive edge)
JOB_DONE	STATIC	BOOL	66.1	TRUE/ FALSE	TRUE	New job can be started
JOB_ERR	STATIC	BOOL	66.2	TRUE/ FALSE	FALSE	Faulty job
JOB_ID	STATIC	INT	68	1, 2	0	Job, 2 = "Clear Remaining Distance"
JOB_STAT	STATIC	WORD	70	0 to FFFF hex	0	Job error number
JOB_VAL	STATIC	DINT	72	-	0	None.

Parameters for the "Length Measurement" Function

This operation is started at the positive edge on the digital input. There are no specific input parameters.

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
MSR_DONE	OUTPUT	BOOL	14.2	TRUE/ FALSE	FALSE	Length measurement completed

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
BEG_VAL	STATIC	DINT	54	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value Start of length measurement
END_VAL	STATIC	DINT	58	$-5x10^8$ to $+5x10^8$ pulses	0	Actual position value Length measurement done
LEN_VAL	STATIC	DINT	62	0 to 10^9 pulses	0	Measured Length

Error Information

Operating mode error (ERROR = TRUE)

The output parameter ERROR is set TRUE if an error is detected. The parameter **STATUS** shows the cause of the error.

Event class Error code	Explanation
W#16#2001	Wrong SFB, use SFB 46
W#16#2004	Wrong channel number (CHANNEL). Set channel "0"
W#16#3001	Run job rejected because of job error in the same SFB call. Correct the respective JOB parameters
W#16#3002	A change of MODE_IN is not permitted while the drive is in operation. Wait for the end of the current positioning run.
W#16#3003	Unknown operating mode (MODE_IN). Permitted is 1 (jog), 3 (reference run), 4 (Relative Step Approach) and 5 (Absolute Step Approach).
W#16#3004	Only one start request is allowed the same time. Valid start requests are DIR_P, DIR_M or START.
W#16#3005	START is only allowed in operating mode “ Absolute Step Approach ”. Start the run with DIR_P or DIR_M
W#16#3006	DIR_P or DIR_M is not allowed for linear axis and in operating mode "Absolute Step

Event class Error code	Explanation
	Approach". Start the run with START
W#16#3007	Axis not synchronized. "Absolute Step Approach" is only possible with a synchronized axis.
W#16#3008	Clear working range. Return run to working position is only allowed in jog mode.
W#16#3101	Start is not enabled because the axis is not parameterized. Parameterize the "Positioning" submodule via HWConfig
W#16#3102	Start not enabled because the drive is not enabled. Set "Enable Drive" on the SFB (DRV_EN=TRUE)
W#16#3103	Start not enabled because STOP is set. Clear the STOP on the SFB (STOP=FALSE)
W#16#3104	Start is not enabled because the drive is currently performing a positioning run (WORKING=TRUE). Wait for the end of the current positioning run.
W#16#3105	Start not enabled because at least one error that is pending has not been cleared. First, eliminate and clear all external errors and the restart the run.
W#16#3201	Wrong speed setpoint SPEED. For positioning with digital outputs only "Creep speed" (0) and "Rapid speed" (1) are allowed.
W#16#3301	Changeover/cut-off difference is too high. Set a maximum changeover/cut-off difference of 10^8
W#16#3303	Changeover difference too low. The changeover difference must be higher than / equal to the cut-off difference.
W#16#3304	Cut-off difference too low. The cut-off difference must be at least half the size of the target range.
W#16#3401	Target setting out of working range. For a linear axis and Step Approach the target setting must be within the range of the software limit switches (inclusive).
W#16#3402	Wrong target setting. For a rotary axis the target setting must be greater than 0 and lower than the rotary axis end value.
W#16#3403	Wrong distance setting. The travel distance setpoint for the Relative Step Approach must be positive.

Event class Error code	Explanation
W#16#3404	Wrong distance setting. The result, the absolute target coordinate, must be greater than -5x10^8.
W#16#3405	Wrong distance setting. The result, the absolute target coordinate, must be lower than 5x10^8.
W#16#3406	Wrong distance setting. The result, the absolute target coordinate, must lie within the working range (+/- half of the target range)
W#16#3501	Travel distance too long. Target coordinate + actual remaining distance must be greater than / equal to -5x10^8
W#16#3502	Travel distance too long. Target coordinate + actual remaining distance must be smaller than / equal to 5x10^8
W#16#3503	Travel distance too short. The travel distance in plus direction must be greater than the specified cut-off difference towards plus
W#16#3504	Travel distance too short. The travel distance in minus direction must be greater than the specified cut-off difference towards minus
W#16#3505	Travel distance too short or the limit switch in plus direction is already actuated. The last approachable target in plus direction (working range or travel distance limit) is too close to the actual position.
W#16#3506	Travel distance too short or the limit switch in minus direction is already actuated. The last approachable target in minus direction (working range or travel distance limit) is too close to the actual position.

Job Error (JOB_ERR = TRUE)

The output parameter JOB_ERROR is set TRUE if an error is detected. The parameter JOB_STAT shows the cause of the error.

Event class Error code	Explanation
W#16#4001	Axis not parameterized. Parameterize the "Positioning" submodule via HWConfig
W#16#4002	Job not possible because positioning is in progress. Jobs can only be carried out if no positioning run is in progress. Wait until WORKING = FALSE, then repeat the job.
W#16#4004	Unknown job. Check the job ID and then repeat the job.
W#16#4101	For a linear axis the reference point coordinate must not be out of range of the working range limits.
W#16#4102	For a linear axis the specified reference point coordinate + actual remaining distance must be greater than / equal to -5x10^8.
W#16#4103	For a linear axis the specified reference point coordinate + actual remaining distance must be smaller than / equal to 5x10^8.
W#16#4104	For a linear axis the specified reference point coordinate + actual remaining difference to the starting point must be greater than / equal to -5x10^8.
W#16#4105	For a linear axis the specified reference point coordinate + actual remaining difference to the starting point of the run must be smaller than / equal to -5x10^8.
W#16#4106	For a rotary axis the reference point coordinate must not be lower than 0 and greater than / equal to the rotary axis end.

External Error (ERR)

The technical circuit monitors the run, travel distance and the connected peripheral devices. Prerequisite is that you have switched on monitoring in the "Drive", "Axis" and "Encoder" parameter screen forms.

An external fault is reported when the monitoring unit is triggered. External errors can occur independent on the started operations. You must always clear external errors with ERR_A = TRUE.

A set bit in the SFB parameter ERR (WORD) the external errors.

Monitoring	Error code	Bit in ERR-WORD
Zero pulse (zero mark)	W#16#0004	2
Travel distance	W#16#0800	11
Working range	W#16#1000	12
Actual value	W#16#2000	13
Target approach	W#16#4000	14
Target range	W#16#8000	15

System Error

A system error is indicated with BIE = FALSE. A system error is caused by errors when reading/writing the instance DB or by a multiple call of the SFB.

25.3 Controlling the Counter with SFB 47 "COUNT"

Description

To control the positioning functions via the user program, use **SFB COUNT (SFB 47)**.

The following operations are available:

- Starting/stopping the counter via software gate **SW_GATE**
- Enabling/controlling the output DO
- Retrieving the status bits **STS_CMP**, **STS_OFLW**, **STS_UFLW** and **STS_ZP**
- Retrieving the actual counter value **COUNTVAL**
- Jobs for reading/writing the internal counter registers

Parameter

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
LADDR	INPUT	WORD	0	CPU specific	W#16#0300	The I/O address of your submodule, specified by you in "HW Config". If the E and A addresses are not equal, the lower one of the two must be specified.
CHANNEL	INPUT	INT	2	CPU 312C: 0 to 1 CPU 313C: 0 to 2 CPU 314C: 0 to 3	0	Channel number:
SW_GATE	INPUT	BOOL	4.0	TRUE/FALSE	FALSE	Software gate for starting/stopping the counter
CTRL_DO	INPUT	BOOL	4.1	TRUE/FALSE	FALSE	Enable output
SET_DO	INPUT	BOOL	4.2	TRUE/FALSE	FALSE	Control output
JOB_REQ	INPUT	BOOL	4.3	TRUE/FALSE	FALSE	Job initialization (positive edge)

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
JOB_ID	INPUT	WORD	6	W#16#0000 Job without function W#16#0001 Write count value W#16#0002 Write load value W#16#0004 Write comparison value W#16#0008 Write hysteresis W#16#0010 Write pulse period W#16#0082 Read load value W#16#0084 Read comparison value W#16#0088 Read hysteresis W#16#0090 Read pulse period	W#16#0000	Job number
JOB_VAL	INPUT	DINT	8	-2^{31} up to $+2^{31}-1$	0	Value for write jobs.
STS_GATE	OUTPUT	BOOL	12.0	TRUE/FALSE	FALSE	Status of the internal gate
STS_STRT	OUTPUT	BOOL	12.1	TRUE/FALSE	FALSE	Status of the hardware gate (Start input)
STS_LTCH	OUTPUT	BOOL	12.2	TRUE/FALSE	FALSE	Status of the latch input
STS_DO	OUTPUT	BOOL	12.3	TRUE/FALSE	FALSE	Output status
STS_C_DN	OUTPUT	BOOL	12.4	TRUE/FALSE	FALSE	Status reverse direction. Displayed is always the last direction of count. The value of STS_C_DN is FALSE after the first call of the SFB.

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
STS_C_UP	OUTPUT	BOOL	12.5	TRUE/FALSE	FALSE	Status forward direction Displayed is always the last direction of count. The value of STS_C_DN is TRUE after the first call of the SFB.
COUNTVAL	OUTPUT	DINT	14	-2^{31} up to $+2^{31}-1$	0	Actual count value
LATCHVAL	OUTPUT	DINT	18	-2^{31} up to $+2^{31}-1$	0	Actual latch value
JOB_DONE	OUTPUT	BOOL	22.0	TRUE/FALSE	TRUE	New job can be started
JOB_ERR	OUTPUT	BOOL	22.1	TRUE/FALSE	FALSE	Faulty job
JOB_STAT	OUTPUT	WORD	24	0 to W#16#FFFF	0	Job error number

Note

If you have set the parameter "Reaction of the output" to "No comparison" via the configuration interface, the following is valid:

- The output will be switched in the same way as a normal output.
- The input parameters CTRL_DO and SET_DO of the SFB are not active.
- The status bit STS_DO and STS_CMP (Status comparator in the IDB) remain reset.

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
STS_CMP	STATIC	BOOL	26.3	TRUE/FALSE	FALSE	Status comparator. Reset with RES_STS. The status bit STS_CMP indicates that the conditions for comparison for the comparator is met or has been met. STS_CMP also indicates that the output was set (STS_DO = TRUE)
STS_OFLW	STATIC	BOOL	26.5	TRUE/FALSE	FALSE	Status overflow Reset with RES_STS.
STS_UFLW	STATIC	BOOL	26.6	TRUE/FALSE	FALSE	Status underflow Reset with RES_STS.

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
STS_ZP	STATIC	BOOL	26.7	TRUE/FALSE	FALSE	Status zero crossing Reset with RES_STS. Only set for counters without master count direction. Indicates the zero crossing. Is also set when the counter is set to 0 or if the counter starts counting as of load value=0.
JOB_OVAL	STATIC	DINT	28	-2^{31} up to $+2^{31}-1$	0	Output value for read jobs.
RES_STS	STATIC	BOOL	32.2	TRUE/FALSE	FALSE	Reset status bits. Resets the status bits STS_CMP, STS_OFLW, STS_UFLW and STS_ZP. Two calls of the SFB are required to reset the status bits.

Error Information

Job Error

JOB_ERR = TRUE is set if a job error occurs. The precise error cause is displayed in **JOB_STAT**.

For information on the valid values for individual parameters, please refer to the user manual.

Event class Error code	Explanation
W#16#0121	Compare value too low.
W#16#0122	Compare value too high.
W#16#0131	Hysteresis too small.
W#16#0132	Hysteresis too wide.
W#16#0141	Pulse period too low.
W#16#0142	Pulse period too high.
W#16#0151	Load value too low.
W#16#0152	Load value too high.
W#16#0161	Counter value too low.
W#16#0162	Counter value too high.
W#16#01FF	Illegal job number.

System Error

BIE = False is set after a system error occurs.

Event class Error code	Explanation
W#16#8001	Wrong operating mode or faulty parameters. Set the correct operating mode in "Configure Hardware" or use an SFB that matches the set operating mode
W#16#8009	Illegal channel number. Set a channel number ≤ 3 (CPU specific value).

25.4 Controlling the Frequency Measurement with SFB 48 "FREQUENCY"

Description

You can operate the frequency counter via user program. In this case you must use **SFB FREQUENC (SFB48)**.

The following operations are available:

- Starting/stopping the via software gate **SW_GATE**
- Enabling/controlling the output DO
- Retrieving the status bits **STS_CMP**, **STS_OFLW** and **STS_UFLW**
- Retrieving the actual frequency value **MEAS_VAL**
- Jobs for reading/writing the internal frequency counter registers

Parameter

Parameter	Decla-ration	Data type	Address (Instance DB)	Range of values	Default	Description
LADDR	INPUT	WORD	0	CPU specific	W#16#0300	The I/O address of your submodule, specified by you in "HW Config". If the I and O addresses are not equal, the lower one of the two must be specified.
CHANNEL	INPUT	INT	2	CPU 312C: 0 to 1 CPU 313C: 0 to 2 CPU 314C: 0 to 3	0	Channel number:
SW_GATE	INPUT	BOOL	4.0	TRUE/FALSE	FALSE	Software gate for starting/stopping the frequency measurement

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
MAN_DO	INPUT	BOOL	4.1	TRUE/FALSE	FALSE	Enable manual output control
SET_DO	INPUT	BOOL	4.2	TRUE/FALSE	FALSE	Control output
JOB_REQ	INPUT	BOOL	4.3	TRUE/FALSE	FALSE	Job initialization (positive edge)
JOB_ID	INPUT	WORD	6	W#16#0000 = Job without function W#16#0001 = Write low limit W#16#0001 = Write upper limit W#16#0004 = Write integration time W#16#0081 = Read low limit W#16#0081 = Read upper limit W#16#0084 = Read integration time	0	Job number
JOB_VAL	INPUT	DINT	8	-2^{31} to $+2^{31}-1$	0	Value for write jobs.
STS_GATE	OUTPUT	BOOL	12.0	TRUE/FALSE	FALSE	Status of the internal gate
STS_STRT	OUTPUT	BOOL	12.1	TRUE/FALSE	FALSE	Status of the hardware gate (Start input)
STS_DO	OUTPUT	BOOL	12.2	TRUE/FALSE	FALSE	Output status
STS_C_DN	OUTPUT	BOOL	12.3	TRUE/FALSE	FALSE	Status reverse direction. Displayed is always the last direction of count. The value of STS_C_DN is FALSE after the first call of the SFB.
STS_C_UP	OUTPUT	BOOL	12.4	TRUE/FALSE	FALSE	Status forward direction Displayed is always the last direction of count. The value of STS_C_UP is TRUE after the first call of the SFB.
MEAS_VAL	OUTPUT	DINT	14	0 to $+2^{31}-1$	0	Actual frequency value
COUNTVAL	OUTPUT	DINT	18	-2^{31} to $+2^{31}-1$	0	Actual count value (starts every time the gate opens at 0)

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
JOB_DONE	OUTPUT	BOOL	22.0	TRUE/FALSE	TRUE	New job can be started
JOB_ERR	OUTPUT	BOOL	22.1	TRUE/FALSE	FALSE	Faulty job
JOB_STAT	OUTPUT	WORD	24	W#16#0000 to W#16#FFFF	W#16#0000	Job error number

Note

If you have set the parameter "Reaction of the output" to "No comparison" via the configuration interface, the following is valid:

- The output will be switched in the same way as a normal output.
- The SFB input parameters MAN_DO and SET_DO are not active.
- The status bit STS_DO remains reset.

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
STS_CMP	STATIC	BOOL	26.3	TRUE/FALSE	FALSE	Status end of measurement. Reset with RES_STS. The measured value is updated after every expired time interval. Here, the end of measurement is reported by the status bit STS_CMP
STS_OFLW	STATIC	BOOL	26.5	TRUE/FALSE	FALSE	Status overflow. Reset with RES_STS.
STS_UFLW	STATIC	BOOL	26.6	TRUE/FALSE	FALSE	Status underflow Reset with RES_STS.
JOB_OVAL	STATIC	DINT	28	-2^{31} up to $2^{31}-1$	0	Output value for read jobs.
RES_STS	STATIC	BOOL	32.2	TRUE/FALSE	FALSE	Reset status bits. Resets the status bits STS_CMP, STS_OFLW, STS_UFLW. Two calls of the SFB are required to reset the status bits.

Job Error

JOB_ERR = TRUE if a job error occurs. The precise error cause is displayed in **JOB_STAT**.

For information on the valid values for individual parameters, please refer to the user manual.

Event class Error code	Explanation
W#16#0221	Integration time too low.
W#16#0222	Integration time too high.
W#16#0231	Lower limit of the frequency is too low.
W#16#0232	Upper limit of the frequency is too high.
W#16#0241	Upper limit of the frequency is too low.
W#16#0242	Upper limit of the frequency is too high.
W#16#02FF	Illegal job number.

System Error

BIE = False is set after a system error occurs.

Event class Error code	Explanation
W#16#8001	Wrong operating mode or faulty parameters. Set the correct operating mode in "Configure Hardware" or use an SFB that matches the set operating mode
W#16#8009	Illegal channel number. Set a channel number ≤ 3 (CPU specific value).

25.5 Controlling Pulse Width Modulation with SFB 49 "PULSE"

Description

To control pulse width modulation via the user program, use **SFB ANALOG (SFB 49)**.

The following operations are available:

- Starting/stopping via software gate **SW_EN**
- Enabling/controlling the output DO
- Retrieving the status bits **STS_EN**, **STS_STRT** and **STS_DO**
- Input of the output value
- Jobs for reading/writing the registers

Parameter

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Meaning
LADDR	INPUT	WORD	0	CPU specific	W#16#0300	The I/O address of your submodule, specified by you in "HW Config". If the E and A addresses are not equal, the lower one of the two must be specified.
CHANNEL	INPUT	INT	2	CPU 312C: 0 to 1 CPU 313C: 0 to 2 CPU 314C: 0 to 3	0	Channel number:
SW_EN	INPUT	BOOL	4.0	TRUE/FALSE	FALSE	Software gate for starting/stopping the output
MAN_DO	INPUT	BOOL	4.1	TRUE/FALSE	FALSE	Enable manual output control
SET_DO	INPUT	BOOL	4.2	TRUE/FALSE	FALSE	Control output
OUTP_VAL	INPUT	INT	6.0	in ppm: 0 to 1,000 as S7 analog value: 0 to 27,648	0	Default output value if you enter an output value > 1 000 or 27648 the CPU limits it to 1,000 or 27,648
JOB_REQ	INPUT	BOOL	8.0	TRUE/FALSE	FALSE	Job initialization (positive edge)

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Meaning
JOB_ID	INPUT	WORD	10	W#16#0000 = Job without function W#16#0001 = Write period time W#16#0001 = Write On delay W#16#0004 = Write minimum pulse period W#16#0081 = Read period time W#16#0081 = Read On delay W#16#0084 = Read minimum pulse period	W#16#0000	Job number
JOB_VAL	INPUT	DINT	12	-2^{31} to $+2^{31}-1$	0	Value for write jobs.
STS_EN	OUTPUT	BOOL	16.0	TRUE/FALSE	FALSE	Enable status
STS_STRT	OUTPUT	BOOL	16.1	TRUE/FALSE	FALSE	Status of the hardware gate (Start input)
STS_DO	OUTPUT	BOOL	16.2	TRUE/FALSE	FALSE	Output status
JOB_DONE	OUTPUT	BOOL	16.3	TRUE/FALSE	TRUE	New job can be started
JOB_ERR	OUTPUT	BOOL	16.4	TRUE/FALSE	FALSE	Faulty job
JOB_STAT	OUTPUT	WORD	18	W#16#0000 to W#16#FFFF	W#16#0000	Job error number

Parameters not assigned to the block (Statistical local data):

Parameter	Declaration	Data type	Address (Instance DB)	Range of values	Default	Description
JOB_OVAL	OUTPUT	DINT	20	-2^{31} up to $2^{31}-1$	0	Output value for read jobs.

Job Error

JOB_ERR = TRUE if a job error occurs. The precise error cause is displayed in **JOB_STAT**.

For information on the valid values for individual parameters, please refer to the user manual.

Event class Error code	Explanation
W#16#0411	Period too low.
W#16#0412	Period too long.
W#16#0421	On delay too short.
W#16#0422	On delay too long.
W#16#0431	Minimum pulse period too low.
W#16#0432	Minimum pulse period too high.
W#16#04FF	Illegal job number.

System Error

BIE = False is set after a system error occurs.

Event class Error code	Explanation
W#16#8001	Wrong operating mode or faulty parameters. Set the correct operating mode in "Configure Hardware" or use an SFB that matches the set operating mode
W#16#8009	Illegal channel number. Set a channel number ≤ 3 (CPU specific value).

25.6 Sending Data (ASCII, 3964(R)) with SFB 60 "SEND_PTP"

Description

You can transmit a data block from a DB via **SFB SEND_PTP (SFB 60)**.

The send operation is executed after the block is called and a positive edge on control input **REQ**.

The range of data to be transmitted is determined in **SD_1** (DB number and start address). The length of the data block is determined in **LEN**.

To enable the SFB to process the job, you must call it with **R**(Reset)=FALSE. At the positive edge on control input R the current transmission is aborted and the SFB is reset to basic state. An aborted job is terminated with an error message (STATUS output).

For your submodule, you declare the I/O address, which you specified in "HW Config", in **LADDR**.

DONE is set TRUE if the job was terminated without error, or **ERROR** is set TRUE if the job was terminated with an error.

If the job was processed with DONE=TRUE this means that:

- When using the ASCII driver: The data where transmitted to the communication partner. It is not ensured that all data has been received by the communication partner.
- When using the procedure 3964(R): The data have been transmitted to the communication partner and they where acknowledged positively by the partner. It is not ensured that the data were passed on the partner CPU.

In **STATUS** the CPU indicates an error or, as a result of a warning, the respective event ID.

DONE or ERROR/STATUS are also output when the SFB is RESET (R=TRUE).

The binary result BIE is reset if an error has occurred. The status of the binary result is TRUE if the block was terminated without error.

Note

A parameter check is not included in the SFB. The CPU might go into STOP mode if the parameterization is faulty.

Instance DB

The SFB SEND_PTP operates in combination with an instance DB. The DB number is passed on with the call. Accessing data in the instance DB is not permitted.

Parameters

Parameters	Declaration	Data type	Range of values	Default	Description
REQ	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Request": activates data exchange at the positive edge.
R	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Reset". Job is aborted. Transmission is locked.
LADDR	INPUT	WORD	CPU specific	W#16#03FF	The I/O address of your submodule, specified by you in "HW Config".
DONE	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): • FALSE Job has not been started or is still being executed. • TRUE Job has been executed error-free.
ERROR	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): Completed job without error
STATUS	OUTPUT	WORD	W#16#0000 to W#16#FFFF	W#16#0000	Status parameter (This parameter is only set for the duration of one call): To display the status, you should copy STATUS to a free data area) STATUS has the following meaning, dependent on the ERROR bit: • ERROR=FALSE: STATUS has the value W#16#0000: Neither the warning nor the error STATUS have the value <> W#16#0000: Warning, STATUS supplies detailed information. • ERROR=TRUE: An error has occurred, STATUS supplies detailed information on the type of error.
SD_1:	INPUT/ OUTPUT	ANY	CPU specific	0	Send parameters: Here you enter the following values: • The number of the DB from which the data are to be transmitted. • The data byte number as of which data are to be transmitted. for example: DB 10 as of byte 2 -> DB10.DBB2
LEN	INPUT/ OUTPUT	INT	1 to 1024	1	Here you declare the length of the data block that is to be transmitted. (Length is set here indirectly.)

Data Consistency

Data consistency is limited to 206 bytes. If you want to consistent data transmission exceeding these 206 bytes, you must take the following into account:

Do not write to the currently used section of the send range SD_1 unless the transmission has been terminated. This is the case when the state parameter DONE has the value TRUE.

25.7 Receiving Data (ASCII, 3964(R)) with SFB 61 "RCV_PTP"

Description

With the **SFB RCV_PTP (SFB 61)** you receive data and then file them in a data block.

The block is ready to receive data after it is called and when the control input **EN_R** is TRUE. You can cancel the current transmission by setting the signal status of parameter EN_R to FALSE. A cancelled job is terminated with an error message (STATUS output). The input is switched off as long as the signal status of parameter EN_R is set to FALSE.

The receiving area declared in **RD_1** (DB number and start address). The length of the data block is declared in **LEN**.

To enable the SFB to process the job, you must call it with **R**(Reset)=FALSE. At the positive edge on control input R the current transmission is aborted and the SFB is reset to basic state. A cancelled job is terminated with an error message (STATUS output).

For your submodule, you declare the I/O address, which you specified by in "HW Config", in **LADDR**.

NDR is set TRUE if the job was terminated without error, or **ERROR** is set TRUE if the job was terminated with an error.

In **STATUS**, the CPU indicates an error or, as a result of a warning, the respective event ID.

NDR or ERROR/STATUS are also output (parameter LEN = 16#00) when the SFB is RESET (R=TRUE).

The binary result BIE is reset if an error has occurred. The status of the binary result is TRUE if the block was terminated without error.

Note

A parameter check is not included in the SFB. The CPU might jump to STOP mode if the configuration is faulty.

Instance DB

The SFB RCV_PTP operates in combination with an instance DB. The DB number is passed on with the call. Access to the data in the instance DB is not allowed.

Parameters

Parameters	Declaration	Data type	Range of values	Default	Description
EN_R	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Enable to receive": Receive enable
R	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Reset". Job is aborted.
LADDR	INPUT	WORD	CPU specific	W#16#03FF	The I/O address of your submodule, specified by you in "HW Config".
NDR	OUTPUT	BOOL	TRUE/FALSE	FALSE	Job done without error, Data was accepted • FALSE Job has not been started or is still being executed • TRUE Job was completed successfully.
ERROR	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): Completed job without error
STATUS	OUTPUT	WORD	W#16#0000 to W#16#FFFF	W#16#0000	Status parameter (This parameter is only set for the duration of one call): To display the status, you should copy STATUS to a free data area) STATUS has the following meaning, dependent on the ERROR bit: • ERROR=FALSE: STATUS has the value W#16#0000: Neither the warning nor the error STATUS have the value <> W#16#0000: Warning, STATUS supplies detailed information. • ERROR=TRUE: An error has occurred, STATUS supplies detailed information on the type of error.
RD_1	INPUT/ OUTPUT	ANY	CPU specific	0	Receive parameter: Here you declare: • The number of the DB in which the received data are to be stored. • The data byte number as of which data are to be stored. for example: DB 20 as of byte 5 -> DB10.DBB2
LEN	INPUT/ OUTPUT	INT	0 to 1024	0	Output of the data length (number of bytes)

Data Consistency

Data consistency is limited to 206 bytes. If you want consistent data transmission to exceed these 206 bytes, you must take the following points into account:

Do not access the receive DB until the data have been completely received (NDR = TRUE). Then, lock the receiving DB (EN_R = FALSE) until you have processed the data.

25.8 Deleting the Receive Buffer (ASCII, 3964(R)) with SFB 62 "RES_RCVB"

Description

You can clear the entire input buffer of the module, using the **SFB RES_RECV (SFB 62)**. All stored telegrams are discarded. An incoming message frame is stored when the SFB RES_RCVB is being called.

The job is activated after the block is called and at the positive edge on control input **REQ**. The job can run across multiple calls (program cycles).

To enable the SFB to process the job, you must call it with **R**(Reset)=FALSE. At the positive edge on control input R the delete process is cancelled and the SFB is reset to basic state. A cancelled job is terminated with an error message (STATUS output).

For your submodule, you declare the I/O address, which was specified by you in "HW Config", in **LADDR**.

DONE is TRUE is the job was terminated without error, or **ERROR** is TRUE if the job was terminated with an error.

In **STATUS**, the CPU indicates an error or, as a result of a warning, the respective event ID.

DONE or ERROR/STATUS are also output when the SFB is RESET (R=TRUE).

The binary result BIE is reset if an error has occurred. The status of the binary result is TRUE if the block was terminated without error.

Note

A parameter check is not included in the SFB. The CPU might jump to STOP mode if the parameterization is faulty.

Instance DB

The SFB RES_RCVB operates in combination with an instance DB. The DB number is passed on with the call. Access to the data in the instance DB is not allowed.

Parameters

Parameters	Declaration	Data type	Range of values	Default	Description
REQ	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Request": Activates the job at the positive edge.
R	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Reset": Job is aborted.
LADDR	INPUT	WORD	CPU specific	W#16#03FF	I/O address of your submodule that you have set in HW Config.
DONE	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): • FALSE Job has not been started or is still being executed. • TRUE Job has been executed error-free.
ERROR	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): Completed job without error
STATUS	OUTPUT	WORD	W#16#0000 to W#16#FFFF	W#16#0000	Status parameter (This parameter is only set for the duration of one call): To display the status, you should copy STATUS to a free data area) STATUS has the following meaning, dependent on the ERROR bit: • ERROR=FALSE: STATUS has the value W#16#0000: Neither the warning nor the error STATUS have the value <> W#16#0000: Warning, STATUS supplies detailed information. • ERROR=TRUE: An error has occurred, STATUS supplies detailed information on the type of error.

25.9 Sending Data (512(R)) with SFB 63 "SEND_RK"

Description

You can send a data from a data block via **SFB SEND_PTP (SFB 63)**.

The send operation is executed after the block is called and a positive edge on control input **REQ**.

The range of data to be transmitted is determined in **SD_1** (DB number and start address). The length of the data block is determined in **LEN**.

In the SFB you also declare the receive range of the partner station. The CPU enters this information in the message frame header (refer also to appendix F) and transfers it to the partner.

The target is specified by the CPU number **R_CPU** (only relevant for multiprocessor communication), the data type in **R_TYPE** (data blocks (DB) and expanded data blocks (DX)), the data block number in **R_DBNO** and the offset in **R_OFFSET** to where the first byte is to be written.

In **R_CF_BYT** and **R_CF_BIT**, declare the connection memory byte and bit on the partner CPU.

In parameter **SYNC_DB**, declare the DB in which the data that you used in all SFBs for the initialization during startup and synchronization is to be stored. The DB numbers must be identical for all SFBs in your user program.

To enable the SFB to process the job, you must call it with **R**(Reset)=FALSE. At the positive edge on control input R the current send operation is cancelled and the SFB is reset to basic state. A cancelled job is terminated with an error message (STATUS output).

For your submodule, you declare the I/O address, which you specified in "HW Config", in **LADDR**.

DONE is set TRUE if the job was terminated without error, or **ERROR** is set TRUE if the job was terminated with an error.

Once the job was processed with DONE = TRUE, the data are sent to the communication partner that confirms them positively and passes them on to the partner CPU.

In **STATUS**, the CPU indicates an error or, as a result of a warning, the respective event ID.

DONE or ERROR/STATUS are also output when the SFB is RESET (R=TRUE).

The binary result BIE is reset if an error has occurred. The status of the binary result is TRUE if the block was terminated without error.

Note

A parameter check is not included in the SFB. If the CPU is assigned the wrong parameters it might jump to STOP mode.

Instance DB

The SFB SEND_RK operates in combination with an instance DB. The DB number is passed on with the call. Access to the data in the instance DB is not allowed.

Special Features for Sending Data

Take the following special features into account when "Sending Data":

- With RK512 you can only send an even number of data. If you declare an odd length (LEN) of data an additional fill byte with the value "0" is appended to the transmitted data.
- In RK512 you can only declare an even offset. If you declare an odd offset the data are stored in the partner station as of the next lower even offset.

  Example: Offset is 7, the data are stored as of byte 6.

Parameters

Parameters	Declaration	Data type	Range of values	Default	Description
SYNC_DB	INPUT	INT	CPU specific, zero is not allowed	0	Number of the DB in which the common data for the synchronization of the RK-SFBs are stored (minimum length = 240 bytes).
REQ	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Request": Activates the job at the positive edge.
R	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Reset". Job is aborted.
LADDR	INPUT	WORD	CPU specific	W#16#03FF	The I/O address of your submodule, specified by you in "HW Config".
R_CPU	INPUT	INT	0 to 4	1	CPU number of the partner CPU (only for multiprocessor operation)
R_TYPE	INPUT	CHAR	‘D’, ‘X’	‘D’	Address type on the partner CPU (only uppercase allowed) ‘D’: Data block ‘X’: Expanded data block
R_DBNO	INPUT	INT	0 to 255	0	Data block number on the partner CPU
R_OFFSET	INPUT	INT	0 to 510 (only even values)	0	Data byte number on the partner CPU
R_CF_BYT	INPUT	INT	0 to 255	255	Connection memory bit on the partner CPU (255: Means: no connection memory bits)
R_CF_BIT	INPUT	INT	0 to 7	0	Connection memory bit on the partner CPU

Parameters	Declaration	Data type	Range of values	Default	Description
DONE	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): • FALSE Job has not been started or is still being executed. • TRUE Job has been executed error-free.
ERROR	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): Completed job without error
STATUS	OUTPUT	WORD	W#16#0000 to W#16#FFFF	W#16#0000	Status parameter (This parameter is only set for the duration of one call): To display the status, you should copy STATUS to a free data area) STATUS has the following meaning, dependent on the ERROR bit: • ERROR=FALSE: STATUS has the value W#16#0000: Neither the warning nor the error STATUS have the value <> W#16#0000: Warning, STATUS supplies detailed information. • ERROR=TRUE: An error has occurred, STATUS supplies detailed information on the type of error.
SD_1:	INPUT/ OUTPUT	ANY	CPU specific	0	Send parameters: Here you declare: • The number of the DB from which the data are to be transmitted. • The data byte number as of which data are to be transmitted. for example: DB 10 as of byte 2 -> DB10.DBB2
LEN	INPUT/ OUTPUT	INT	1 to 1024	1	Here you declare the length of the data block that is to be transmitted. (Length is set here indirectly.)

Declarations in the Message Frame

The table below shows the declarations in the message frame header of the RK 512 message frame.

Source on your S7 automation system (local CPU)	To target, partner CPU	Message frame header, bytes		
		3/4 Instruction type	**5/6 D-DBNR/D Offset**	**7/8 Number in**
Data block	Data block	AD	DB/DW	Words

Source on your S7 automation system (local CPU)	To target, partner CPU	Message frame header, bytes		
		3/4 Instruction type	5/6 D-DBNR/D Offset	7/8 Number in
Data block	Expanded data block	AD	DB/DW	Words

Explanation of the abbreviations used:

D-DBNR	Destination data block number
D Offset	Destination start address
DW	Offset in Words

Data Consistency

Data consistency is limited to 128 bytes. If you want to consistent data transmission exceeding these 128 bytes, you must take the following into account:

Do not write to the currently used section of the send range SD_1 unless the transmission has been terminated. This is the case when the state parameter DONE has the value TRUE.

25.10 Fetching Data (RK 512) with SFB 64 "FETCH RK"

Description

SFB FETCH_RK (SFB 64) is used to fetch a data block from a partner and store them in a data block.

The send operation is executed after the block is called and a positive edge on control input **REQ**.

The area in which the fetched data is stored is declared in **RD_1** (DB number and start address). The length of the data block is declared in **LEN**.

In the SFB you also specify the partner area from which the data are fetched. The CPU enters this information in the RK512 message frame header and transfers it to the partner.

The partner area is determined by the CPU number in **R_CPU** (only relevant for multiprocessor communication), the data type in **R_TYPE** (data blocks, expanded data blocks, memory bits, inputs, outputs, counters and times), the data block number in **R_DBNO** (only relevant for data blocks and expanded data blocks) and the offset in **R_OFFSET** from where the first byte is to be fetched.

In **R_CF_BYT** and **R_CF_BIT** you declare the connection memory byte and the connection memory bit on the partner CPU.

In parameter **SYNC_DB** you declare the DB in which the data that you used in all SFBs for the initialization during startup and synchronization is to be stored. The DB numbers must be identical for all SFBs in your user program.

To enable the SFB to process the job, you must call it with **R**(Reset)=FALSE. At the positive edge on control input R the current transmission is cancelled and the SFB is reset to basic state. An cancelled job is closed with an error message (STATUS output).

For your submodule, you declare the I/O address, which you specified in "HW Config", in **LADDR**.

DONE is set TRUE if the job was terminated without error, or **ERROR** is set TRUE if the job was terminated with an error.

In **STATUS**, the CPU indicates an error or, as a result of a warning, the respective event ID.

DONE or ERROR/STATUS are also output when the SFB is RESET (R=TRUE).

The binary result BIE is reset if an error has occurred. The status of the binary result is TRUE if the block was terminated without error.

Note

A parameter check is not included in the SFB. The CPU might go to STOP mode if the configuration is faulty.

Note

When data are fetched from your CPU, you must program the SFB "SERVE_RK" for your CPU.

Instance DB

The SFB FETCH_RK operates in combination with an instance DB. The DB number is passed on with the call. Access to the data in the instance DB is not allowed.

Special Features for (Expanded) Data Blocks

Note the following special features when "Fetching Data" from a data block or an expanded data block:

- With RK512 you can only send an even number of data. An additional byte is transmitted if you enter an odd length (LEN). In the target DB, however, always the correct number of data is entered.
- In RK512 you can only declare an even offset. If you declare an odd offset the data are stored in the partner station as of the next smaller even offset.

  Example: Offset is 7, the data are stored as of byte 6.

Special Features for Timers and Counters

When you fetch times or counters from your communication partner, you must take into account that you need to fetch two bytes for every time or counter. For example, if you want to fetch 10 counters you must declare a length of 20.

Parameters

Note: In this SFB the range of values are all represented in the German memory conventions.

Parameters	Declaration	Data type	Range of values	Default	Description
SYNC_DB	INPUT	INT	CPU specific, zero is not allowed	0	Number of the DB in which the common data for the synchronization of the RK-SFBs are stored (minimum length = 240 bytes).
REQ	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Request": Activates the job at the positive edge.
R	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Reset". Job is aborted.
LADDR	INPUT	WORD	CPU specific	W#16#03FF	The I/O address of your submodule, specified by you in "HW Config".
R_CPU	INPUT	INT	0 to 4	1	CPU number of the partner CPU

Parameters	Declaration	Data type	Range of values	Default	Description
					(only for multiprocessor operation)
R_TYPE	INPUT	CHAR	'D', 'X', 'M', 'E', 'A', 'Z', 'T'	'D'	Address type on the partner CPU 'D': Data block 'X': Expanded data block 'M': Memory bit 'E': Inputs 'A': Outputs 'Z': Counters 'T': Times
R_DBNO	INPUT	INT	0 to 255	0	Data block number on the partner CPU
R_OFFSET	INPUT	INT	Refer to the table: "Parameter in the FB for data source (Partner CPU)"	0	Data byte number on the partner CPU
R_CF_BYT	INPUT	INT	0 to 255	255	Connection memory bit on the partner CPU (255: Means: no connection memory bits)
R_CF_BIT	INPUT	INT	0 to 7	0	Connection memory bit on the partner CPU
DONE	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): • FALSE Job has not been started or is still being executed. • TRUE Job has been executed error-free.
ERROR	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): Completed job without error
STATUS	OUTPUT	WORD	W#16#0000 to W#16#FFFF	W#16#0000	Status parameter (This parameter is only set for the duration of one call): To display the status, you should copy STATUS to a free data area) STATUS has the following meaning, dependent on the ERROR bit: • ERROR=FALSE: STATUS has the value W#16#0000: Neither the warning nor the error STATUS have the value <> W#16#0000: Warning, STATUS supplies detailed information. • ERROR=TRUE: An error has occurred, STATUS supplies detailed information on the type of error.

Parameters	Declaration	Data type	Range of values	Default	Description
RD_1	INPUT/ OUTPUT	ANY	CPU specific	0	Receive parameter: Here you declare: • The number of the DB in which the fetched data are to be stored. • The data byte number as of which the fetched data are to be stored. For example: DB 10 as of byte 2 -> DB10.DBB2
LEN	INPUT/ OUTPUT	INT	1 to 1024	1	Here you declare the byte length of the data block that is to be fetched. You must declare two bytes per time and per counter. (Length is set here indirectly.)

Parameter in the SFB for Data Source (Partner CPU)

The table below shows the data types which can be transmitted.
The value for R_OFFSET is determined by the partner CPU.

Source on the partner CPU	R_TYP	R_NO	R_OFFSET (in bytes)
Data block	‘D’	0 - 255	0 - 510, only even values are appropriate
Expanded data block	‘X’:	0 - 255	0 - 510, only even values are appropriate
Memory bit	’M’	Irrelevant	0 – 255
Inputs	’E’	Irrelevant	0 – 255
Outputs	’A’	Irrelevant	0 – 255
Counters	‘Z’	Irrelevant	0 – 255
Times	’T’	Irrelevant	0 – 255

Declarations in the Message Frame

The table below shows the declarations in the message frame header of the RK512 message frame.

Source on the partner CPU	to the target, your S7 automation system (local CPU)	Message frame header, bytes		
		3/4 Instruction type	**5/6 S-DBNR/S Offset**	**7/8 Number in**
Data block	Data block	ED	DB/DW	Words
Expanded data block	Data block	EX	DB/DW	Words
Memory bit	Data block	EM	Byte address	Bytes
Inputs	Data block	EI	Byte address	Bytes
Outputs	Data block	EO	Byte address	Bytes
Counters	Data block	EC	Counter number	Words
Times	Data block	ET	Timer number	Words

Explanation of the abbreviations used:

S-DBNO	Source Data Block Number
S Offset	Source start address

Data Consistency

Data consistency is limited to 128 bytes. If you want to consistent data transmission exceeding these 128 bytes, you must take the following into account:

Do not write to the currently used section of the send range SD_1 unless the transmission has been terminated. This is the case when the state parameter DONE value is set to TRUE.

25.11 Receiving and Providing Data (RK 512) with SFB 65 "SERVE_RK"

Description

Use the **SFB SERVE_RK (SFB 65)** to receive and provide data.

- Receiving data: The data are stored in the area that is specified by the partner in the RK512 message frame header. A call of the SFB is required when the communication partner executes a "Send Data" (SEND jobs) job.
- Providing Data: The data are fetched from the area that is specified by the partner in the RK512 message frame header. A call of the SFB is required when the communication partner executes a "Fetch Data" (FETCH jobs) job.

The block is ready to after it is called with the control input **EN_R** value TRUE. You can cancel the current transmission by setting the signal status of parameter EN_R to FALSE. A cancelled job is terminated with an error message (STATUS output). The input is switched off as long as the signal status of parameter EN_R is set to FALSE.

In parameter **SYNC_DB** you declare the DB in which the data that is used by you in all SFBs for the initialization during startup and synchronization is to be stored. The DB numbers must be identical for all SFBs in your user program.

To enable the SFB to process the job, you must call it with **R**(Reset)=FALSE. At the positive edge on control input R the current transmission is cancelled and the SFB is reset to basic state. A cancelled job is terminated with an error message (STATUS output).

For your submodule, you declare the I/O address, which was specified by you in "HW Config", in **LADDR**.

NDR is set TRUE if the job was terminated without error, or **ERROR** is set TRUE if the job was terminated with an error.

With NDR=TRUE for an SFB call the CPU indicates in the parameters **L_TYPE**, **L_DBNO** and **L_OFFSET** the area where data were stored or fetched from. Also shown for a call are the parameters **L_CF_BYT** and **L_CF_BIT** and the length **LEN** of the respective job.

In **STATUS**, the CPU indicates an error or, as a result of a warning, the respective event ID (refer to the appendix).

NDR or ERROR/STATUS are also output (parameter LEN == 16#00) when the SFB is RESET (R=TRUE).

- The binary result BIE is reset if an error has occurred. The status of the binary result is TRUE if the block was terminated without error.

Note

A parameter check is not included in the SFB. The CPU might go to STOP mode if the configuration is faulty.

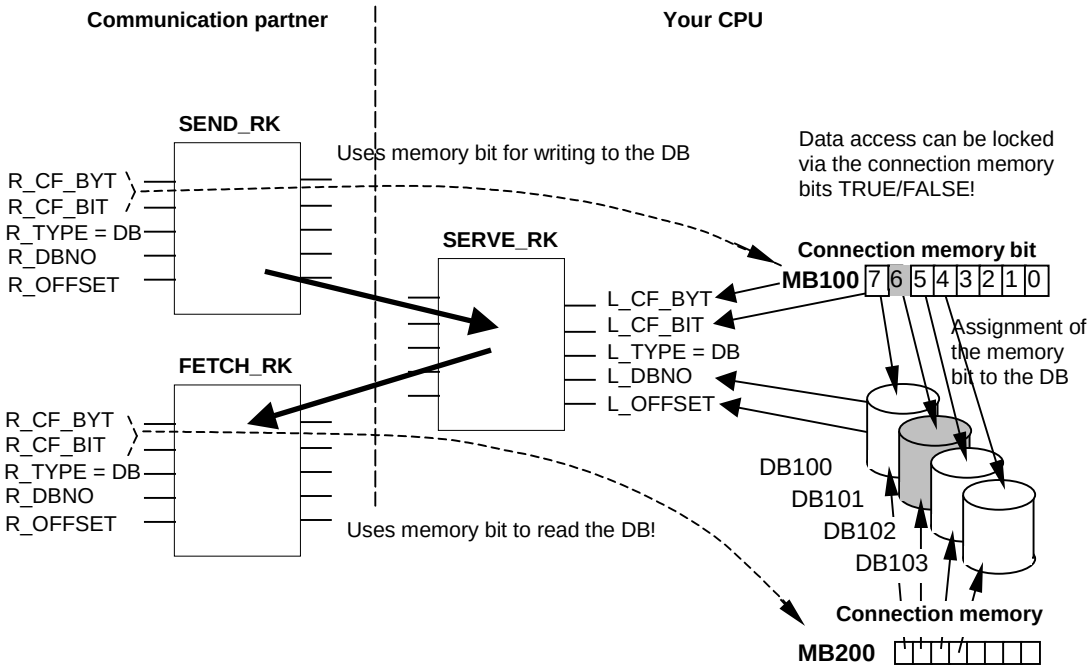
Instance DB

The SFB SERVE_RK operates in combination with an instance DB. The DB number is passed on with the call. Access to the data in the instance DB is not allowed.

How to Use Connection Memory Bits

You can lock or enable SEND and FETCH jobs of your communication partner via connection memory bit. Thus, you can prevent overwriting or reading of data that have not been processed yet.

You can specify a connection memory bit for every job.



Example: SEND_RK with connection memory bit

In this example the communication partner transmits data to DB 101 on your CPU

1. In your CPU, set the connection memory bit 100.6 to FALSE.
2. In your communication partner, specify connection memory bit 100.6 (parameters R_CF_BYT, R_CF_BIT) for the SEND job.

   The connection memory bit is transferred to your CPU in the RK 512 message frame header.

   Before it processes the job, the CPU verifies the connection memory bit that is specified in the RK512 message frame header. Th job is only processed if the connection memory bit is set to FALSE value on your CPU. If the connection memory bit is set to TRUE the error message "32 hex" is transmitted in the response message frame to the communication partner.

   After the data are transferred to the DB101 connection memory 100.6 is set to TRUE on your CPU by SFB SERVE. Also, the connection memory byte and bit is output on SFB SERVE for the duration of one call (if NDR =TRUE).
3. When you evaluate the connection memory (connection memory 100.6 =TRUE) in your user program you can see whether the job is completed and the transmitted data can be processed.
4. After you have processed the data in your user program you must reset the connection memory 100.6 to FALSE. Not until the can your partner execute the job again without error.

Parameters

Note: In this SFB the range of values are all represented in the German memory conventions.

Parameters	Declaration	Data type	Range of values	Default	Description
SYNC_DB	INPUT	INT	CPU specific	0	Number of the DB in which the common data for the synchronization of the RK-SFBs are stored (minimum length = 240 bytes).
EN_R	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Enable to receive" Job enable
R	INPUT	BOOL	TRUE/FALSE	FALSE	Control parameter "Reset". Job is aborted.
LADDR	INPUT	WORD	CPU specific	W#16#03FF	The I/O address of your submodule, specified by you in "HW Config".
NDR	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter "New Data Ready" (This parameter is only set for the duration of one call): • FALSE Job has not been started or is still being executed. • TRUE Job was executed successfully.

Parameters	Declaration	Data type	Range of values	Default	Description
ERROR	OUTPUT	BOOL	TRUE/FALSE	FALSE	Status parameter (This parameter is only set for the duration of one call): Completed job without error
STATUS	OUTPUT	WORD	W#16#0000 to W#16#FFFF	W#16#0000	Status parameter (This parameter is only set for the duration of one call): To display the status, you should copy STATUS to a free data area) STATUS has the following meaning, dependent on the ERROR bit: • ERROR=FALSE: STATUS has the value W#16#0000: Neither the warning nor the error STATUS have the value <> W#16#0000: Warning, STATUS supplies detailed information. • ERROR=TRUE: An error has occurred, STATUS supplies detailed information on the type of error.
L_TYPE	OUTPUT	CHAR	‘D’ ’D’, ’X’, ’M’, ’E’, ’A’, ’Z’, ’T’	’ ’	Receiving data: Type of the target area on the local CPU (only uppercase allowed): ‘D’: Data block Providing data: Type of the source area on the local CPU (only uppercase allowed): ‘D’: Data block ’M’: Memory bit ’E’: Inputs ’A’: Outputs ’Z’: Counters ’T’: Timers This parameter is only set for the duration of one call.
L_DBNO	OUTPUT	INT	CPU specific	0	Data block number on local CPU. This parameter is only set for the duration of one call.
L_OFFSET	OUTPUT	INT	0 - 510	0	Data byte number on local CPU. This parameter is only set for the duration of one call.
L_CF_BYT	OUTPUT	INT	0 to 255	0	Connection memory byte on local CPU. This parameter is only set for the duration of one call. (255: Means: no connection memory)
L_CF_BIT	OUTPUT	INT	0 to 7	0	Connection memory bit on local CPU. This parameter is only set for the duration of one call.

Parameters	Declaration	Data type	Range of values	Default	Description
LEN	INPUT/ OUTPUT	INT	0 to 1024	0	Length of message frame, number in bytes (This parameter is only set for the duration of one call).

Data Consistency

Data consistency is limited to 128 bytes. If you want to consistent data transmission exceeding these 128 bytes, you must take the following into account:

Use the connection memory function. Do not access the data until they are completely transmitted (evaluation of the connection memory specified for this job; the connection memory is active for a call on SFB if NDR = TRUE). Do not reset the connection memory to FALSE unless you have processed the data.

25.12 Additional Error Information of the SFBs 60 to 65

Error Information

The table below shows the diverse event classes and event IDs.

Error in the SFB parameter configuration"		
Event class Error code	**Event**	**Remedy**
W#16#0301	Source/destination data type illegal or does not exist. Illegal range (start address, length). DB illegal or does not exist. Other data type is illegal or does not exist. Invalid connection memory byte or bit ID.	Verify the parameterization and correct it if required. Partner delivers illegal parameters in the message frame header. Verify the parameters, create a block if required. Refer to the job tables for info on permitted data types. The partner delivers the wrong parameters in the message frame header.
W#16#0303	Range cannot be accessed	Verify the parameters. Refer to the job tables for info on permitted start addresses and lengths, or the partner supplies the wrong parameters in the message frame header.

"Job processing errors"		
Event class Error code	**Event**	**Remedy**
W#16#0501	The current job was aborted by restart or reset.	Repeat the aborted job. When you re-assign parameters via PG you should make sure that no more jobs are being processed before you write to an interface.
W#16#0502	The job is not allowed while in this operating state (for example, no parameters assigned to the device interface).	Assign parameters to the device interface.
W#16#050E	• Invalid message frame length • The end-of-message ID assigned in the parameters has not occurred within the maximum permitted length.	• The message frame length > 1024 bytes. Select a smaller message frame length • Add the end-of-message IDs at the desired position in the send buffer.
W#16#0513	Data type error (DB ...): Unknown data type or illegal data type (for example, DE) The source and target data types specified in the SFB do not match.	Refer to the job tables for info on permitted data types and their combinations.
W#16#0515	Wrong bit number declared in the coordination memory.	Permitted bit no. 0 to 7
W#16#0516	CPU number specified too high.	Permitted CPU no. 0, 1, 2, 3 or 4
W#16#0517	Transmission > 1024 byte is too large	Split the job into several jobs of a smaller length.
W#16#051D	Send/receive job aborted by • Reset of the communication block • Re-assigning parameters	Repeat the call of the communication block.
W#16#0522	A new SEND job was started even though the previous job has not yet been completed yet.	Do not start the new SEND job unless the previous send job is terminated with DONE or ERROR.

"Error when processing a partner job" only for RK512		
Event class Error code	**Event**	**Remedy**
W#16#0601	Error in the 1st instruction byte (not 00 or FFH)	Basic header structure error in the partner. Prove the faulty behavior of the partner device if required by hooking up an interface tester to the data link.
W#16#0602	Error in 3rd instruction byte (not A, 0 or E)	Basic header structure error in the partner. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0603	Error in 3rd instruction byte in subsequent message frames (instruction not as in the first message frame)	Basic header structure error in the partner. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0604	Error in 4th instruction byte (wrong instruction character)	Basic header structure error in the partner or an illegal instruction combination was requested. Check the permitted instructions. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0606	Error in 5th instruction byte (illegal DB number)	Refer to the job tables for info on permitted DB numbers, start addresses or lengths.
W#16#0607	Error in 5th or 6th instruction byte (start address too high)	Refer to the job tables for info on permitted DB numbers, start addresses or lengths.
W#16#0609	Error in 9th or 10th instruction byte (coordination memory bit not permitted for this data type or the bit number is too high).	Basic header structure error in the partner. Refer to the job tables for info on when a coordination memory bit is permitted.
W#16#060A	Error in 10th instruction byte (illegal CPU number)	Basic header structure error in the partner.

"Send error"		
Event class Error code	**Event**	**Remedy**
W#16#0701	**Only for 3964(R):** Sending the first repetition: • An error was detected when sending the message frame • The partner requested a repetition with a negative confirmation character (NCC).	A repetition does not represent an error. However, it can be an indication of disruptions on the data link or of a faulty behavior of the partner. If the message frame is not transmitted within the maximum number of repetitions an error number is reported which describes the error that first occurred.
W#16#0702	**Only for 3964(R):** Error while establishing the connection After STX was transmitted the NCC or any character (except DLE or STX) was received.	Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0703	**Only for 3964(R):** Confirmation time (QVZ) exceeded: After the transmission of STX the partner did not respond within the confirmation delay time.	The partner device is too slow or not ready to receive, or the data link is interrupted. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0704	**Only for 3964(R):** Abortion by the partner: One or several characters were received from the partner while the send operation was in progress.	Check whether the partner also indicates an error because possibly it has not received all of the transmitted data (for example, due to an interrupted data link), or because fatal errors are pending, or the behavior of the partner device is faulty. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0705	**Only for 3964(R):** Negative confirmation while transmission was in progress	Check whether the partner also indicates an error because possibly it has not received all of the transmitted data (for example, due to an interrupted data link), or because fatal errors are pending, or the behavior of the partner device is faulty. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0706	**Only for 3964(R):** End-of-transmission error: • The partner has rejected the message frame at the end with NCC or any character (except DLE) • The confirmation character (DLE) was received too early.	Check whether the partner also indicates an error because possibly it has not received all of the transmitted data (for example, due to an interrupted data link), or because fatal errors are pending, or the behavior of the partner device is faulty. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0707	**Only for 3964(R):** End-of-transmission confirmation delay time / response watchdog time was exceeded: The partner did not respond within the QVZ after the connection went down via DLE ETX.	The partner device is too slow or disrupted. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.

"Send error"		
Event class Error code	**Event**	**Remedy**
W#16#0708	**Only for ASCII drivers:** The waiting time on XON has expired.	The communication partner is disrupted, too slow or switched offline. Check the communication partner or change the parameters if required.
W#16#0709	**Only for 3964(R):** Could not establish a connection, the permitted number of startup attempts was exceeded.	Check the interface cord or the transmission parameters. Also check in the partner whether the parameters for the receive function between CPU and CP have been correctly assigned.
W#16#070A	**Only for 3964(R):** Could not transmit data, the permitted number of attempts to transmit was exceeded.	Check the interface cord or the transmission parameters.
W#16#070B	**Only for 3964(R):** Initialization conflict cannot be solved because both partners are set to high priority.	Change the parameters.
W#16#070C	**Only for 3964(R):** Initialization conflict cannot be solved because both partners are set to low priority.	Change the parameters.

"Receive error"		
Event class Error code	**Event**	**Remedy**
W#16#0801	Only for 3964(R): Expecting the first repetition: An error was detected when the message frame was received and the CPU requested a repetition with a negative confirmation (NCC) from the partner.	A repetition does not represent an error. However, it can be an indication of disruptions on the data link or of a faulty behavior of the partner. If the message frame is not transmitted within the maximum number of repetitions an error number is reported which describes the error that first occurred.
W#16#0802	Only for 3964(R): Error while establishing the connection • One or several characters (except NCC or STX) were received when idle • After having received the STX the partner transmitted more characters without waiting for the response DLE. After the partner is powered ON: • The CPU receives an undefined character while the partner is switched on.	Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0805	Only for 3964(R): Logical receive error: After receiving the DLE another character was received (except DLE, ETX).	Check whether the partner doubles the DLE in the message frame header and in the data string or if the connection is established via DLE ETX. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0806	Character Delay Time (CDT) was exceeded: • Two subsequent characters were not received within the CDT. Only for 3964(R): • 1. The character was not received within the CDT after sending the DLE when the connection was established.	The partner device is too slow or disrupted. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#0807	Illegal message frame length: A message frame with 0 length was received.	Receiving a message frame with 0 length does not represent an error. Check why the communication partner transmits message frames without user data.
W#16#0808	Only for 3964(R): Block Check Character error BCC: The internally generated value for the BCC does not correspond with the BCC received by the partner at the end of the communication link.	Check whether the communication is seriously disrupted. In this case you can also see occasional error codes. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.

"Receive error"		
Event class Error code	**Event**	**Remedy**
W#16#0809	**Only for 3964(R):** The delay time for block repetition has expired	Declare the same block delay time in the parameters for the communication partner and your module. Prove the faulty behavior of the partner if required by hooking up an interface tester to the data link.
W#16#080A	There is no free input buffer: There was no cleared input buffer available for receiving.	The SFB RCV must be called more frequently.
W#16#080C	Transfer error • A transfer error was detected (parity/stop bit/overflow error). Only for 3964(R): • If a disrupted character is received while in idle mode the error is reported immediately in order to recognize disturbing influences on the data link as soon as possible. Only for 3964(R): • The repetitions are started if this happens during send and receive operations.	Disturbances on the data link cause message frame repetitions and, thus, reduce user data throughput. The risk of not detecting an error increases. Change your system structure or your wiring. Check the data links of the communication partners or whether the same baud rate, parity and stop bits are set on both devices.
W#16#080D	BREAK: The receive link to the partner is interrupted.	Establish the link again or switch on the partner.
W#16#080E	Input buffer overflow while flow control is not enabled.	The SFB for receiving must be called more frequently in the user program or you must assign parameters with flow control to the communication.
W#16#0810	Parity error	Check the data links of the communication partners or whether the same baud rate, parity and stop bits are set on both devices.
W#16#0811	Character frame error	Check the data links of the communication partners or whether the same baud rate, parity and stop bits are set on both devices. Change your system structure or your wiring.
W#16#0812	Only for ASCII drivers: Further characters were received after the CPU has transmitted XOFF.	Re-assign the parameters in the communication partner or dispose of the data faster.
W#16#0814	Only for ASCII drivers: One or several message frames were lost because operation was carried out without flow control.	Operate with flow control as far as possible. Utilize the entire input buffer. In the basic parameters set the "Reaction to CPU STOP" parameter to "Continue operation".
W#16#0816	The length of a received message frame exceeded the maximum specified length.	Needs to be corrected in the partner station.

"Receiving a reaction message frame with error or an error message frame from the communication partner"		
Event class Error code	**Event**	**Remedy**
W#16#0902	**Only for RK 512:** Memory access error in the partner station (memory does not exist) With SIMATIC S5 as partner: • Wrong range in the display word • Data range does not exist (except DB/DX) • Data range insufficient (except DB/DX)	Check whether the partner is equipped with the required data range and whether it is of a sufficient size or check the parameters of the called SFB. Check the length specified in the SFB.
W#16#0903	**Only for RK 512:** DB/DX access error in the partner station (DB/DX does not exist or too short) With SIMATIC S5 as partner: • DB/DX does not exist • DB/DX too short • Illegal DB/DX no. Permitted source range exceeded by FETCH job.	Check whether the partner is equipped with the required data range and whether it is of a sufficient size or check the parameters of the called SFB. Check the length specified in the SFB.
W#16#0904	**Only for RK 512:** Partner reports “Job type not permitted”.	Faulty partner behavior because the CPU never outputs a system instruction.
W#16#0905	**Only for RK 512:** Partner error or SIMATIC S5 partner error: • Source/target type not permitted • Memory error in partner device • Error in partner CP/CPU communication • Partner PLC is in STOP mode	Check whether the partner is able to transfer the desired data type. Check the partner’s hardware structure. Switch the partner PLC to RUN mode.
W#16#0908	**Only for RK 512:** Partner recognizes synchronization error: The message frame sequence is disrupted.	This error occurs when you restart your own or your partner’s PLC. This is a normal startup behavior of the system. No remedies are required. When operation is in progress this error might occur as a result of previous errors. Otherwise, you can assume faulty behavior of the partner.
W#16#0909	**Only for RK 512:** DB/DX locked in the partner by coordination memory bit.	In the partner program: Reset the coordination memory bit after you have processed the last transmitted data! The program: Repeat job!
W#16#090A	**Only for RK 512:** Errors in the message frame header recognized by the partner: 3. Wrong instruction byte in the header	Check whether the error is the result if disturbances or faulty partner behavior. Prove this with the help of an interface tester you hook up into the data link.
W#16#090C	**Only for RK 512:** Partner detects wrong message frame length (length total).	Check whether the error is the result if disturbances or faulty partner behavior. Prove this with the help of an interface tester you hook up into the data link.

"Receiving a reaction message frame with error or an error message frame from the communication partner"		
Event class Error code	**Event**	**Remedy**
W#16#090D	**Only for RK 512:** Up to now there is no restart at the partner.	Restart the partner PLC or set the operating mode selection switch to RUN.
W#16#090E	**Only for RK 512:** Received unknown error number in the response message frame.	Check whether the error is the result if disturbances or faulty partner behavior. Prove this with the help of an interface tester you hook up into the data link.
"The CPU has detected errors in the response message frame of the partners"		
Event class Error code	**Event**	**Remedy**
W#16#0A02	**Only for RK 512:** Error in the structure of the received response message frame (1. Byte not 00 or FF)	Prove the faulty behavior of the partner, if required, by hooking up an interface tester to the data link.
W#16#0A03	**Only for RK 512:** received response message frame contains too many or insufficient data.	Prove the faulty behavior of the partner, if required, by hooking up an interface tester to the data link.
W#16#0A05	**Only for RK 512:** No response message frame from the partner within the monitoring time.	Is the partner a very slow device? Quite often this error is also displayed as a result of a previous error. For example, receive procedure errors (event class 8) can be displayed after a FETCH message frame was transmitted. Reason: The response message frame could not be received due to disturbances, the watchdog time expires. This error might also occur if the partner is restarted before it was able to respond to the last received FETCH message frame.
"Warnings"		
Event class Error code	**Event**	**Remedy**
W#16#0B01	Input buffer loaded over 2/3 of its capacity	Call the receive block more frequently in order to avoid an input buffer overflow.

26 SFCs for H CPUs

26.1 Controlling Operation in H Systems with SFC 90 "H_CTRL"

Description

With SFC 90 "H_CTRL," you can influence H systems as follows:

- You can prevent the standby link-up in the master CPU. This is then disabled until you cancel the setting with SFC 90 "H_CTRL."

  Any request from the standby CPU to link-up with the master is stored.

- You can disable updating on the master CPU. This is then disabled until you cancel the setting with SFC 90 "H_CTRL."

  Any request from the standby CPU to update is stored.

Caution

If you have disabled update but not connect, the hardware system still can determine the connection status as before. Please note that when the master CPU is connecting, it does not process any remove/insert interrupts, station failure/returned interrupts or rack failure/returned interrupts.

- You can remove a test component from the cyclical self-test, add it again or start immediately.

Note

If you use a CPU 414-4H or 417-4H in a redundant system, please observe the following: If you disable the component for more than 24 hours, the CPU goes into STOP mode. For redundant systems, the applicable regulation states that certain tests must be completed within 24 hours.

The following table explains the permitted combinations of the input parameters MODE and SUBMODE.

Job	MODE Input	SUBMODE Input
Disable link-up	3	0
Re-enable link-up	4	0
Disable updating	1	0
Re-enable updating	2	0
Remove the test component specified in the SUBMODE from the cyclical self-test. A test component can only be removed once.	20	0.1,...5
Add the test component specified in the SUBMODE to the cyclical self-test again. A test component can only be added again if it has been previously removed.	21	0.1,...5
Start the test component specified in the SUBMODE immediately. The test component can't have been removed.	22	0.1,...5

The following table shows the assignment of the individual test components for the cyclical self-test with the SUBMODE input values. (only relevant for the values 20, 21, and 22 of the input MODE)

Value from SUBMODE	Associated Test Component
0	SP7 – ASIC – Test
1	Code memory test
2	Data memory test
3	Operating system code checksum test
4	Code block checksum test
5	Comparison of numbers, times, markers and data blocks in redundant operation.

How the SFC Operates

SFC 90 "H_CTRL" is an asynchronous SFC, in other words its execution can extend over several SFC calls.

You start the job by calling SFC 90 with REQ=1.

If the job could be executed immediately, the SFC returns the value 0 at the BUSY output parameter. Initialization of a long-term test routine ends with the first SFC call (BUSY=0), even if the test covers multiple cycles (RET_VAL=W#16#0001 with MODE=22). If BUSY has the value 1, the job is still active (see also Meaning of the Parameters REQ, RET_VAL and BUSY with Asynchronous SFCs).

Identifying A Job

The input parameters MODE and SUBMODE specify the job. If these match a job that is not yet completed, the SFC call is a follow-on call.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	I, Q, M, D, L, const.	Level-triggered control parameter REQ=1: Triggers the job
MODE	INPUT	BYTE	I, Q, M, D, L, const.	Job
SUBMODE	INPUT	WORD	I, Q, M, D, L, const.	Secondary job
RET_VAL	OUTPUT	INT	I, Q, M, D, L	If an error occurs while the function is being executed, the return value contains an error code. Make sure that you evaluate RET_VAL each time the block has been executed.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	BUSY=1: The job is not yet completed.

Error Information

Error Code (W#16#...)	Explanation
0000	Job executed without error
7000	REQ = 0 at first call: the job was not activated; BUSY has the value 0.
7001	REQ = 1 at first call: the job was started; BUSY has the value 1.
7002	Follow-on call (REQ irrelevant). The activated job is still running; BUSY has the value 1.
0001	• When MODE=1: updating was already disabled. • When MODE=3: link-up was already disabled. • When MODE=22: the test component is already running and cannot be restarted.
8082	• When MODE=1: updating is already active and can no longer be disabled. • When MODE=3: link-up is already active and can no longer be disabled. • When MODE=20: the test component indicated has already been removed from the cyclical self-test. • When MODE=21: the test component indicated has not been removed from the cyclical self-test. • When MODE=22: the test component indicated cannot be executed because you have removed in from the cyclical self-test.
8090	The MODE input parameter has an invalid value.
8091	The SUBMODE input parameter has an invalid value.

Example of Using SFC 90

With SFC 90 "H_CTRL," you can make sure that no link-up and no updating is started at times when the maximum resources of the CPU are required.

You can achieve this by including the following program sections on the master CPU prior to the period of increased process activity:

- Call SFC 90 with MODE = 3 and SUBMODE =0 (disable link-up)
- Call SFC 90 with MODE = 1 and SUBMODE = 0 (disable updating)

At the end of the period of increased activity, include the following program sections on the master CPU:

- Call SFC 90 with MODE = 4 and SUBMODE = 0 (re-enable link-up)
- Call SFC 90 with MODE = 2 and SUBMODE = 0 (re-enable updating).

27 SFCs for WinLC RTX

27.1 Updating the Process Image Partition in a Synchronous Cycle with SFC 126 "SYNC_PI"

Description

With SFC 126 "SYNC_PI“ you can update a process image partition input table in a synchronous cycle. A user program linked to a DP cycle can use this SFC to consistently and synchronously update input data located in a process image partition.

SFC 126 can be interrupted and can only be called in OBs 61, 62, 63 and 64.

Note

A call of SFC 126 "SYNC_PI" in OBs 61 to 64 is only permitted if you have assigned the affected process image partition to the associated OB in HW Config.

Parameters

Parameter	Declaration	Data type	Value range	Default	Meaning
PART	INPUT	BYTE	1 to 30		Number of the process image partition input table to be updated in a synchronous cycle.
RET_VAL	OUTPUT	INT			Error information
FLADDR	OUTPUT	WORD			Address of the first byte to cause an error, in case of an access error.

Error information

Event class Error code	Explanation
W#16#8090	Illegal value at parameter PART or and update of the specified process image partition input table is not a not permitted in this OB. The process image partition input table was not updated.
W#16#8091	The specified process image partition was still not defined or is not located in a accessible process image area on the CPU. The process image partition input table was not updated.
W#16#80A0	During updating an access error was detected. The affected inputs were set to "0".
W#16#80A1	The update time lies after the permitted access window or the input data were not updated by the DP master. The process image partition input table was not updated.
W#16#80C1	The update time lies before the permitted access window. The process image partition input table was not updated.

27.2 Updating the Process Image Partition in a Synchronous Cycle with SFC 127 "ISO_PO"

Description

With SFC 127 "SYNC_PO“ you can update a process image partition output table in a synchronous cycle. A user program linked to a DP cycle can use this SFC to synchronously update output data located in a process image partition and consistently transmit them to I/O devices.

SFC 127 can be interrupted and can only be called in OBs 61, 62, 63 and 64.

Note

A call of SFC 127 "SYNC_PO" in OBs 61 to 64 is only permitted if you have assigned the affected process image partition to the associated OB in HW Config.

Parameter

Parameter	Declaration	Data type	Value range	Default	Meaning
PART	INPUT	BYTE	1 to 30		Number of the process image partition output table to be updated in a synchronous cycle.
RET_VAL	OUTPUT	INT			If an error occurs while processing this function, the return value contains an error code.
FLADDR	OUTPUT	WORD			Address of the first byte to cause an error,

Error information

Event class Error code	Explanation
W#16#0001	Consistency warning. The update of the process image partition table was distributed over two DP cycles. However, the data in one slave were consistently transferred.
W#16#8090	Illegal value at parameter PART or and update of the specified process image partition output table is not a not permitted in this OB. Outputs were not transferred to the I/O devices. The process image partition output table was not changed.
W#16#8091	The specified process image partition was still not defined or is not located in a accessible process image area on the CPU. Outputs were not transferred to the I/O devices. The process image partition output table was not changed.
W#16#80A0	During updating an access error was detected. Outputs were not transferred to the I/O devices. The process image partition output table was not changed.
W#16#80A1	The update time lies after the permitted access window or the output data were not updated by the DP master. Outputs were not transferred to the I/O devices. The process image partition output table was not changed.
W#16#80C1	The update time lies before the permitted access window. Outputs were not transferred to the I/O devices. The process image partition output table was not changed.

27.3 Determining the OB Program Run Time with SFC 78 "OB_RT"

Description

With SFC 78 “OB_RT” you can determine the run times for individual OBs over different time periods.

Parameter

Parameter	Declaration	Data type	Memory area	Meaning
OB_NR	INPUT	INT	I, O, M, D, L	The OB whose last evaluated times are to be queried. Valid OB numbers are those for the interrupt and asynchronous error OBs in the given CPU. Processing of synchronous errors is counted as part of the processing time of the given error-causing OB. If OBs 121 or 122 or OBs not in the CPU are indicated, this will result in an error message. For OB=0, the data for the OB used to call the SFC are transferred. If SFC 78 is called in OBs 121 or 122 with OB_NR=0, the times of the OB causing the interrupt including the times in OB 12x are output.
RET_VAL	OUTPUT	INT	I, O, M, D, L	If an error occurs while processing this function, the return value contains an error code. Otherwise, RET_VAL contains the OB number for which these data were requested.
PRIO	OUTPUT	INT	I, O, M, D, L	The priority class of the queried OB is output in PRIO
LAST_RT	OUTPUT	DINT	I, O, M, D, L	The run time for the last completed processing of the given OB in microseconds. If no OB number is indicated or OB_NR=0, the queried OB time is deleted. Until the next processing of this OB is complete, a run time of 0xFFFF FFFF will be indicated.
LAST_ET	OUTPUT	DINT	I, O, M, D, L	The elapsed time in microseconds since the last completed processing of the given OB. If no OB number is indicated or OB_NR=0, the queried OB time is deleted. Until the next processing of this OB is complete, a run time of DW#16#FFFF FFFF will be indicated.

Parameter	Declaration	Data type	Memory area	Meaning
CUR_T	OUTPUT	DINT	I, O, M, D, L	Time of the OB request of the given OB being processed as a relative time value in microseconds.
CUR_RT	OUTPUT	DINT	I, O, M, D, L	Elapsed run time in microseconds of the current processing of the given OB. CUR_RT is 0 if the OB is (still) not being processed. After processing, the run time in LAST_RT is applied and CUR_RT is set to 0.
CUR_ET	OUTPUT	DINT	E, A, M, D, L	The elapsed time in microseconds sincethe request for the given OB being processed. CUR_ET is 0 if the OB is not being processed. After processing, the run time in LAST_RT is applied and CUR_ET is set to 0.
NEXT_ET	OUTPUT	DINT	E, A, M, D, L	The elapsed time in microseconds since the next pending start event of the given OB. OB. NEXT_ET is 0 if there are no further start events to process except those currently being processed or pending for the given OB. WinLC RTX does not use this parameter.

The times also include the run times for any nested processing of synchronous error interrupts (OB 121, OB 122).

Error information

Event class Error code	Explanation
W#16#0001 to W#16#0102	Number of the OB to which information is being transferred.
W#16#8080	OB_NR parameter contains an illegal value.

28 Integrated Functions (for CPUs with integrated I/Os)

28.1 SFB 29 (HS_COUNT)

Description

With SFB 29 "HS_COUNT" (counter), you can influence the integrated counter function of a CPU with integrated I/Os, as follows:

- Set and enter a start value.
- Select and set comparison values.
- Enable counters.
- Enable digital outputs.
- Read current counted values and current comparison values.
- Query the relationship between the counted value and the comparison value.

Further Information

The meaning of the individual parameters of SFB 29 in conjunction with the parameters for the integrated function counter and the hardware inputs and outputs of the CPU is described in detail in *S7-300 Programmable Controller, Integrated Functions* manual.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
PRES_COUNT	INPUT	DINT	I, Q, M, D, L, constant	Start value for the counter
PRES_COMP_A	INPUT	DINT	I, Q, M, D, L, constant	New comparison value COMP_A
PRES_COMP_B	INPUT	DINT	I, Q, M, D, L, constant	New comparison value COMP_B
EN_COUNT	INPUT		I, Q, M, D, L	Enable the counter
EN_DO	INPUT	BOOL	I, Q, M, D, L, constant	Enable the digital outputs
SET_COUNT	INPUT	BOOL	I, Q, M, D, L, constant	Set input for the start value PRES_COUNT
SET_COMP_A	INPUT	BOOL	I, Q, M, D, L, constant	Set input for the comparison value COMP_A
SET_COMP_B	INPUT	BOOL	I, Q, M, D, L, constant	Set input for the comparison value COMP_B
COUNT	OUTPUT	DINT	I, Q, M, D, L	Actual value of the counter
COMP_A	OUTPUT	DINT	I, Q, M, D, L	Current comparison value COMP_A
COMP_B	OUTPUT	DINT	I, Q, M, D, L	Current comparison value COMP_B
STATUS_A	OUTPUT	BOOL	I, Q, M, D, L	Status bit STATUS_A 1: COUNT ≥ COMP_A 0: COUNT < COMP_A
STATUS_B	OUTPUT	BOOL	I, Q, M, D, L	Status bit STATUS_B 1: COUNT ≥ COMP_B 0: COUNT < COMP_B

28.2 SFB 30 (FREQ_MES)

Description

With SFB 30 "FREQ_MES" (frequency meter), you can influence the integrated frequency meter function of a CPU with integrated I/Os, as follows:

- Select and set comparison values.
- Output the measured frequency.
- Read the current comparison values.
- Query the relationship of the measured frequency to the comparison value.

Further Information

The meaning of the individual parameters of SFB 30 in conjunction with the parameters for the integrated frequency meter function and the hardware inputs and outputs of the CPU is described in detail in the *S7-300 Programmable Controller, Integrated Functions* manual.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
PRES_U_LIMIT	INPUT	DINT	I, Q, M, D, L, constant	New (upper) comparison value U_LIMIT
PRES_L_LIMIT	INPUT	DINT	I, Q, M, D, L, constant	New (lower) comparison value L_LIMIT
SET_U_LIMIT	INPUT	BOOL	I, Q, M, D, L, constant	Set input for new comparison value U_LIMIT
SET_L_LIMIT	INPUT	BOOL	I, Q, M, D, L, constant	Set input for new comparison value L_LIMIT
FREQ	OUTPUT	DINT	I, Q, M, D, L	Measured frequency in MHz
U_LIMIT	OUTPUT	DINT	I, Q, M, D, L	Current comparison value (upper limit)
L_LIMIT	OUTPUT	DINT	I, Q, M, D, L	Current comparison value (lower limit)
STATUS_U	OUTPUT	BOOL	I, Q, M, D, L	Status bit "1": FREQ > U_LIMIT "0": FREQ ≤ U_LIMIT
STATUS_L	OUTPUT	BOOL	I, Q, M, D, L	Status bit "1": FREQ < L_LIMIT "0": FREQ ≥ U_LIMIT

28.3 SFB 38 (HSC_A_B)

Description

With SFB 38 (HSC_A_B), you can influence the integrated A/B counter function of a CPU with integrated inputs/outputs, as follows:

- Specify and adopt the start value
- Specify and set comparison values
- Enable counters
- Enable digital outputs
- Read current counted values and current comparison values
- Query the counted value relative to the comparison value

SFB 38 (HSC_A_B) reads or writes data from the user program in the instance DB of the integrated function. The A/B counter consists of two counters A and B that can count simultaneously and are independent of each other (counting up and down is possible).
The counters function identically; count pulses can be registered up to a frequency of 10 kHz.

Further Information

The precise meaning of the parameters of SFB 38 in conjunction with the parameters of the integrated function A/B counter and the hardware inputs and outputs of the CPU is described in detail in the manual *S7-300 Programmable Controller, Integrated Functions CPU 312 IFM/314 IFM*.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
PRES_COMP	INPUT	DINT	I, Q, M, D, L, constant	New comparison value COMP
EN_COUNT	INPUT	BOOL	I, Q, M, D, L constant	Enable the counter
EN	INPUT	BOOL	I, Q, M, D, L, constant	Enable the digital output
SET_COMP	INPUT	BOOL	I, Q, M, D, L, constant	Set input for comparison value COMP
COUNT	OUTPUT	DINT	I, Q, M, D, L	Actual value of the counter
COMP	OUTPUT	DINT	I, Q, M, D, L	Current comparison value COMP
ENO	OUTPUT	BOOL	I, Q, M, D, L	Error handling: 1 : no error in execution 0 : error in execution

28.4 SFB 39 (POS)

Description

With SFB 39 (POS), you can influence the integrated positioning function of a CPU with integrated inputs/outputs. SFB 39 (POS) provides the following functions:

- Synchronization
- Execution of the inching mode
- Positioning

SFB 39 (POS) for the integrated positioning function reads or writes data from the user program to the instance DB of the integrated function. The integrated positioning function acquires signals from asymmetrical 24 V incremental encoders up to a frequency of 10 kHz. It controls a rapid/creep mode or a frequency converter via specified integrated outputs of the CPU 314 IFM (controlled positioning)

Further Information

The precise meaning of the parameters of SFB 39 in conjunction with the parameters of the integrated function A/B counter and the hardware inputs and outputs of the CPU is described in detail in the manual *S7-300 Programmable Controller, Integrated Functions CPU 312 IFM/314 IFM*.

Parameter

Parameter	Declaration	Data Type	Memory Area	Description
EN	INPUT	BOOL	I, Q, M, D, L, constant	Enable the digital inputs
DEST_VAL	INPUT	DINT	I, Q, M, D, L, constant	Destination position for the integrated positioning function
REF_VAL	INPUT	DINT	I, Q, M, D, L, constant	Reference point for synchronization
SWITCH_OFF_DIFF	INPUT	WORD	I, Q, M, D, L, constant	Switch-off difference (difference between the switch-off point and the destination position) in travel increments
PRES_COMP	INPUT	DINT	I, Q, M, D, L, constant	New comparison value COMP
BREAK	INPUT	BYTE	I, Q, M, D, L, constant	Maximum analog value with which the traversing movement is controlled
POS_MODE1, POS_MODE2	INPUT	BOOL	I, Q, M, D, L, constant	Start and execute inching mode
POS_STRT	INPUT	BOOL	I, Q, M, D, L, constant	Start positioning operation on rising edge

Parameter	Declaration	Data Type	Memory Area	Description
SET_POS	INPUT	BOOL	I, Q, M, D, L, constant	When there is a rising edge, the value of the input parameter REF_VAL is adopted as the new actual value
ENO	OUTPUT	BOOL	I, Q, M, D, L	Error handling: 1 : no error in execution 0 : error in execution
ACTUAL_POS	OUTPUT	DINT	I, Q, M, D, L	Current actual value
POS_READY (status message)	OUTPUT	BOOL	I, Q, M, D, L	Positioning / inching completed if POS_READY=1
REF_VALID (status message)	OUTPUT	BOOL	I, Q, M, D, L	Reference point switch reached or not
POS_VALID (status message)	OUTPUT	BOOL	I, Q, M, D, L	Actual position of the axis synchronized with the actual position of the integrated function

29 Plastics Techology

29.1 SFC 63 (AB_CALL)

Description

SFC 63 (AB_CALL) calls an assembly code block.
Assembly code blocks are logic blocks that were written in the programming language "C" or in Assembler and then compiled.

Application

You can only use assembly code blocks for the CPU 614.

Further Information

The meaning of the individual parameters of SFC 63 is explained in detail in the documentation for the CPU 614. There is a separate programming guide for programming assembly code blocks.

Parameters

Parameter	Declaration	Data Type	Memory Area	Description
AB_NUMBER	INPUT	WORD	I, Q, M, D, L, constant	Bits for the assembly code blocks to be called
CALL_REASON	INPUT	WORD	I, Q, M, D, L, constant	Organization block in which the SFC was called or evaluation of the DB pointer (parameter DB_NUMBER) or activation of the debugger
DB_NUMBER	INPUT	WORD	I, Q, M, D, L, constant	Number of the DB pointer
RET_VAL	OUTPUT	WORD	I, Q, M, D, L	Return value of the SFC

30 Diagnostic Data

30.1 Overview of the Structure of Diagnostic Data

Data Record 0 and 1 of the System Data

The diagnostic data of a module are located in data records 0 and 1 of the system data area (see Writing and Reading Data Records).

- Data record 0 contains 4 bytes of diagnostic data that describe the current status of a signal module.
- Data record 1 contains

- The 4 bytes of diagnostic data, also located in data record 0 and
- The diagnostic data specific to the module.

Structure and Contents of the Diagnostic Data

This section describes the structure and contents of the individual bytes of the diagnostic data.

Whenever an error occurs, the corresponding bit is set to "1."

30.2 Diagnostic Data

Structure and contents of the diagnostic data:

Byte	Bit	Meaning	Remarks
0	0	Module fault	
	1	Internal error	
	2	External error	
	3	Channel error	
	4	No external auxiliary voltage	
	5	No front connector	
	6	No parameter assignment	
	7	Wrong parameters in the module	

Byte	Bit	Meaning	Remarks	
1	0 to 3	Module class	0101 0000 1000 1100 1111 0011 1011 0100	Analog module CPU Function module CP Digital module DP standard slave I slave IM
	4	Channel information exists		
	5	User information exists		
	6	Diagnostic interrupt from substitute		
	7	Reserved		
2	0	No or wrong memory card		
	1	Communication problem		
	2	Mode	0 1	RUN STOP
	3	Cycle monitoring responded		
	4	Internal module supply voltage failed		
	5	Battery exhausted		
	6	Entire battery backup failed		
	7	Reserved		
Byte	**Bit**	**Meaning**	**Remarks**	
3	0	Expansion rack failure		
	1	Processor failure		
	2	EPROM error		
	3	RAM error		
	4	ADC/DAC error		
	5	Fuse tripped		
	6	Hardware interrupt lost		
	7	Reserved		
4	0 to 6	Channel type	B#16#70 B#16#72 B#16#71 B#16#73 B#16#74 B#16#75 B#16#76 B#16#77 B#16#78 B#16#79 B#16#7D B#16#7E B#16#7F	Digital input Digital output Analog input Analog output FM-POS FM-REG FM-ZAEHL FM-TECHNO FM-NCU to reserved US300 reserved
	7	Further channel type exists?	0 1	no yes

Byte	Bit	Meaning	Remarks
5	0 to 7	Number of diagnostic bits output per channel by a module.	The number of diagnostic bits per channel is rounded up to byte boundaries
6	0 to 7	Number of channels of one channel type on a module	If different channel types exist on a module, the structure is repeated in data record 1 from byte 4 onwards for each channel type.
7	0	Channel error channel 0/ Channel group 0	
	1	Channel error channel 1/ Channel group 1	
	2	Channel error channel 2/ Channel group 2	
	3	Channel error channel 3/ Channel group 3	First byte of the channel error vector (the length of the channel error vector depends on the number of channels and is rounded up to a byte boundary).
	4	Channel error channel 4/ Channel group 4	
	5	Channel error channel 5/ Channel group 5	
	6	Channel error channel 6/ Channel group 6	
	7	Channel error channel 7/ Channel group 7	
...	-	Channel-specific errors (see Structure of Channel-Specific Diagnostic Data)	

30.3 Structure of Channel-Specific Diagnostic Data

Channel-Specific Errors

Starting at the byte immediately following the channel error vector, the channel-specific errors are indicated for each channel of the module. The tables below show the structure of channel-specific diagnostic data for the different channel types. The bits have the following meaning:

- 1 = Error
- 0 = No error

Analog Input Channel

Diagnostic byte for an analog input channel:

Bit	Meaning	Remarks
0	Configuration/ parameter assignment error	Can be signaled by SFC 52 and EVENTN = W#16#8x50
1	Common mode error	Can be signaled by SFC 52 and EVENTN = W#16#8x51
2	P short circuit	Can be signaled by SFC 52 and EVENTN = W#16#8x52
3	M short circuit	Can be signaled by SFC 52 and EVENTN = W#16#8x53
4	Wire break	Can be signaled by SFC 52 and EVENTN = W#16#8x54
5	Reference channel error	Can be signaled by SFC 52 and EVENTN = W#16#8x55
6	Current below measuring range	Can be signaled by SFC 52 and EVENTN = W#16#8x56
7	Current above measuring range	Can be signaled by SFC 52 and EVENTN = W#16#8x57

Analog Output Channel

Diagnostic byte for an analog output channel:

Bit	Meaning	Remarks
0	Configuration/ parameter assignment error	Can be signaled by SFC 52 and EVENTN = W#16#8x60
1	Common mode error	Can be signaled by SFC 52 and EVENTN = W#16#8x61
2	P short circuit	Can be signaled by SFC 52 and EVENTN = W#16#8x62
3	M short circuit	Can be signaled by SFC 52 and EVENTN = W#16#8x63
4	Wire break	Can be signaled by SFC 52 and EVENTN = W#16#8x64
5	0	Reserved
6	No load voltage	Can be signaled by SFC 52 and EVENTN = W#16#8x66
7	0	Reserved

Digital Input Channel

Diagnostic byte for a digital input channel:

Bit	Meaning	Remarks
0	Configuration/parameter assignment error	Can be signaled by SFC 52 and EVENTN = W#16#8x70
1	Ground error	Can be signaled by SFC 52 and EVENTN = W#16#8x71
2	P short circuit (sensor)	Can be signaled by SFC 52 and EVENTN = W#16#8x72
3	M short circuit	Can be signaled by SFC 52 and EVENTN = W#16#8x73
4	Wire break	Can be signaled by SFC 52 and EVENTN = W#16#8x74
5	No sensor power supply	Can be signaled by SFC 52 and EVENTN = W#16#8x75
6	0	Reserved
7	0	Reserved

Digital Output Channel

Diagnostic byte for a digital output channel:

Bit	Meaning	Remarks
0	Configuration/parameter assignment error	Can be signaled by SFC 52 and EVENTN = W#16#8x80
1	Ground error	Can be signaled by SFC 52 and EVENTN = W#16#8x81
2	P short circuit	Can be signaled by SFC 52 and EVENTN = W#16#8x82
3	M short circuit	Can be signaled by SFC 52 and EVENTN = W#16#8x83
4	Wire break	Can be signaled by SFC 52 and EVENTN = W#16#8x84
5	Fuse tripped	Can be signaled by SFC 52 and EVENTN = W#16#8x86
6	No load voltage	Can be signaled by SFC 52 and EVENTN = W#16#8x86
7	Over temperature	Can be signaled by SFC 52 and EVENTN = W#16#8x87

31 System Status Lists (SSL)

31.1 Overview of the System Status Lists (SSL)

Overview of This Chapter

This chapter describes all the partial lists of the system status list that relate to the following:

- CPUs
- Modules whose partial lists are not module-specific (for example, SSL-IDs W#16#00B1, W#16#00B2, W#16#00B3).

Module-specific partial lists, for example, for CPs and FMs are included in the descriptions of the particular modules.

Definition: System Status List

The system status list (SSL) describes the current status of a programmable logic controller. The contents of the SSL can only be read using information functions but cannot be modified. The partial lists are virtual lists, in other words, they are only created by the operating system of the CPUs when specifically requested.

You can only read one system status list using SFC 51 "RDSYSST."

Contents

The system status lists contain information about the following:

- System data
- Module status data in the CPU
- Diagnostic data on modules
- Diagnostic buffer

System Data

System data are fixed or assigned characteristic data of a CPU. They provide information about the following:

- The configuration of the CPU
- The status of the priority classes
- Communication

Module Status Data

Module status data describe the current status of the components monitored by system diagnostic functions.

Diagnostic Data on Modules

The modules with diagnostic capabilities assigned to a CPU have diagnostic data that are stored directly on the module.

Diagnostic Buffer

The diagnostic buffer contains diagnostic entries in the order in which they occur.

31.2 Structure of a Partial SSL List

Basics

You can read partial lists and partial list extracts using SFC 51 "RDSYSST." You specify what you want to read using the parameters SSL_ID and INDEX.

Structure

A partial list consists of the following:

- A header and
- The data records.

Header

The header of a partial list consists of the following:

- SSL-ID
- Index
- Length of a data record of the partial list in bytes
- Number of data records contained in the partial list.

Index

With certain partial lists or partial list extracts an object type ID or an object number must be specified. The index is used for this purpose. If it is not required for the information, its contents are irrelevant.

Data Records

A data record in a partial list has a specific length. This depends on the information in the partial list. How the data words in a data record are used also depends on the particular partial list.

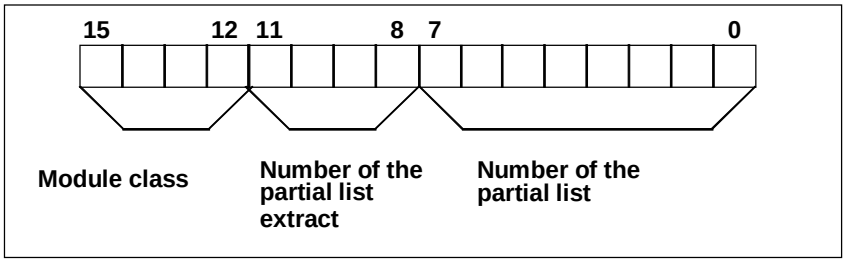
31.3 SSL-ID

SSL-ID

Every partial system status list has a number. You can output a complete partial list or an extract from it. The possible partial list extracts are predefined and are identified by a number. The SSL-ID consists of the number of the partial list, the number of the partial list extract, and the module class.

Structure

The SSL-ID is one word long. The meaning of the bits in the SSL-ID is as follows:



Structure of the SCL_ID

Module Class

Examples of module classes:

Module Class	Coding (Binary)
CPU	0000
IM	0100
FM	1000
CP	1100

Number of the Partial List Extract

The number of the partial list extracts and their meaning depend on the particular system status list to which they belong. With the number of the partial list extract, you specify which subset of a partial list you want to read.

Number of the Partial List

Using the number of the partial list, you specify which partial list of the system status list you want to read.

31.4 Possible Partial System Status Lists

Subset

Any one module only has a subset of all the possible partial lists. Which partial lists are available depends on the particular module.

Possible SSL Partial Lists

The following table lists all the possible partial lists with the number contained in the SSL-ID.

Partial List	SSL-ID
Module identification	W#16#xy11
CPU characteristics	W#16#xy12
User memory areas	W#16#xy13
System areas	W#16#xy14
Block types	W#16#xy15
Status of the module LEDs	W#16#xy19
Interrupt status	W#16#xy22
Communication status data	W#16#xy32
H CPU group information	W#16#xy71
Status of the module LEDs	W#16#xy74
Switched DP slaves in the H-system	W#16#xy75
Module status information	W#16#xy91
Rack / station status information	W#16#xy92
Diagnostic buffer of the CPU	W#16#xyA0
Module diagnostic information (data record 0)	W#16#00B1
Module diagnostic information (data record 1), geographical address	W#16#00B2
Module diagnostic information (data record 1), logical address	W#16#00B3
Diagnostic data of a DP slave	W#16#00B4

31.5 SSL-ID W#16#xy11 - Module Identification

Purpose

If you read the system status list with SSL-ID W#16#xy11, you obtain the module identification of this module.

Header

The header of system status list SSL-ID W#16#xy11 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract W#16#0111: a single identification data record
INDEX	Number of a particular data record W#16#0001: identification of the module W#16#0006: identification of the basic hardware W#16#0007: identification of the basic firmware
LENTHDR	W#16#001C: one data record is 14 words long (28 bytes)
N_DR	Number of data records

Data Record

A data record of system status list SSL-ID W#16#xy11 has the following structure:

Name	Length	Meaning
Index	1 word	Index of an identification data record
MlFB	20 bytes	With INDEX W#16#0007: reserved With INDEX W#16#0001 and W#16#0006: Order number of the module; String consists of 19 characters and a blank (20H); such as for CPU 314: "6ES7 314-0AE01-0AB0"
BGTyp	1 word	Reserved
Ausbg1	1 word	With Index W#16#0001: version of the module With Index W#16#0006 and W#16#0007:"V" and first number of the version ID
Ausbg2	1 word	With Index W#16#0001: reserved With Index W#16#0006 and W#16#0007: remaining numbers of the version ID

31.6 SSL-ID W#16#xy12 - CPU Characteristics

Purpose

CPU modules have different characteristics depending on the hardware being used. Each characteristic is assigned an ID. If you read the partial list with SSL-ID W#16#xy12, you obtain the characteristics of the module.

Header

The header of partial list SSL-ID W#16#xy12 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract: W#16#0012: all characteristics W#16#0112: characteristics of a group You specify the group in the INDEX parameter. W#16#0F12: partial list header information
INDEX	Group W#16#0000: MC7 processing unit W#16#0100: time system W#16#0200: system response W#16#0300: MC7 language description of the CPU W#16#0400: availability of SFCs
LENTHDR	W#16#0002: one data record is 1 word long (2 bytes)
N_DR	Number of data records

Data Record

A data record of partial list SSL-ID W#16#xy12 is one word long. An identifier is entered for each characteristic. A characteristics identifier is one word long.

Characteristics Identifier

The following table lists all the characteristics identifiers.

Identifier	Meaning
W#16#0000 - 00FF	MC7 processing unit (group with index 0000)
W#16#0001	MC7 processing generating code
W#16#0002	MC7 interpreter
W#16#0100 - 01FF	Time system (group with index 0100)
W#16#0101	1 ms resolution
W#16#0102	10 ms resolution
W#16#0103	No real time clock
W#16#0104	BCD time-of-day format
W#16#0200 - 02FF	System response (group with index 0200)
W#16#0201	Capable of multiprocessor mode

Identifier	Meaning
W#16#202	Cold restart, warm restart and hot restart possible
W#16#203	Cold restart and hot restart possible
W#16#204	Warm restart and hot restart possible
W#16#205	Only warm restart possible
W#16#0300 - 03FF	MC7 Language description of the CPU (group with index 0300)
W#16#0301	Reserved
W#16#0302	All 32 bit fixed-point instructions
W#16#0303	All floating-point instructions
W#16#0304	sin, asin, cos, acos, tan, atan, sqr, sqrt, ln, exp
W#16#0305	Accumulator 3/accumulator 4 with corresponding instructions (ENT,PUSH,POP,LEAVE)
W#16#0306	Master Control Relay instructions
W#16#0307	Address register 1 exists with corresponding instructions
W#16#0308	Address register 2 exists with corresponding instructions
W#16#0309	Operations for area-crossing addressing
W#16#030A	Operations for area-internal addressing
W#16#030B	All memory-indirect addressing instructions for bit memory (M)
W#16#030C	All memory-indirect addressing instructions for data blocks (DB)
W#16#030D	All memory-indirect addressing instructions for data blocks (DI)
W#16#030E	All memory-indirect addressing instructions for local data (L)
W#16#030F	All instructions for parameter transfer in FCs
W#16#0310	Memory bit edge instructions for process image input (I)
W#16#0311	Memory bit edge instructions for process image output (Q)
W#16#0312	Memory bit edge instructions for bit memory (M)
W#16#0313	Memory bit edge instructions for data blocks (DB)
W#16#0314	Memory bit edge instructions for data blocks (DI)
W#16#0315	Memory bit edge instructions for local data (L)
W#16#0316	Dynamic evaluation of the FC bit
W#16#0317	Dynamic local data area with the corresponding instructions
W#16#0318	Reserved
W#16#0319	Reserved
W#16#0401	SFC 87 "C_DIAG" is available
W#16#0402	SFC 88 "C_CNTRL" is available

31.7 SSL-ID W#16#xy13 - Memory Areas

Purpose

If you read the partial list with SSL-ID W#16#xy13, you obtain information about the memory areas of the module.

Header

The header of partial list SSL-ID W#16#xy13 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract W#16#0113: data record for one memory area You specify the memory area with the INDEX parameter.
INDEX	Specifies a memory area (only with SSL-ID W#16#0113) W#16#0001: work memory
LENTHDR	W#16#0024: one data record is 18 words long (36 bytes)
N_DR	Number of data records

Data Record

A data record of partial list SSL-ID W#16#xy13 has the following structure:

Name	Length	Meaning
Index	1 word	Index of a memory area W#16#0001: work memory
Code	1 word	Memory type: W#16#0001: volatile memory (RAM) W#16#0002: non-volatile memory (FEPROM) W#16#0003: mixed memory (RAM + FEPROM)
Size	2 words	Total size of the selected memory (total of area 1 and area 2)
Mode	1 word	Logical mode of the memory Bit 0: volatile memory area Bit 1: non-volatile memory area Bit 2: mixed memory area For work memory: Bit 3: code and data separate Bit 4: code and data together
Granu	1 word	Always has the value 0
Ber1	2 words	Size of the volatile memory area in bytes.
Belegt1	2 words	Size of the volatile memory area being used
Block1	2 words	Largest free block in the volatile memory area If 0: no information available or cannot be ascertained.
Ber2	2 words	Size of the non-volatile memory area in bytes
Belegt2	2 words	Size of the non-volatile memory area being used
Block2	2 words	Largest free block in the non-volatile memory area

Name	Length	Meaning
		If 0: no information available or cannot be ascertained.

31.8 SSL-ID W#16#xy14 - System Areas

Purpose

If you read the partial list with SSL-ID W#16#xy14, you obtain information about the system areas of the module.

Header

The header of partial list SSL-ID W#16#xy14 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract W#16#0014: all system areas of a module W#16#0F14: only for partial list header information
INDEX	Not relevant
LENTHDR	W#16#0008: one data record is 4 words long (8 bytes)
N_DR	Number of data records You must at least assign a number of 9 data records. If you select a target area which is too small, SFC51 does not provide a data record.

Data Record

A data record of partial list SSL-ID W#16#xy14 has the following structure:

Name	Length	Meaning
Index	1 word	Index of the system area W#16#0001: PII (number in bytes) W#16#0002: PIQ (number in bytes) W#16#0003: memory (number in bits) **Note:** This index is only provided by the CPU, where the number of flags can be shown in one word. If your CPU does not provide this value, you must evaluate index W#16#0008. W#16#0004: timers (number) W#16#0005: counters (number) W#16#0006: number of bytes in the logical address area W#16#0007: local data (entire local data area of the CPU in bytes) **Note:** This index is only provided by the CPU, where the number of flags can be shown in one word. If your CPU does not provide this value, you must evil index W#16#0009. W#16#0008: memory (number in bytes) W#16#0009: local data (entire local data area of the CPU in Kbytes)
Code	1 word	Memory type

Name	Length	Meaning
		W#16#0001: volatile memory (RAM) W#16#0002: non-volatile memory (FEPROM) W#16#0003: mixed memory (RAM and FEPROM)
Quantity	1 word	Number of elements of the system area
Reman	1 word	Number of retentive elements

31.9 SSL-ID W#16#xy15 - Block Types

Purpose

If you read the partial list with SSL-ID W#16#xy15, you obtain the block types that exist on the module.

Header

The header of partial list SSL-ID W#16#xy15 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract W#16#0015: Data records of all block types of a module
INDEX	Not relevant
LENTHDR	W#16#0006: one data record is 5 words long (10 bytes)
N_DR	Number of data records

Data Record

A data record of partial list SSL-ID W#16#xy15 has the following structure:

Name	Length	Meaning
Index	1 word	Block type number W#16#0800: OB W#16#0A00: DB W#16#0B00: SDB W#16#0C00: FC W#16#0E00: FB
MaxAnz	1 word	Maximum number of blocks of the type OBs: max. possible number of OBs for a CPU DBs: max. possible number of DBs including DB0 SDBs: max. possible number of SDBs including SDB2 FCs and FBs: max. possible number of loadable blocks
MaxLng	1 word	Maximum total size of the object to be loaded in Kbytes
Maxabl	2 words	Maximum length of the work memory part of a block in bytes

31.10 SSL-ID W#16#xy19 - Status of the Module LEDs

Purpose

If you read the partial list with SSL-ID W#16#xy19, you obtain the status of the module LEDs.

Note

If you want to read out the partial list W#16#16#xy19 for an H CPU, remember that this is only possible in the non-redundant H operating modes.

Header

The header of partial list W#16#xy19 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial system status list W#16#0019 Status of all LEDs W#16#0119 Status of one LED
LENTHDR	W#16#0004: one data record is 2 words long (4 bytes)
N_DR	Number of data records

Data Record

A data record of the partial list with SSL-ID W#16#xy19 has the following structure:

Name	Length	Meaning
Index	1 word	LED ID (only relevant for SSL-ID W#16#0119) W#16#0001: SF (group error) W#16#0002: INTF (internal error) W#16#0003: EXTF (external error) W#16#0004: RUN W#16#0005: STOP W#16#0006: FRCE (force) W#16#0007: CRST (restart) W#16#0008: BAF (battery fault/overload, short circuit of battery voltage on bus) W#16#0009: USR (user-defined) W#16#000A: USR1 (user-defined) W#16#000B: BUS1F (bus error interface 1) W#16#000C: BUS2F (bus error interface 2) W#16#000D: REDF (redundancy error) W#16#000E: MSTR (master) W#16#000F: RACK0 (rack number 0) W#16#0010: RACK1 (rack number 1) W#16#0011: RACK2 (rack number 2) W#16#0012: IFM1F (interface error interface module 1) W#16#0013: IFM2F (interface error interface module 2)
led_on	1 byte	Status of the LED: 0 : off 1 : on
led_blink	1 byte	Flashing status of the LED: 0: not flashing 1: flashing normally (2 Hz) 2: flashing slowly (0.5 Hz)

31.11 SZL-ID W#16#xy1C - Component Identification

Purpose

If you read the partial list with SSL-ID W#16#xy1C, you can identify the CPU or the PLC.

Header

The header of partial list W#16#xy1C is structured as follows:

Contents	Meaning	
SSL-ID	The SSL-ID of the partial list extract	
	W#16#001C:	Identification of all components
	W#16#011C:	Identification of one component
	W#16#0F1C:	SSL partial list header information only
INDEX	Identification of the component for the partial system status list with the SSL ID W#16#011C	
	W#16#0001:	Name of the PLC
	W#16#0002:	Name of the CPU
	W#16#0003:	Plant identification of the CPU
	W#16#0004:	Copyright entry
	W#16#0006:	Reserved for operating system
LENTHDR	W#16#0022:	A data record is 17 words long (34 bytes)
N_DR	Number of data records	

Data Record

A data record of the partial list with SSL-ID W#16#xy1C has the following structure:

- INDEX = W#16#0001

Name	Length	Meaning
index	1 word	Component identification: W#16#0001
name	12 words	Name of the PLC (max. 24 characters; when using shorter names, the gaps are filled with B#16#00)
res	4 words	reserved

- INDEX = W#16#0002

Name	Length	Meaning
index	1 word	Component identification: W#16#0002
name	12 words	Name of the CPU (max. 24 characters; when using shorter names, the gaps are filled with B#16#00)
res	4 words	reserved

- INDEX = W#16#0003

Name	Length	Meaning
index	1 word	Component identification: W#16#0003
tag	16 words	Plant identification of the CPU (max. 32 characters; when using a shorter plant identification the gaps are filled with B#16#00)

- INDEX = W#16#0004

Name	Length	Meaning
index	1 word	Component identification: W#16#0004
copyright	13 words	Constant character sequence "Original Siemens Equipment"
res	3 words	reserved

- INDEX = W#16#0006

  The corresponding data record is reserved for the operating system.

31.12 SSL-ID W#16#xy22 - Interrupt Status

Purpose

If you read the partial list with SSL-ID W#16#xy22, you obtain information about the current status of interrupt processing and the interrupts generated by the module.

Header

The header of partial list SSL-ID W#16#xy22 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial system status list W#16#0222 Data record for the specified interrupt. You specify the interrupt (OB number) using the INDEX parameter.
INDEX	Priority class or OB number (with SSL-ID W#16#0222) W#16#0000: free cycle W#16#0A0A: time-of-day interrupt W#16#1414: time-delay interrupt W#16#1E23: cyclic interrupt W#16#2828: hardware interrupt W#16#5050: asynchronous error interrupt W#16#005A: background W#16#0064: startup W#16#7878: synchronous error interrupt
LENTHDR	W#16#001C: one data record is 14 words long (28 bytes)
N_DR	Number of data records

Data Record

A data record of partial list SSL-ID W#16#xy22 has the following structure:

Name	Length	Meaning
Info	10 words	Start information for the corresponding OB, with the following exceptions: • For OB1, the current minimum (in byte 8 and 9) and maximum (in byte 10 and 11) cycle times are available (time base: ms, byte count-up starts with 0).. • For OB80, the configured minimum and maximum cycle (in byte 10 and 11) times can be read. • For error interrupts without the current information • For interrupts, the status information contains the current parameter assignment of the source of the interrupt. • For synchronous errors, B#16#7F is entered as the priority class if the OBs have not yet been executed, otherwise the priority class of the last call. • If an OB has several start events and if these have not occurred when the information is fetched, event number W#16#xyzz is returned with x: event class, zz: lowest defined number of group, y: undefined. Otherwise the number of the last start event to occur is used.
al 1	1 word	Processing identifier Bit 0: Interrupt event disabled/enabled by parameter assignment = 0: enabled = 1: disabled Bit 1: Interrupt event disabled by SFC 39 "DIS_IRT" = 0: not disabled = 1: disabled Bit 2 = 1:Interrupt source is active (generate job exists for time interrupts, time-of-day interrupt OB started, time-delay interrupt OB started, cyclic interrupt OB: time started) Bit 4: Interrupt OB = 0: not loaded = 1: loaded Bit 5: Interrupt OB disabled by test and installation function = 1: disabled Bit 6: entry into the diagnostic buffer = 1: disabled
al 2	1 word	Reaction if OB not loaded/disabled Bit 0 = 1:Disable interrupt source Bit 1 = 1:Generate interrupt event error Bit 2 = 1:CPU changes to the STOP mode Bit 3 = 1:Only discard interrupt
al 3	2 words	Discard by test and installation functions: Bit number x set means: the event number that is x higher than the lowest event number of the corresponding OB is discarded by the test and installation function.

31.13 SSL-ID W#16#xy32 - Communication Status Data

Purpose

If you read the partial list with SSL-ID W#16#xy32 you obtain the status data of module communication.

Header

The header of partial list SSL-ID W#16#xy32 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract W#16#0132 Status data for one communication section of the CPU (always one data record). You specify the communication section of the CPU with the INDEX parameter. W#16#0232 Status data for one communication section (in an H system in the RUN-REDUNDANT mode, n data records are returned, where n is the number of redundant CPUs in the H system). You specify the communication section of the CPU with the INDEX parameter.
INDEX	Communication section of the CPU: • For SSL-ID W#16#0132: W#16#0004 CPU protection level, operator switch and version identification W#16#0005 Diagnostics W#16#0008 Time system • For SSL-ID W#16#0232: W#16#0004 CPU protection level and operator control settings
LENTHDR	W#16#00028: one data record is 20 words long (40 bytes)
N_DR	Number of data records

Data Record

A data record of partial list SSL-ID W#16#0132 is always 20 words long. The data records have different contents. The contents depend on the INDEX parameter, in other words, on the communication section of the CPU to which the data record belongs.

31.14 Data Record of the Partial List Extract with SSL-ID W#16#0132 Index W#16#0005

Contents

The partial list extract with SSL-ID W#16#0132 and index W#16#0005 contains information about the status of the diagnostics on the module.

Data Record

A data record of partial list extract SSL-ID W#16#0132 with index W#16#0005 has the following structure:

Name	Length	Meaning
Index	1 word	W#16#0005: Diagnostics
Erw	1 word	Extended functions 0: no 1: yes
Send	1 word	Automatic sending 0: no 1: yes
Moeg	1 word	Sending user-defined diagnostic messages currently possible 0: no 1: yes
Res	16 words	Reserved

31.15 Data Record of the Partial List Extract with SSL-ID W#16#0132 Index W#16#0008

Contents

The partial list extract with SSL-ID W#16#0132 and index W#16#0008 contains information about the status of the time system on the module.

Data Record

A data record of partial list extract SSL-ID W#16#01032 with index W#16#0008 has the following structure:

Name	Length	Meaning
Index	1 word	W#16#0008: Time system status
Zykl	1 word	Cycle time of the synchronization frames
Korr	1 word	Correction factor for the time
clock 0	1 word	Run-time meter 0: time in hours
clock 1	1 word	Run-time meter 1: time in hours
clock 2	1 word	Run-time meter 2: time in hours
clock 3	1 word	Run-time meter 3: time in hours
clock 4	1 word	Run-time meter 4: time in hours
clock 5	1 word	Run-time meter 5: time in hours
clock 6	1 word	Run-time meter 6: time in hours
clock 7	1 word	Run-time meter 7: time in hours
Time	4 words	Current date and time (format: DATE_AND_TIME)
bszl_0 to bszl_1	2 bytes	Run-time meter active (bit =1: run-time meter active)
bszl_0	1 byte	Bit x: run-time meter x, $0 \leq x \leq 7$
bszl_1	1 byte	Reserved
bszü_0 to bszü_1	2 bytes	Run-time meter overflow (bit = 1: overflow)
bszü_0	1 byte	Bit x: run-time meter x, $0 \leq x \leq 7$
bszü_1	1 byte	Reserved
Status	1 word	Time status (for bit assignment, see below)
Res	3 byte	Reserved
status_valid	1 byte	Validity of variable status: B#16#01: status valid

status

Bit	Default Value	Description
15	0	Sign for the correction value (0: positive, 1: negative)
14 to 10	00000	Correction value This parameter allows the basic time in the frame to be corrected to local time: Local time = basic time ± correction value * 0.5 h This correction takes into account the time zone and the time difference in summer time (daylight savings time) and winter time (standard time).
9	0	Reserved
8	0	Reserved
7	0	Notification hour This parameter indicates whether the next time adjustment also includes a switchover from summer (daylight savings time) to winter time (standard time) or vice versa. (0: no adjustment made, 1: adjustment made).
6	0	Summer (daylight savings time)/winter time (standard time) indicator The parameter indicates whether the local time calculated using the correction value is summer or winter time. (0: winter time, 1: summer time)
5	0	Parameter not used by S7.
4 to 3	00	Time resolution This parameter indicates the resolution of the transmitted clock time. (00: 0.001 s, 01: 0.01 s, 10: 0.1 s, 11: 1 s)
2	0	Parameter not used by S7.
1	0	Parameter not used by S7.
0	0	Synchronization failure This parameter indicates whether the time transmitted in the frame. (0: synchronization failed, 1: synchronization occurred) Note: Evaluation of this bit in a CPU is only meaningful if there is continuous external time synchronization.

31.16 Data Record of the Partial List Extract with SSL-ID W#16#0232 Index W#16#0004

Contents

The partial list extract with SSL-ID W#16#0232 and index W#16#0004 contains information about the CPU protection level and the settings of the operator mode switch and version identifications of the hardware configuration and the user program.

In an H system in the RUN-REDUNDANT mode, one data record per redundant CPU is returned.

Data Record

A data record of partial list extract SSL-ID W#16#0232 and index W#16#0004 has the following structure:

Name	Length	Meaning
Index	1 word	• Byte 1: B#16#04: CPU protection level and operator control settings and version identifications • Byte 0: Standard CPU: B#16#00 H CPU: Bits 0 to 2: rack number Bit 3: 0 = standby CPU, 1 = master CPU Bits 4 to 7: 1111
sch_schal	1 word	Protection level set with the mode selector (1, 2, 3)
sch_par	1 word	Protection level set in parameters (0, 1, 2, 3; 0: no password, protection level invalid)
sch_rel	1 word	Valid protection level of the CPU
bart_sch	1 word	Mode selector setting (1:RUN, 2:RUN-P, 3:STOP, 4:MRES, 0:undefined or cannot be determined)
anl_sch	1 word	Startup switch setting (1:CRST, 2:WRST, 0:undefined, does not exist of cannot be determined)
ken_rel	1 word	ID for valid version identification (0: invalid)
ken_ver1_hw	1 word	Version ID 1 of the hardware configuration
ken_ver2_hw	1 word	Version ID 2 of the hardware configuration
ken_ver1_awp	1 word	Version ID 1 of the user program
ken_ver2_awp	1 word	Version ID 2 of the user program
Res	8 words	reserved

31.17 SSL-ID W#16#xy71 - H CPU Group Information

Purpose

The partial list extract with SSL-ID W#16#xy71 contains information about the current status of the H system.

Header

The header of partial list SSL-ID W#16#xy71 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract: W#16#0071: Information about the current status of the H system W#16#0F71: Only SSL partial list header information
INDEX	W#16#0000
LENTHDR	W#16#0010: One data record is 8 words long (16 bytes)
N_DR	W#16#0001: Number of data records

Data Record

A data record of partial list extract ID W#16#xy71 has the following structure:

Contents	Length	Meaning
Redinf	2 bytes	Information about redundancy W#16#0011: Single H CPU W#16#0012: 1 of 2 H system
Mwstat1	1 byte	Status byte 1 Bit 0: reserved Bit 1: reserved Bit 2: reserved Bit 3: reserved Bit 4: H status of CPU in rack 0 =0: standby CPU =1: master CPU Bit 5: H status of CPU in rack 1 =0: standby CPU =1: master CPU Bit 6: reserved Bit 7: reserved

Contents	Length	Meaning
Mwstat2	1 byte	Status byte 2 Bit 0: Status of the synchronization link-up 01: Synchronization between CPU 0 and CPU 1 =0: not possible =1: possible Bit 1: 0 Bit 2: 0 Bit 3: reserved Bit 4: =0: CPU not inserted in rack 0 =1: CPU inserted in rack 0 (in redundant mode: bit 4 = 0) Bit 5: =0: CPU not inserted in rack 1 =1: CPU inserted in rack 1 (in redundant mode: bit 5 = 0) Bit 6: reserved Bit 7: Standby-master switchover since last Depassivation =0: no =1: yes
Hsfcinfo	2 bytes	Info word for SFC 90 "H_CTRL" Bit 0: =0: Depassivation inactive =1: Depassivation active Bit 1: =0: Updating of standby enabled =1: Updating of standby disabled Bit 2: =0: Link-up to standby enabled =1: Link-up to standby disabled Bit 3: reserved Bit 4: reserved Bit 5: reserved Bit 6: =reserved Bit 7: =1: Upgrade with updating requested Bit 8: =1: Upgrade without updating requested
Samfehl	2 bytes	Reserved

Contents	Length	Meaning
Bz_cpu_0	2 bytes	Mode of CPU in rack 0 W#16#0001: STOP (update) W#16#0002: STOP (reset memory) W#16#0003: STOP (self-initialization) W#16#0004: STOP (internal) W#16#0005: STARTUP (cold restart) W#16#0006: STARTUP (warm restart) W#16#0007: STARTUP (hot restart) W#16#0008: RUN (solo mode) W#16#0009: RUN-R (redundant mode) W#16#000A: HOLD W#16#000B: LINK-UP W#16#000C: UPDATE W#16#000D: DEFECTIVE W#16#000E: SELFTEST W#16#000F: NO POWER
Bz_cpu_1	2 bytes	Mode of CPU in rack 1 (values as for bz_cpu_0)
Bz_cpu_2	2 bytes	Reserved
Cpu_valid	1 byte	Validity of the variables bz_cpu_0 and bz_cpu_1 B#16#01: bz_cpu_0 valid B#16#02: bz_cpu_1 valid B#16#03: bz_cpu_0 and bz_cpu_1 valid
Reserve	1 byte	Reserved

31.18 SSL-ID W#16#xy74 - Status of the Module LEDs

Purpose

If you read the partial list SSL-ID W#16#xy74, with standard CPUs (if present) and with the H CPUs, you obtain the status of the module LEDs.

If the H CPUs are in a non-redundant H mode, you obtain the LED status of the CPU addressed. If the H CPUs are in the RUN-REDUNDANT mode, the LED status of all redundant H CPUs is returned.

Header

The header of partial list SSL-ID W#16#xy74 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract W#16#0174 Status of an LED. You select the LED with the INDEX parameter.
INDEX	LED ID (only relevant for SSL-ID W#16#0174) W#16#0001: SF (group error) W#16#0002: INTF (internal error) W#16#0003: EXTF (external error) W#16#0004: RUN W#16#0005: STOP W#16#0006: FRCE (force) W#16#0007: CRST (cold restart) W#16#0008: BAF (battery fault/overload, short circuit of battery voltage on bus) W#16#0009: USR (user-defined) W#16#000A: USR1 (user-defined) W#16#000B: BUS1F (bus error interface 1) W#16#000C: BUS2F (bus error interface 2) W#16#000D: REDF (redundancy error) W#16#000E: MSTR (master) W#16#000F: RACK0 (rack number 0) W#16#0010: RACK1 (rack number 1) W#16#0011: RACK2 (rack number 2) W#16#0012: IFM1F (interface error interface module 1) W#16#0013: IFM2F (interface error interface module 2)
LENTHDR	W#16#0004: one data record is 2 words long (4 bytes)
N_DR	Number of data records

Data Record

A data record of partial list extract SSL-ID W#16#0074 has the following structure:

Name	Length	Meaning
cpu_led_ID	1 word	• Byte 0 - - Standard CPU: B#16#00 - - H-CPU: Bits 0 to 2: rack number Bit 3: 0=standby CPU, 1=master CPU Bits 4 to 7: 1111 Byte 1: LED ID
led_on	1 byte	Status of the LED: 0: off 1: on
led_blink	1 byte	Flashing status of the LED: 0: not flashing 1: flashing normally (2 Hz) 2: flashing slowly (0.5 Hz))

31.19 SSL-ID W#16#xy75 - Switched DP Slaves in the H System

Purpose

If you read the partial list SSL-ID W#16#xy75, with CPUs of an H system in a redundant H operating mode, you obtain the status information on the communication between the H system and the switched DP slaves.

The partial list tells you in which rack the DP master system interface module currently being used for communication with a DP slave is inserted.

Header

The header of partial list SSL-ID W#16#xy75 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract W#16#0C75: Communication status between the H system and a switched DP slave. You select the DP slave with the INDEX parameter.
INDEX	Diagnostic address of the DP slave interface module(s)
LENTHDR	W#16#0010: One data record is 8 words long (16 bytes)
N_DR	W#16#0001: Number of data records

Data Record

A data record of partial list ID SSL-ID W#16#xy75 has the following structure:

Name	Length	Meaning
adr1_bgt0	1 word	First address section of the DP slave interface module whose DP master interface module is inserted in rack 0: DP master system ID and station number
adr2_bgt0	1 word	Second address section of the DP slave interface module whose DP master interface module is inserted in rack 0: Slot and submodule slot
adr1_bgt1	1 word	First address section of the DP slave interface module whose DP master interface module is inserted in rack 1: DP master system ID and station number
adr2_bgt1	1 word	Second address section of the DP slave interface module whose DP master interface module is inserted in rack 1: Slot and submodule slot
Res	2 words	Reserved
Logadr	1 word	Diagnostic address of the DP slave interface module(s): • Bits 0 to 14: logical base address • Bit 15: I/O identifier (0 = input, 1 = output)
Slavestatus	1 word	Communication status: • Bit 0 = 1: No access to DP the slave interface module whose DP master interface module is inserted in rack 0 • Bit 1 = 1: No access to DP the slave interface module whose DP master interface module is inserted in rack 1 • Bits 2 to 7: Reserved (each = 0) • Bit 8 = 1: Both communication channels functioning properly; communication currently taking place via the DP master interface module in rack 0 • Bit 9 = 1: Both communication channels functioning properly; communication currently taking place via the DP master interface module in rack 1 • Bits 10 to 15: Reserved (each = 0)

31.20 SZL-ID W#16#xy90 - DP Master System Information

Purpose

If you read the partial list SSL-ID W#16#xy90, you obtain the status information of all DP master systems known to the CPU.

Header

The header of partial list SSL-ID W#16#xy90 is structured as follows:

Contents	Meaning	
SSL-ID	The SSL-ID of the partial list extract	
	W#16#0090:	Information of all DP master systems known to the CPU
	W#16#0190:	Information of one DP master system
	W#16#0F90:	SSL partial list header information only
INDEX	• For the partial list extract with the SSL-ID W#16#0190: Low Byte: B#16#00 High Byte: DP master system ID • For the partial list extracts with the SSL-IDs W#16#0090 and W#16#0F90: W#16#0000	
LENTHDR	W#16#000E:	A data record is 7 words long (14 bytes)
N_DR	Number of data records • For the partial list extract with the SSL-ID W#16#0190: 0 to 1 • For the partial list extract with the SSL-ID W#16#0090: • with a standard CPU: 0 to 14 • with a H system: 0 to 12 (in all system states except redundant) 0 to 2 x 12 (in redundant system state)	

Data Record

A data record of partial list ID W#16#xy90 has the following structure:

Name	Length	Meaning	
dp_m_id	1 byte	DP master system ID	
rack_dp_m	1 byte	Rack number of the DP master • with a standard CPU: 0 • with a H system: 0 or 1	
steckpl_dp_m	1 byte	Slot of the DP master or slot of the CPU (with integrated DP interface)	
subm_dp_m	1 byte	• with integrated DP interface: interface number of the DP master: • 1: X2 • 2: X1 • 3: IF1 • 4: IF2 • with external DP interface: 0	
logadr	1 word	logic start address of the DP master	
dp_m_sys_cpu	1 word	reserved	
dp_m_sys_dpm	1 word	reserved	
dp_m_state	1 byte	further properties of the DP master system	
		Bit 0:	DP mode • 0: S7 compatible • 1: DPV1
		Bit 1 to 6:	reserved
		Bit 7:	DP master type • 0: integrated DP master • 1: external DP master
reserve	3 bytes	reserved	

31.21 SSL-ID W#16#xy91 - Module Status Information

Purpose

If you read the partial list SSL-ID W#16#xy91, you obtain the status information of modules assigned to the CPU.

Header

The header of partial list SSL-ID W#16#xy91 is structured as follows:

Contents	Meaning	
SSL-ID	The SSL-ID of the partial list extract	
	W#16#0091	Module status information of all plugged in modules and submodules (S7-400 only)
	W#16#0191	Status information of all modules and racks with wrong type identifier (S7-400 only)
	W#16#0291	Module status information of all faulty modules (S7-400 only)
	W#16#0391	Module status information of all unavailable modules (S7-400 only)
	W#16#0591	Module status information of all submodules of the host module
	W#16#0991	Module status information of a DP master system
	W#16#0A91	Module status information of all DP submodules and DP master systems (S7-300 only, not CPU 318-2 DP)
	W#16#0C91	Module status information of a module in the central rack or of an integrated DP interface via the logical base address
	W#16#4C91	Module status information of a module connected to an external DP interface via the logical base address
		If you use more than four external DP interfaces the result by mistake can be RET_VAL W#16#80A4.
	W#16#0D91	Module status information of all modules in the specified rack/in the specified station (DP)
	W#16#0E91	Module status information of all configured modules
	W#16#0E91	Module status information of all configured modules
	W#16#0F91	Module status information of all configured modules
	W#16#0F91	Number of data records with module status information for all inserted modules/submodules

Contents	Meaning
INDEX	• For the partial list extract with SSL-ID W#16#0C91: - S7-400: bit 0 to 14: logical base address of the module bit 15: 0 = input, 1 = output - S7-300: module start address • For the partial list extract with SSL-ID W#16#4C91 (S7-400 only): Bits 0 to 14 : logical base address of the module Bit 15 : 0 = input, 1 = output • For the partial list extract with SSL-IDs W#16#0091, W#16#0191,W#16#0291, W#16#0391, W#16#0491, W#16#0591, W#16#0A91, W#16#0E91,W#16#0F91: INDEX is irrelevant, all modules (in the rack and in the distributed I/Os) For the partial list extract with SSL-Ids W#16#0991 and W#16#0D91: W#16#00xx all modules and submodules of a rack (xx contains the number of the rack) W#16#xx00 all modules of a DP master system (xx contains the DP master system ID) W#16#xxyy all modules of a DP station (xx contains the DP master system ID, yy station number)
LENTHDR	W#16#0010: One data record is 8 words long (16 bytes)
N_DR	Number of data records. Depending on the product the number of records transferred in the SFC 51 can be lower

In the case of W#16#0091, W#16#0191 and W#16#0F91 two additional data records are supplied per rack:

- A record for the power supply in as far as it exists and has been planned and
- A record for the rack.
- The sequence of the records in case of a centralized structure is: PS, Slot 1, Slot 2, ..., Slot 18, rack.

A data record of partial list ID W#16#xy91 has the following structure:

Name	Length	Meaning
adr1	1 word	Number of the rack (DP master system ID and station number with DP) of the geographical address
adr2	1 word	Slot and submodule slot
logadr	1 word	First assigned logical I/O address (base address)
solltyp	1 word	Reserved
isttyp	1 word	Reserved
alarm	1 word	Reserved (00xx=CPU No. 1-4)

Name	Length	Meaning
eastat	1 word	I/O status Bit 0 = 1: module fault (detected by diagnostic interrupt) Bit 1 = 1: module exists Bit 2 = 1: module does not exist (detected as access error) Bit 5 = 1: module can be host module for submodule Bit 6 = 1: reserved for S7-400 Bit 7 = 1: module on local bus segment Bit 8 to 15: data ID for logical address (input: B#16#B4, output: B#16#B5, external DP interface: B#16#FF)
ber_bgbr	1 word	Area ID/module width Bit 0 to bit 2 : module width Bit 3: reserved Bit 4 to bit 6 : area ID 0 = S7-400 1 = S7-300 2 = ET area 3 = P area 4 = Q area 5 = IM3 area 6 = IM4 area Bit 7: reserved

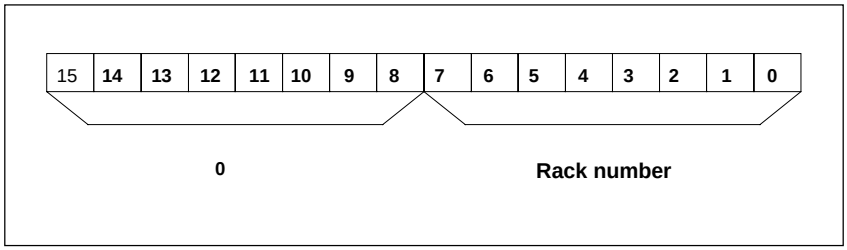
At certain modules the following values are indicated in the record:

Name	PS (only S7-400)	CPU	IFM-CPU (S7-300)	Rack (only S7-400)
adr1	Number of the rack	Standard information as described above	Standard information as described above	Number of the rack
adr2	W#16#01FF	W#16#0200 or W#16#0200 to W#16#1800	W#16#0200	W#16#00FF
logadr	W#16#0000	W#16#7FFF	W#16#007C	W#16#0000
solltyp	Standard information as described above	W#16#00C0 or W#16#0081 or W#16#0082	W#16#00C0	Standard information as described above
eastat	W#16#0000	Standard information as described above	Standard information as described above	W#16#0000
ber_bgbr	W#16#0000	W#16#0011 or W#16#0001 or W#16#0002	W#16#0011	W#16#0000

adr1

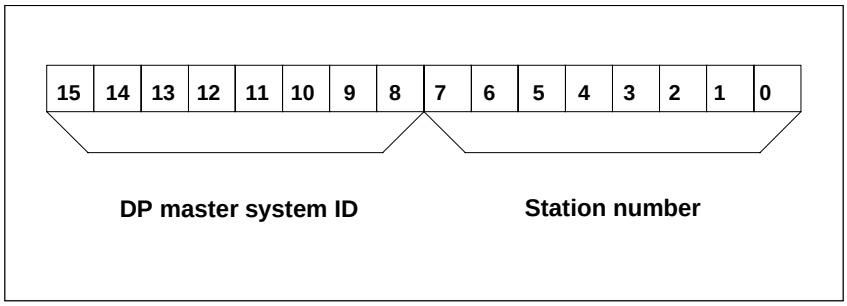
The parameter adr1 contains the following:

- when installed centrally, the rack number.



Bits of the parameter adr1 when installed centrally

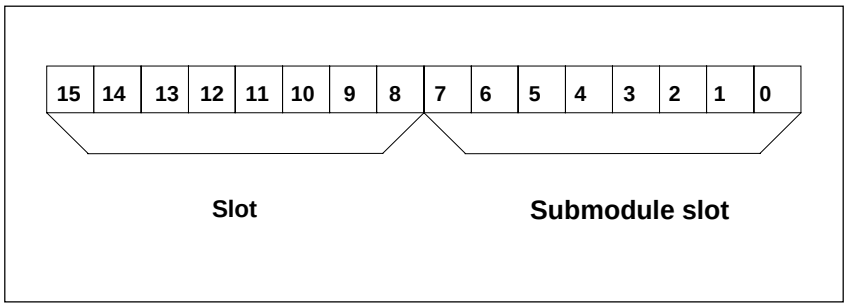
- with a distributed configuration
  the DP master system ID
  the station number.



Bits of the parameter adr1 in a distributed configuration.

adr2

The parameter adr2 contains the slot and submodule slot.



Bits of the parameter adr2.

Information on Multicomputing (only S7-400)

All the partial lists only supply information on the modules which are assigned to a CPU. In multicomputing mode you must therefore sample all the CPUs in order to obtain the data of all the connected modules

31.22 SSL-ID W#16#xy92 - Rack / Station Status Information

Purpose

If you read the partial list SSL-ID W#16#xy92, you obtain information about the expected and the current hardware configuration of centrally installed racks and stations of a DP master system.

Reading out the SSL with SFC51 „RDSYSST“ using a S7-400 CPU

If you read out the partial list with SFC51 you must see to the fact that the parameters SSL_ID and INDEX of SFC51 match each other.

SSL_ID	INDEX
W#16#0092 or W#16#0292 or W#16#0692 or	DP master system ID of a DP master system which is connected via an **integrated** DP switch.
W#16#4092 or W#16#4292 or W#16#4692 or	DP master system ID of a DP master system which is connected via an **external** DP switch.

Header

The header of partial list SSL-ID W#16#xy92 is structured as follows:

Contents	Meaning	
SSL-ID	SSL-ID of the partial list extract:	
	W#16#0092:	Expected status of the central racks/stations of a DP master system connected via an integrated DP interface
	W#16#4092:	Expected status of the stations of a DP master system connected via an external DP interface
	W#16#0292:	Actual status of the central racks/stations of a DP master system connected via an integrated DP interface
	W#16#0392	Status of battery powered buffering of a CPU rack/station if at least one battery has failed
	W#16#0492	Status of the entire battery powered buffering of all racks/stations of a CPU
	W#16#0592	Status of the 24 V supply of all racks/stations of a CPU
	W#16#4292:	Actual status of the stations of a DP master system connected via an external DP interface
	W#16#0692:	OK state of the expansion racks in the central configuration/of the stations of a DP master system connected via an integrated DP interface
	W#16#4692:	OK state of the stations of a DP master system connected via an external DP interface

Contents	Meaning
INDEX	0/ DP master system ID
LENTHDR	W#16#0010: one data record is 8 words long (16 bytes)
N_DR	Number of data records

Data Record

A data record of the partial list with the ID W#16#xy92 has the following structure:

Contents	Length	Meaning
status_0 to status_15	16 bytes	Rack status/ station status or backup status. (The backup status is only relevant for DP modules) W#16#0092: Bit=0: rack/station not configured Bit=1: rack/station configured W#16#4092 Bit=0: station not configured Bit=1: station configured W#16#0192: Bit=0: station is not configured or configured and activated Bit=1: station is configured and activated W#16#0292: Bit=0: rack/station failure, deactivated or not configured Bit=1: rack/station exists, activated and has not failed W#16#4292: Bit=0: station failure, deactivated or or not configured Bit=1: station exists, activated and has not failed W#16#0692: Bit=0: all modules of the expansion rack/ of a station exist, are available and no problems and the station is activated Bit=1: at least 1 module of the expansion rack/ of a station is not OK or the station is deactivated W#16#4692: Bit=0: all modules of a station exist are available and no problems, and the station is activated Bit=1: at least 1 module of a station is not ok or the station is deactivated
status_0	1 byte	Bit 0: Central rack (INDEX = 0) or station 1 (INDEX tu0) Bit 1: 1. Expansion rack or station 2 : : Bit 7: 7. Expansion rack or station 8
status_1	1 byte	Bit 0: 8. Expansion rack or station 9 : : Bit 7: 15. Expansion rack or station 16
status_2	1 byte	Bit 0: 16. Expansion rack or station 17 : : Bit 5: 21. Expansion rack or station 22

Contents	Length	Meaning
		Bit 6: 0 or station 23 Bit 7: 0 or station 24
status_3	1 byte	Bit 0: 0 or station 25 : : Bit 5: 0 or station 30 Bit 6: Expansion rack (SIMATIC S5 area) or station 31 Bit 7: 0 or station 32
status_4	1 byte	Bit 0: 0 or station 33 : : Bit 7: 0 or station 40
: :		
status_15	1 byte	Bit 0: 0 or station 121 : : Bit 7: 0 or station 128

31.23 SSL-ID W#16#xyA0 - Diagnostic Buffer

Purpose

If you read the partial list SSL-ID W#16#xyA0, you obtain the entries in the diagnostic buffer of the module.

Header

The header of partial list SSL-ID W#16#xyA0 is structured as follows:

Contents	Meaning
SSL-ID	The SSL-ID of the partial list extract: W#16#00A0: All entries possible in the current mode W#16#01A0: The most recent entries; you specify the number of most recent entries with the INDEX parameter. If the number of messages in the diagnostic buffer is smaller than the configured maximum number of messages, the SFC51 may provide invalid values using this partial list extract. You therefore should avoid a power loss which is not backed up! W#16#0FA0: Only partial list header information
INDEX	Only for SSL-ID W#16#01A0: Number of most recent entries
LENTHDR	W#16#0014: one data record is 10 words long (20 bytes)
N_DR	Number of data records

Data Record

A data record of partial list SSL-ID W#16#xyA0 has the following structure:

Name	Length	Meaning
ID	1 word	Event ID
info	5 words	Information about the event and its consequences
time	4 words	Time stamp of the event

Diagnostic Buffer

You obtain more information about the events in the diagnostic buffer using STEP 7.

31.24 SSL-ID W#16#00B1 - Module Diagnostic Information

Purpose

If you read the partial list SSL-ID W#16#00B1, you obtain the first 4 diagnostic bytes of a module with diagnostic capability.

Header

The header of partial list SSL-ID W#16#00B1 is structured as follows:

	Meaning
SSL-ID	W#16#00B1
INDEX	Bit 0 to bit 14: logical base address Bit 15: 0 = input, 1 = output
LENTHDR	W#16#0004: one data record is 2 words long (4 bytes)
N_DR	1

Data Record

A data record of partial list SSL-ID W#16#00B1 has the following structure:

Name	Length	Meaning	
byte1	1 byte	Bit 0: Bit 1: Bit 2: Bit 3: Bit 4: Bit 5: Bit 6: Bit 7:	Module fault/OK (group fault ID) Internal fault External fault Channel error exists No external auxiliary voltage No front connector Module not assigned parameters Wrong parameters on module
byte2	1 byte	Bit 0 to bit 3: Bit 4: Bit 5: Bit 6: Bit 7:	Module class (CPU, FM, CP, IM, SM, ...) Channel information exists User information exists Diagnostic interrupt from substitute Reserve (initialized with 0)
byte3	1 byte	Bit 0: Bit 1: Bit 2: Bit 3: Bit 4: Bit 5: Bit 6: Bit 7:	User module incorrect/does not exist Communication fault Mode RUN/STOP (0 = RUN, 1 = STOP) Watchdog responded Internal module power supply failed Battery exhausted (BFS) Entire buffer failed Reserve (initialized with 0)
byte4	1 byte	Bit 0: Bit 1: Bit 2: Bit 3: Bit 4: Bit 5: Bit 6: Bit 7:	Expansion rack failure (detected by IM) Processor failure EPROM error RAM error ADC/DAC error Fuse blown Hardware error lost Reserve (initialized with 0)

31.25 SSL-ID W#16#00B2 - Diagnostic Data Record 1 with Physical Address

Purpose

If you read the partial list with SSL-ID W#16#00B2, you obtain diagnostic data record 1 of a module in a central rack (not for DP or submodules). You specify the number using the rack and slot number.

Header

The header of partial list SSL-ID W#16#00B2 is structured as follows:

Contents	Meaning
SSL-ID	W#16#00B2
INDEX	W#16#xxyy: xx contains the number of the rack yy contains the slot number
LENTHDR	The length of the data record depends on the module.
N_DR	1

Data Record

The size of a data record of partial list SSL-ID W#16#00B2 and its contents depend on the particular module. For further information refer to /**70**/, /**101**/ and to the manual describing the module concerned.

31.26 SSL-ID W#16#00B3 - Module Diagnostic Data with Logical Base Address

Purpose

If you read the partial list SSL-ID W#16#00B3, you obtain all the diagnostic data of a module. You can also obtain this information for DP and submodules. You select the module using its logical base address.

Header

The header of partial list SSL-ID W#16#00B3 is structured as follows:

Contents	Meaning
SSL-ID	W#16#00B3
INDEX	Bit 0 to bit 14: logical base address Bit 15: 0 = input, 1 = output
LENTHDR	The length of the data record depends on the module.
N_DR	1

Data Record

The size of a data record of partial list SSL-ID W#16#00B3 and its contents depend on the particular module. For further information refer to **/70/Gloss.70.**, **/101/** and to the manual describing the module concerned.

Note

With SFC51 you must read out the partial list with the SSL-ID W#16#00B3 only outside OB82.

31.27 SSL-ID W#16#00B4 - Diagnostic Data of a DP Slave

Purpose

If you read the partial list SSL-ID W#16#00B4, you obtain the diagnostic data of a DP slave. This diagnostic data is structured in compliance with EN 50 170 Volume 2, PROFIBUS. You select the module using the diagnostic address you configured.

Header

The header of partial list SSL-ID W#16#00B4 is structured as follows:

Contents	Meaning
SSL-ID	W#16#00B4
INDEX	Configured diagnostic address of the DP slave
LENTHDR	Length of a data record. The maximum length is 240 bytes. For standard slaves which have a diagnostic data length of more than 240 bytes up to a maximum of 244 bytes, the first 240 bytes are read and the overflow bit is set in the data.
N_DR	1

Data Record

A data record of partial list SSL-ID W#16#00B4 has the following structure:

Name	Length	Meaning
status1	1 byte	Station status1
status2	1 byte	Station status2
status3	1 byte	Station status3
stat_nr	1 byte	Master station number
ken_hi	1 byte	Vendor ID (high byte)
ken_lo	1 byte	Vendor ID (low byte)
....	....	Further diagnostic data specific to the particular slave

32 Events

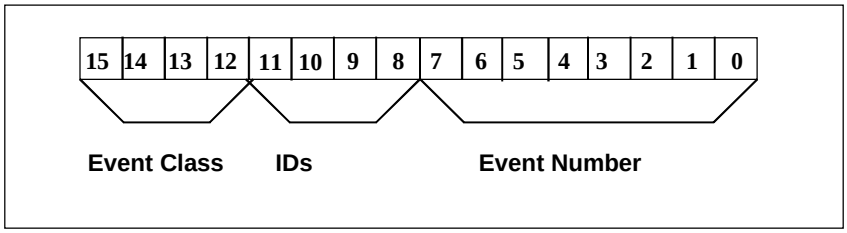
32.1 Events and Event ID

Event

All events are numbered within the SIMATIC S7 programmable logic controller. This allows you to relate a message text to an event.

Event ID

An event ID is assigned to every event. The event ID is structured as follows:



Structure of the Event ID.

Event Class

The event classes are as follows:

Number	Event Class
1	Standard OB events
2	Synchronous errors
3	Asynchronous errors
4	Mode transitions
5	Run-time events
6	Communication events
7	Events for fail-safe and fault-tolerant systems
8	Standardized diagnostic data on modules
9	Predefined user events
A, B	Freely definable events
C , D, E	Reserved
F	Events for modules other than CPUs (for example, CPs, FMs)

Identifier

The identifier is used to distinguish the type of events. The four bits have the following significance:

Bit No. in the Event ID	Meaning	
8	= 0 = 1	Event leaving state Event entering state
9	= 1	Entry in diagnostic buffer
10	= 1	Internal error
11	= 1	External error

32.2 Event Class 1 - Standard OB Events

Event ID	Event
W#16#1164	Synchronous cycle interrupt 1
W#16#1165	Synchronous cycle interrupt 2
W#16#1166	Synchronous cycle interrupt 3
W#16#1167	Synchronous cycle interrupt 4
W#16#1381	Request for manual warm restart
W#16#1382	Request for automatic warm restart
W#16#1383	Request for manual hot restart
W#16#1384	Request for automatic hot restart
W#16#1385	Request for manual cold restart
W#16#1386	Request for automatic cold restart
W#16#1387	Master CPU: request for manual cold restart
W#16#1388	Master CPU: request for automatic cold restart
W#16#138A	Master CPU: request for manual warm restart
W#16#138B	Master CPU: request for automatic warm restart
W#16#138C	Standby CPU: request for manual hot restart
W#16#138D	Standby CPU: request for automatic hot restart

32.3 Event Class 2 - Synchronous Errors

Event ID	Event	OB
W#16#2521	BCD conversion error	OB 121
W#16#2522	Area length error when reading	OB 121
W#16#2523	Area length error when writing	OB 121
W#16#2524	Area error when reading	OB 121
W#16#2525	Area error when writing	OB 121
W#16#2526	Timer number error	OB 121
W#16#2527	Counter number error	OB 121
W#16#2528	Alignment error when reading	OB 121
W#16#2529	Alignment error when writing	OB 121
W#16#2530	Write error when accessing the DB	OB 121
W#16#2531	Write error when accessing the DI	OB 121
W#16#2532	Block number error when opening a DB	OB 121
W#16#2533	Block number error when opening a DI	OB 121
W#16#2534	Block number error when calling an FC	OB 121
W#16#2535	Block number error when calling an FB	OB 121
W#16#253A	DB not loaded	OB 121
W#16#253C	FC not loaded	OB 121
W#16#253D	SFC not loaded	OB 121
W#16#253E	FB not loaded	OB 121
W#16#253F	SFB not loaded	OB 121
W#16#2942	I/O access error, reading	OB 122
W#16#2943	I/O access error, writing	OB 122
W#16#2944	I/O access error in the nth read access (n>1)	OB 122
W#16#2945	I/O access error in the nth write access (n>1)	OB 122

32.4 Event Class 3 - Asynchronous Errors

Event ID	Event	OB
W#16#3501	Cycle time exceeded.	OB 80
W#16#3502	User interface (OB or FRB) request error	OB 80
W#16#3503	Delay too long processing a priority class	-
W#16#3505	Time-of-day interrupt(s) skipped due to new clock setting	OB 80
W#16#3506	Time-of-day interrupt(s) skipped when changing to RUN after HOLD	OB 80
W#16#3507	Multiple OB request errors caused internal buffer overflow	OB 80
W#16#3508	Synchronous cycle interrupt-timing error	OB 80
W#16#3921/3821	BATTF: failure on at least one backup battery of the central rack/ problem eliminated Note: the event entering state only occurs if one of the backup battery fails (if there are redundant backup batteries). If the other backup battery should also happen to fail, the event will no occur again.	OB 81
W#16#3922/3822	BAF: failure of backup voltage on central rack/ problem eliminated	OB 81
W#16#3923/3823	24 volt supply failure on central rack / problem eliminated	OB 81
W#16#3925/3825	BATTF: failure on at least one backup battery of the redundant central rack/ problem eliminated	OB 81
W#16#3926/3826	BAF: failure of backup voltage on redundant central rack/ problem eliminated	OB 81
W#16#3917/3827	24 volt supply failure on redundant central rack / problem eliminated	OB 81
W#16#3931/3831	BATTF: failure of at least one backup battery of the expansion rack/ problem eliminated	OB 81
W#16#3932/3832	BAF: failure of backup voltage on expansion rack/ problem eliminated	OB 81
W#16#3933/3833	24 volt supply failure on at least one expansion rack/ problem eliminated	OB 81
W#16#3942	Module fault present	OB 82
W#16#3842	Module OK	OB 82
W#16#3861	Module/interface module inserted, module type OK	OB 83
W#16#3961	Module/interface module removed, cannot be addressed	OB 83
W#16#3863	Module/interface module plugged in, but wrong module type	OB 83
W#16#3864	Module/interface module plugged in, but causing problem (type ID unreadable)	OB 83
W#16#3866	Module plugged in, but error in module parameter assignment	OB 83
W#16#3966	Module can be addressed again, load voltage error removed	OB 83
W#16#3865	Module cannot be addressed, load voltage error	OB 83
W#16#3884	Interface module plugged in	OB 83
W#16#3984	Interface module removed	OB 83
W#16#3981	Interface error entering state	OB 84
W#16#3881	Interface error leaving state	OB 84

Event ID	Event	OB
W#16#35A1	User interface (OB or FRB) not found	OB 85
W#16#35A2	OB not loaded (started by SFC or operating system due to configuration)	OB 85
W#16#35A3	Error when operating system accesses a block	OB 85
W#16#39B1	I/O access error when updating the process image input table	OB 85
W#16#39B2	I/O access error when transferring the process image to the output modules	OB 85
W#16#39B3/38B3	I/O access error updating the process image input table	OB 85
W#16#39B4/38B4	I/O access error when transferring the process image to the output modules	OB 85
W#16#38C1	Expansion rack operational again (1 to 21), leaving state	OB 86
W#16#39C1	Expansion rack failure (1 to 21), entering state	OB 86
W#16#38C2	Expansion rack operational again but mismatch between setpoint and actual configuration	OB 86
W#16#39C3	Distributed I/Os: master system failure entering state	OB 86
W#16#39C4	Distributed I/Os: station failure, entering state	OB 86
W#16#38C4	Distributed I/Os: station failure, leaving state	OB 86
W#16#39C5	Distributed I/Os: station fault, entering state	OB 86
W#16#38C5	Distributed I/Os: station fault, leaving state	OB 86
W#16#38C6	Expansion rack operational again, but error(s) in module parameter assignment	OB 86
W#16#38C7	DP: station operational again, but error(s) in module parameter assignment	OB 86
W#16#38C8	DP: station operational again, but mismatch between setpoint and actual configuration	OB 86
W#16#35D2	Diagnostic entries cannot be sent at present	OB 87
W#16#35D3	Synchronization frames cannot be sent	OB 87
W#16#35D4	Illegal time jump resulting from synchronization	OB 87
W#16#35D5	Error adopting the synchronization time	OB 87
W#16#35E1	Incorrect frame ID in GD	OB 87
W#16#35E2	GD packet status cannot be entered in DB	OB 87
W#16#35E3	Frame length error in GD	OB 87
W#16#35E4	Illegal GD packet number received	OB 87
W#16#35E5	Error accessing DB in communication SFBs for configured S7 connections	OB 87
W#16#35E6	GD total status cannot be entered in DB	OB 87

32.5 Event Class 4 - Stop Events and Other Mode Changes

Event ID	Event
W#16#4300	Backed-up power on
W#16#4301	Mode transition from STOP to STARTUP
W#16#4302	Mode transition from STARTUP to RUN
W#16#4303	STOP caused by stop switch being activated
W#16#4304	STOP caused by PG STOP operation or by SFB 20 "STOP"
W#16#4305	HOLD: breakpoint reached
W#16#4306	HOLD: breakpoint exited
W#16#4307	Memory reset started by PG operation
W#16#4308	Memory reset started by switch setting
W#16#4309	Memory reset started automatically (power on not backed up)
W#16#430A	HOLD exited, transition to STOP
W#16#430D	STOP caused by other CPU in multicomputing
W#16#430E	Memory reset executed
W#16#430F	STOP on the module due to STOP on a CPU
W#16#4510	STOP violation of the CPU's data range
W#16#4520	DEFECTIVE: STOP not possible
W#16#4521	DEFECTIVE: failure of instruction processing processor
W#16#4522	DEFECTIVE: failure of clock chip
W#16#4523	DEFECTIVE: failure of clock pulse generator
W#16#4524	DEFECTIVE: failure of timer update function
W#16#4525	DEFECTIVE: failure of multicomputing synchronization
W#16#4926	DEFECTIVE: failure of the watchdog for I/O access
W#16#4527	DEFECTIVE: failure of I/O access monitoring
W#16#4528	DEFECTIVE: failure of scan time monitoring
W#16#4530	DEFECTIVE: memory test error in internal memory
W#16#4931	STOP or DEFECTIVE: memory test error in memory submodule
W#16#4532	DEFECTIVE: failure of core resources
W#16#4933	DEFECTIVE: checksum error
W#16#4934	DEFECTIVE: memory not available
W#16#4935	DEFECTIVE: cancelled by watchdog/processor exceptions
W#16#4536	DEFECTIVE: switch defective
W#16#4540	STOP: Memory expansion of the internal work memory has gaps. First memory expansion too small or missing.
W#16#4541	STOP caused by priority class system
W#16#4542	STOP caused by object management system
W#16#4543	STOP caused by test functions
W#16#4544	STOP caused by diagnostic system
W#16#4545	STOP caused by communication system
W#16#4546	STOP caused by CPU memory management
W#16#4547	STOP caused by process image management

Event ID	Event
W#16#4548	STOP caused by I/O management
W#16#4949	STOP caused by continuous hardware interrupt
W#16#454A	STOP caused by configuration: an OB deselected with STEP 7 was being loaded into the CPU during STARTUP
W#16#494D	STOP caused by I/O error
W#16#494E	STOP caused by power failure
W#16#494F	STOP caused by configuration error
W#16#4550	DEFECTIVE: internal system error
W#16#4555	No restart possible, monitoring time elapsed
W#16#4556	STOP: memory reset request from communication system
W#16#4357	Module watchdog started
W#16#4358	All modules are ready for operation
W#16#4959	One or more modules not ready for operation
W#16#4562	STOP caused by programming error (OB not loaded or not possible, or no FRB)
W#16#4563	STOP caused by I/O access error (OB not loaded or not possible, or no FRB)
W#16#4567	STOP caused by H event
W#16#4568	STOP caused by time error (OB not loaded or not possible, or no FRB)
W#16#456A	STOP caused by diagnostic interrupt (OB not loaded or not possible, or no FRB)
W#16#456B	STOP caused by removing/inserting module (OB not loaded or not possible, or no FRB)
W#16#456C	STOP caused by CPU hardware error (OB not loaded or not possible, or no FRB)
W#16#456D	STOP caused by program sequence error (OB not loaded or not possible, or no FRB)
W#16#456E	STOP caused by communication error (OB not loaded or not possible, or no FRB)
W#16#456F	STOP caused by rack failure OB (OB not loaded or not possible, or no FRB)
W#16#4571	STOP caused by nesting stack error
W#16#4572	STOP caused by master control relay stack error
W#16#4573	STOP caused by exceeding the nesting depth for synchronous errors
W#16#4574	STOP caused by exceeding interrupt stack nesting depth in the priority class stack
W#16#4575	STOP caused by exceeding block stack nesting depth in the priority class stack
W#16#4576	STOP caused by error when allocating the local data
W#16#4578	STOP caused by unknown opcode
W#16#457A	STOP caused by code length error
W#16#457B	STOP caused by DB not being loaded on on-board I/Os
W#16#457F	STOP caused by STOP command
W#16#4580	STOP: back-up buffer contents inconsistent (no transition to RUN)
W#16#4590	STOP caused by overloading the internal functions
W#16#49A0	STOP caused by parameter assignment error or non-permissible variation of setpoint and actual extension: Start-up blocked.
W#16#49A1	STOP caused by parameter assignment error: memory reset request
W#16#49A2	STOP caused by error in parameter modification: startup disabled
W#16#49A3	STOP caused by error in parameter modification: memory reset request
W#16#49A4	STOP: inconsistency in configuration data

Event ID	Event
W#16#49A5	STOP: distributed I/Os: inconsistency in the loaded configuration information
W#16#49A6	STOP: distributed I/Os: invalid configuration information
W#16#49A7	STOP: distributed I/Os: no configuration information
W#16#49A8	STOP: error indicated by the interface module for the distributed I/Os
W#16#43B0	Firmware update was successful
W#16#49B1	Firmware update data incorrect
W#16#49B2	Firmware update: hardware version does not match firmware
W#16#49B3	Firmware update: module type does not match firmware
W#16#49D0	LINK-UP aborted due to violation of coordination rules
W#16#49D1	LINK-UP/UPDATE sequence aborted
W#16#49D2	Standby CPU changed to STOP due to STOP on the master CPU during link-up
W#16#43D3	STOP on standby CPU
W#16#49D4	STOP on a master, since partner CPU is also a master (link-up error)
W#16#45D5	LINK-UP rejected due to mismatched CPU memory configuration of the sub-PLC
W#16#45D6	LINK-UP rejected due to mismatched system program of the sub-PLC
W#16#49D7	LINK-UP rejected due to change in user program or in configuration
W#16#45D8	DEFECTIVE: hardware fault detected due to other error
W#16#45D9	STOP due to SYNC module error
W#16#45DA	STOP due to synchronization error between H CPUs
W#16#43DC	Abort during link-up with switchover
W#16#45DD	LINK-UP rejected due to running test or other online functions
W#16#43DE	Updating aborted due to monitoring time being exceeded during the n-th attempt, new update attempt initiated
W#16#43DF	Updating aborted for final time due to monitoring time being exceeded after completing the maximum amount of attempts. User intervention required.
W#16#43E0	Change from solo mode after link-up
W#16#43E1	Change from link-up after updating
W#16#43E2	Change from updating to redundant mode
W#16#43E3	Master CPU: change from redundant mode to solo mode
W#16#43E4	Standby CPU: change from redundant mode after error-search mode
W#16#43E5	Standby CPU: change from error-search mode after link-up or STOP
W#16#43E6	Link-up aborted on the standby CPU
W#16#43E7	Updating aborted on the standby CPU
W#16#43E8	Standby CPU: change from link-up after startup
W#16#43E9	Standby CPU: change from startup after updating
W#16#43F1	Reserve-master switchover
W#16#43F2	Coupling of incompatible H-CPUs blocked by system program

32.6 Event Class 5 - Mode Run-time Events

Event ID	Event
W#16#530D	New startup information in the STOP mode
W#16#5311	Startup despite no ready message from module(s)
W#16#5961	Parameter assignment error
W#16#5962	Parameter assignment error preventing startup
W#16#5963	Parameter assignment error with memory reset request
W#16#5966	Parameter assignment error when switching
W#16#5969	Parameter assignment error with startup blocked
W#16#5371	Distributed I/Os: end of the synchronization with a DP master
W#16#5979/5879	Diagnostic message from DP interface: EXTF LED on/off
W#16#597C	DP Global Control command failed or moved
W#16#5380	Diagnostic buffer entries of interrupt and asynchronous errors disabled
W#16#5395	Distributed I/Os: reset of a DP master
W#16#596	Parameter assignment error when switching

32.7 Event Class 6 - Communication Events

Event ID	Event
W#16#6500	Connection ID exists twice on module
W#16#6501	Connection resources inadequate
W#16#6502	Error in the connection description
W#16#6905/6805	Resource problem on configured connections/eliminated
W#16#6510	CFB structure error detected in instance DB when evaluating EPROM
W#16#6514	GD packet number exists twice on the module
W#16#6515	Inconsistent length specifications in GD configuration information
W#16#6316	Interface error when starting programmable controller
W#16#6521	No memory submodule and no internal memory available
W#16#6522	Illegal memory submodule: replace submodule and reset memory
W#16#6523	Memory reset request due to error accessing submodule
W#16#6524	Memory reset request due to error in block header
W#16#6526	Memory reset request due to memory replacement
W#16#6527	Memory replaced, therefore restart not possible
W#16#6528	Object handling function in the STOP/HOLD mode, no restart possible
W#16#6529	No startup possible during the "load user program" function
W#16#652A	No startup because block exists twice in user memory
W#16#652B	No startup because block is too long for submodule - replace submodule
W#16#652C	No startup due to illegal OB on submodule
W#16#6532	No startup because illegal configuration information on submodule

Event ID	Event
W#16#6533	Memory reset request because of invalid submodule content
W#16#6534	No startup: block exists more than once on submodule
W#16#6535	No startup: not enough memory to transfer block from submodule
W#16#6536	No startup: submodule contains an illegal block number
W#16#6537	No startup: submodule contains a block with an illegal length
W#16#6538	Local data or write-protection ID (for DB) of a block illegal for CPU
W#16#6539	Illegal command in block (detected by compiler)
W#16#653A	Memory reset request because local OB data on submodule too short
W#16#6543	No startup: illegal block type
W#16#6544	No startup: attribute "relevant for processing" illegal
W#16#6545	Source language illegal
W#16#6546	Maximum amount of configuration information reached
W#16#6547	Parameter assignment error assigning parameters to modules (not on P bus, cancel download)
W#16#6548	Plausibility error during block check
W#16#6549	Structure error in block
W#16#6550	A block has an error in the CRC
W#16#6551	A block has no CRC
W#16#6560	SCAN overflow
W#16#6981	Interface error entering state
W#16#6881	Interface error leaving state
W#16#6390	Formatting of Micro Memory Card complete

32.8 Event Class 7 - H/F Events

Event ID	Event	OB
W#16#72A2	Failure of a DP master or a DP master system	OB 70
W#16#72A3	Redundancy restored on the DP slave	OB 70
W#16#7310/7210	Loss of redundancy I/O	OB 70
W#16#7311/7211	Partial loss of redundancy I/O	OB 70
W#16#73A3	Loss of redundancy on the DP slave	OB 70
W#16#7301	Loss of redundancy (1 of 2) due to failure of a CPU	OB 72
W#16#7302	Loss of redundancy (1 of 2) due to STOP on the standby triggered by user	OB 72
W#16#7303	H system (1 of 2) changed to redundant mode	OB 72
W#16#7520	Error in RAM comparison	OB 72
W#16#7521	Error in comparison of process image output value	OB 72
W#16#7522	Error in comparison of memory bits, timers, or counters	OB 72
W#16#7323	Discrepancy found in operating system data	OB 72
W#16#7331	Standby-master switchover due to master failure	OB 72
W#16#7333	Standby-master switchover due to operator intervention	OB 72

Event ID	Event	OB
W#16#7934	Standby-master switchover due to connection problem at the SYNC module	OB 72
W#16#7335	Standby-master switchover triggered by SFC 90 "H_CTRL"	OB 72
W#16#7340	Synchronization error in user program due to elapsed wait time	OB 72
W#16#7341	Synchronization error in user program due to waiting at different synchronization points	OB 72
W#16#7342	Synchronization error in operating system due to waiting at different synchronization points	OB 72
W#16#7343	Synchronization error in operating system due to elapsed wait time	OB 72
W#16#7344	Synchronization error in operating system due to incorrect data	OB 72
W#16#734A	The "depassivation" job triggered by SFC 90 "H_CTRL" was executed.	OB 72
W#16#73C1	Update process canceled	OB 72
W#16#73C2	Updating aborted due to monitoring time being exceeded during the n-th attempt ($1 \leq n \leq$ max. possible number of update attempts after abort due to excessive monitoring time)	OB 72
W#16#7950	Synchronization module missing	OB 72
W#16#7951	Change at the SYNC module without Power On	OB 72
W#16#7952/7852	SYNC module removed/inserted	OB 72
W#16#7953	Change at the SYNC-module without reset	OB 72
W#16#7954	SYNC module: rack number assigned twice	OB 72
W#16#7955/7855	SYNC module error/eliminated	OB 72
W#16#7956	Illegal rack number set on SYNC module	OB 72
W#16#73E0/72E0	Loss of redundancy in communication/ problem eliminated	OB 73
W#16#734A	The "Depassivation" order triggered by SFC 90 "H_CTRL" has been carried out.	
W#16#734B	The "Depassivation" order triggered by the operating system has been carried out.	

32.9 Event Class 8 - Diagnostic Events for Modules

Event ID	Event	Module type
W#16#8x00	Module fault/OK	Any
W#16#8x01	Internal error	
W#16#8x02	External error	
W#16#8x03	Channel error	
W#16#8x04	No external auxiliary voltage	
W#16#8x05	No front connector	
W#16#8x06	No parameter assignment	
W#16#8x07	Incorrect parameters in module	
W#16#8x30	User submodule incorrect/not found	
W#16#8x31	Communication problem	
W#16#8x32	Operating mode: RUN/STOP (STOP: entering state, RUN: leaving state)	
W#16#8x33	Time monitoring responded (watchdog)	
W#16#8x34	Internal module power failure	
W#16#8x35	BATTF: battery exhausted	
W#16#8x36	Total backup failed	
W#16#8x37	Reserved	
W#16#8x40	Expansion rack failed	
W#16#8x41	Processor failure	
W#16#8x42	EPROM error	
W#16#8x43	RAM error	
W#16#8x44	ADC/DAC error	
W#16#8x45	Fuse blown	
W#16#8x46	Hardware interrupt lost	
W#16#8x47	Reserved	
W#16#8x50	Configuration/parameter assignment error	Analog input
W#16#8x51	Common mode error	
W#16#8x52	Short circuit to phase	
W#16#8x53	Short circuit to ground	
W#16#8x54	Wire break	
W#16#8x55	Reference channel error	
W#16#8x56	Below measuring range	
W#16#8x57	Above measuring range	
W#16#8x60	Configuration/parameter assignment error	Analog output
W#16#8x61	Common mode error	
W#16#8x62	Short circuit to phase	
W#16#8x63	Short circuit to ground	
W#16#8x64	Wire break	
W#16#8x65	Reserved	
W#16#8x66	No load voltage	

Event ID	Event	Module type
W#16#8x70	Configuration/parameter assignment error	Digital input
W#16#8x71	Chassis ground fault	
W#16#8x72	Short circuit to phase (sensor)	
W#16#8x73	Short circuit to ground (sensor)	
W#16#8x74	Wire break	
W#16#8x75	No sensor power supply	
W#16#8x80	Configuration/parameter assignment error	Digital output
W#16#8x81	Chassis ground fault	
W#16#8x82	Short circuit to phase	
W#16#8x83	Short circuit to ground	
W#16#8x84	Wire break	
W#16#8x85	Fuse tripped	
W#16#8x86	No load voltage	
W#16#8x87	Excess temperature	
W#16#8xB0	Counter module, signal A faulty	FM
W#16#8xB1	Counter module, signal B faulty	
W#16#8xB2	Counter module, signal N faulty	
W#16#8xB3	Counter module, incorrect value passed between the channels	
W#16#8xB4	Counter module, 5.2 V sensor supply faulty	
W#16#8xB5	Counter module, 24 V sensor supply faulty	

32.10 Event Class 9 - Standard User Events

Event ID	Event
W#16#9001	Automatic mode
W#16#9101	Manual mode
W#16#9x02	OPEN/CLOSED, ON/OFF
W#16#9x03	Manual command enable
W#16#9x04	Unit protective command (OPEN/CLOSED)
W#16#9x05	Process enable
W#16#9x06	System protection command
W#16#9x07	Process value monitoring responded
W#16#9x08	Manipulated variable monitoring responded
W#16#9x09	System deviation greater than permitted
W#16#9x0A	Limit position error
W#16#9x0B	Runtime error
W#16#9x0C	Command execution error (sequencer)
W#16#9x0D	Operating status running > OPEN
W#16#9x0E	Operating status running > CLOSED
W#16#9x0F	Command blocking
W#16#9x11	Process status OPEN/ON
W#16#9x12	Process status CLOSED/OFF
W#16#9x13	Process status intermediate position
W#16#9x14	Process status ON via AUTO
W#16#9x15	Process status ON via manual
W#16#9x16	Process status ON via protective command
W#16#9x17	Process status OFF via AUTO
W#16#9x18	Process status OFF via manual
W#16#9x19	Process status OFF via protective command
W#16#9x21	Function error on approach
W#16#9x22	Function error on leaving
W#16#9x31	Actuator (DE/WE) limit position OPEN
W#16#9x32	Actuator (DE/WE) limit position not OPEN
W#16#9x33	Actuator (DE/WE) limit position CLOSED
W#16#9x34	Actuator (DE/WE) limit position not CLOSED
W#16#9x41	Illegal status, tolerance time elapsed
W#16#9x42	Illegal status, tolerance time not elapsed
W#16#9x43	Interlock error, tolerance time = 0
W#16#9x44	Interlock error, tolerance time > 0
W#16#9x45	No reaction
W#16#9x46	Final status exited illegally, tolerance time = 0
W#16#9x47	Final status exited illegally, tolerance time > 0
W#16#9x50	Upper limit of signal range USR
W#16#9x51	Upper limit of measuring range UMR

Event ID	Event
W#16#9x52	Lower limit of signal range LSR
W#16#9x53	Lower limit of measuring range LMR
W#16#9x54	Upper alarm limit UAL
W#16#9x55	Upper warning limit UWL
W#16#9x56	Upper tolerance limit UTL
W#16#9x57	Lower tolerance limit LTL
W#16#9x58	Lower warning limit LWL
W#16#9x59	Lower alarm limit LAL
W#16#9x60	GRAPH7 step entering/leaving
W#16#9x61	GRAPH7 interlock error
W#16#9x62	GRAPH7 execution error
W#16#9x63	GRAPH7 error noted
W#16#9x64	GRAPH7 error acknowledged
W#16#9x70	Trend exceeded in positive direction
W#16#9x71	Trend exceeded in negative direction
W#16#9x72	No reaction
W#16#9x73	Final state exited illegally
W#16#9x80	Limit value exceeded, tolerance time = 0
W#16#9x81	Limit value exceeded, tolerance time > 0
W#16#9x82	Below limit value, tolerance time = 0
W#16#9x83	Below limit value, tolerance time > 0
W#16#9x84	Gradient exceeded, tolerance time = 0
W#16#9x85	Gradient exceeded, tolerance time > 0
W#16#9x86	Below gradient, tolerance time = 0
W#16#9x87	Below gradient, tolerance time > 0
W#16#9190/9090	User parameter assignment error entering/leaving
W#16#91F0	Overflow
W#16#91F1	Underflow
W#16#91F2	Division by 0
W#16#91F3	Illegal calculation operation

32.11 Event Classes A and B - Free User Events

Event ID	Event
W#16#Axyz	Events available for user
W#16#Bxyz	

32.12 Reserved Event Classes

Reserved

The following event classes are reserved for later expansions:

- C
- D
- E
- F Reserved for modules not in central rack (for example, CPs or FMs)

33 List of SFCs, and SFBs

33.1 List of SFCs, Sorted Numerically

No.	Short Name	Function
SFC 0	SET_CLK	Set System Clock
SFC 1	READ_CLK	Read System Clock
SFC 2	SET_RTM	Set Run-time Meter
SFC 3	CTRL_RTM	Start/Stop Run-time Meter
SFC 4	READ_RTM	Read Run-time Meter
SFC 5	GADR_LGC	Query Logical Address of a Channel
SFC 6	RD_SINFO	Read OB Start Information
SFC 7	DP_PRAL	Trigger a Hardware Interrupt on the DP Master
SFC 9	EN_MSG	Enable Block-Related, Symbol-Related and Group Status Messages
SFC 10	DIS_MSG	Disable Block-Related, Symbol-Related and Group Status Messages
SFC 11	DPSYC_FR	Synchronize Groups of DP Slaves
SFC 12	D_ACT_DP	Deactivation and activation of DP slaves
SFC 13	DPNRM_DG	Read Diagnostic Data of a DP Slave (Slave Diagnostics)
SFC 14	DPRD_DAT	Read Consistent Data of a Standard DP Slave
SFC 15	DPWR_DAT	Write Consistent Data to a DP Standard Slave
SFC 17	ALARM_SQ	Generate Acknowledgeable Block-Related Messages
SFC 18	ALARM_S	Generate Permanently Acknowledged Block-Related Messages
SFC 19	ALARM_SC	Query the Acknowledgment Status of the last ALARM_SQ Entering State Message
SFC 20	BLKMOV	Copy Variables
SFC 21	FILL	Initialize a Memory Area
SFC 22	CREAT_DB	Create Data Block
SFC 23	DEL_DB	Delete Data Block
SFC 24	TEST_DB	Test Data Block
SFC 25	COMPRESS	Compress the User Memory
SFC 26	UPDAT_PI	Update the Process Image Update Table
SFC 27	UPDAT_PO	Update the Process Image Output Table
SFC 28	SET_TINT	Set Time-of-Day Interrupt
SFC 29	CAN_TINT	Cancel Time-of-Day Interrupt
SFC 30	ACT_TINT	Activate Time-of-Day Interrupt
SFC 31	QRY_TINT	Query Time-of-Day Interrupt

No.	Short Name	Function
SFC 32	SRT_DINT	Start Time-Delay Interrupt
SFC 33	CAN_DINT	Cancel Time-Delay Interrupt
SFC 34	QRY_DINT	Query Time-Delay Interrupt
SFC 35	MP_ALM	Trigger Multicomputing Interrupt
SFC 36	MSK_FLT	Mask Synchronous Errors
SFC 37	DMSK_FLT	Unmask Synchronous Errors
SFC 38	READ_ERR	Read Error Register
SFC 39	DIS_IRT	Disable New Interrupts and Asynchronous Errors
SFC 40	EN_IRT	Enable New Interrupts and Asynchronous Errors
SFC 41	DIS_AIRT	Delay Higher Priority Interrupts and Asynchronous Errors
SFC 42	EN_AIRT	Enable Higher Priority Interrupts and Asynchronous Errors
SFC 43	RE_TRIGR	Re-trigger Cycle Time Monitoring
SFC 44	REPL_VAL	Transfer Substitute Value to Accumulator 1
SFC 46	STP	Change the CPU to STOP
SFC 47	WAIT	Delay Execution of the User Program
SFC 48	SNC_RTCB	Synchronize Slave Clocks
SFC 49	LGC_GADR	Query the Module Slot Belonging to a Logical Address
SFC 50	RD_LGADR	Query all Logical Addresses of a Module
SFC 51	RDSYSST	Read a System Status List or Partial List
SFC 52	WR_USMSG	Write a User-Defined Diagnostic Event to the Diagnostic Buffer
SFC 54	RD_PARM	Read Defined Parameters
SFC 55	WR_PARM	Write Dynamic Parameters
SFC 56	WR_DPARM	Write Default Parameters
SFC 57	PARM_MOD	Assign Parameters to a Module
SFC 58	WR_REC	Write a Data Record
SFC 59	RD_REC	Read a Data Record
SFC 60	GD_SND	Send a GD Packet
SFC 61	GD_RCV	Fetch a Received GD Packet
SFC 62	CONTROL	Query the Status of a Connection Belonging to a Communication SFB Instance
SFC 63	AB_CALL	Assembly Code Block
SFC 64	TIME_TCK	Read the System Time
SFC 65	X_SEND	Send Data to a Communication Partner outside the Local S7 Station
SFC 66	X_RCV	Receive Data from a Communication Partner outside the Local S7 Station
SFC 67	X_GET	Read Data from a Communication Partner outside the Local S7 Station
SFC 68	X_PUT	Write Data to a Communication Partner outside the Local S7 Station
SFC 69	X_ABORT	Abort an Existing Connection to a Communication Partner outside the Local S7 Station
SFC 72	I_GET	Read Data from a Communication Partner within the Local S7 Station
SFC 73	I_PUT	Write Data to a Communication Partner within the Local S7 Station
SFC 74	I_ABORT	Abort an Existing Connection to a Communication Partner within the Local S7 Station

No.	Short Name	Function
SFC 78	OB_RT	Determine OB program runtime
SFC 79	SET	Set a Range of Outputs
SFC 80	RSET	Reset a Range of Outputs
SFC 81	UBLKMOV	Uninterruptible Block Move
SFC 82	CREA_DBL	Generating a Data Block in the Load Memory
SFC 83	READ_DBL	Reading from a Data Block in Load Memory
SFC 84	WRIT_DBL	Writing from a Data Block in Load Memory
SFC 87	C_DIAG	Diagnosis of the Actual Connection Status
SFC 90	H_CTRL	Control Operation in H Systems
SFC 100	SET_CLKS	Setting the Time-of-Day and the TOD Status
SFC 102	RD_DPARA	Redefined Parameters
SFC 105	READ_SI	Reading Dynamically Occupied System Resources
SFC 106	DEL_SI	Deleting Dynamically Occupied System Resources
SFC 107	ALARM_DQ	Generating Always Acknowledgeable and Block-Related Messages
SFC 108	ALARM_D	Generating Always Acknowledgeable and Block-Related Messages
SFC 126	SYNC_PI	Update process image partition input table in synchronous cycle
SFC 127	SYNC_PO	Update process image partition output table in synchronous cycle

* SFC 63 "AB_CALL" only exists for CPU 614. For a detailed description, refer to the corresponding manual.

33.2 List of SFCs, Sorted Alphabetically

Short Name	No.	Function
AB_CALL	SFC 63	Assembly Code Block
ACT_TINT	SFC 30	Activate Time-of-Day Interrupt
ALARM_D	SFC 108	Generating Always Acknowledgeable and Block-Related Messages
ALARM_DQ	SFC 107	Generating Always Acknowledgeable and Block-Related Messages
ALARM_S	SFC 18	Generate Permanently Acknowledged Block-Related Messages
ALARM_SC	SFC 19	Query the Acknowledgment Status of the last ALARM_SQ Entering State Message
ALARM_SQ	SFC 17	Generate Acknowledgeable Block-Related Messages
BLKMOV	SFC 20	Copy Variables
C_DIAG	SFC 87	Diagnosis of the Actual Connection Status
CAN_DINT	SFC 33	Cancel Time-Delay Interrupt
CAN_TINT	SFC 29	Cancel Time-of-Day Interrupt
COMPRESS	SFC 25	Compress the User Memory
CONTROL	SFC 62	Query the Status of a Connection Belonging to a Communication SFB Instance
CREA_DBL	SFC 82	Generating a Data Block in the Load Memory
CREAT_DB	SFC 22	Create Data Block
CTRL_RTM	SFC 3	Start/Stop Run-time Meter
D_ACT_DP	SFC 12	Deactivation and activation of DP slaves
DEL_DB	SFC 23	Delete Data Block
DEL_SI	SFC 106	Deleting Dynamically Occupied System Resources
DIS_AIRT	SFC 41	Delay Higher Priority Interrupts and Asynchronous Errors
DIS_IRT	SFC 39	Disable New Interrupts and Asynchronous Errors
DIS_MSG	SFC 10	Disable Block-Related, Symbol-Related and Group Status Messages
DMSK_FLT	SFC 37	Unmask Synchronous Errors
DP_PRAL	SFC 7	Trigger a Hardware Interrupt on the DP Master
DPNRM_DG	SFC 13	Read Diagnostic Data of a DP Slave (Slave Diagnostics)
DPRD_DAT	SFC 14	Read Consistent Data of a Standard DP Slave
DPSYC_FR	SFC 11	Synchronize Groups of DP Slaves
DPWR_DAT	SFC 15	Write Consistent Data to a DP Standard Slave
EN_AIRT	SFC 42	Enable Higher Priority Interrupts and Asynchronous Errors
EN_IRT	SFC 40	Enable New Interrupts and Asynchronous Errors
EN_MSG	SFC 9	Enable Block-Related, Symbol-Related and Group Status Messages
FILL	SFC 21	Initialize a Memory Area
GADR_LGC	SFC 5	Query Logical Address of a Channel
GD_RCV	SFC 61	Fetch a Received GD Packet
GD_SND	SFC 60	Send a GD Packet
H_CTRL	SFC 90	Control Operation in H Systems
I_ABORT	SFC 74	Abort an Existing Connection to a Communication Partner within the Local S7 Station

Short Name	No.	Function
I_GET	SFC 72	Read Data from a Communication Partner within the Local S7 Station
I_PUT	SFC 73	Write Data to a Communication Partner within the Local S7 Station
LGC_GADR	SFC 49	Query the Module Slot Belonging to a Logical Address
MP_ALM	SFC 35	Trigger Multicomputing Interrupt
MSK_FLT	SFC 36	Mask Synchronous Errors
PARM_MOD	SFC 57	Assign Parameters to a Module
QRY_DINT	SFC 34	Query Time-Delay Interrupt
QRY_TINT	SFC 31	Query Time-of-Day Interrupt
RD_DPARA	SFC 102	Redefined Parameters
RD_LGADR	SFC 50	Query all Logical Addresses of a Module
RD_PARM	SFC 54	Read Defined Parameters
RD_REC	SFC 59	Read a Data Record
RD_SINFO	SFC 6	Read OB Start Information
RDSYSST	SFC 51	Read a System Status List or Partial List
RE_TRIGR	SFC 43	Re-trigger Cycle Time Monitoring
READ_CLK	SFC 1	Read System Clock
READ_DBL	SFC 83	SFC 83
READ_ERR	SFC 38	Read Error Register
READ_RTM	SFC 4	Read Run-time Meter
READ_SI	SFC 105	Reading Dynamically Occupied System Resources
REPL_VAL	SFC 44	Transfer Substitute Value to Accumulator 1
RSET	SFC 80	Reset a Range of Outputs
SET	SFC 79	Set a Range of Outputs
SET_CLK	SFC 0	Set System Clock
SET_CLKS	SFC 100	Setting the Time-of-Day and the TOD Status
SET_RTM	SFC 2	Set Run-time Meter
SET_TINT	SFC 28	Set Time-of-Day Interrupt
SNC_RTCB	SFC 48	Synchronize Slave Clocks
SRT_DINT	SFC 32	Start Time-Delay Interrupt
STP	SFC 46	Change the CPU to STOP
SYNC_PI	SFC 126	Update process image partition input table in synchronous cycle
SYNC_PO	SFC 127	Update process image partition output table in synchronous cycle
TEST_DB	SFC 24	Test Data Block
TIME_TCK	SFC 64	Read the System Time
UBLKMOV	SFC 81	Uninterruptible Block Move
UPDAT_PI	SFC 26	Update the Process Image Update Table
UPDAT_PO	SFC 27	Update the Process Image Output Table
WAIT	SFC 47	Delay Execution of the User Program
WR_DPARM	SFC 56	Write Default Parameters
WR_PARM	SFC 55	Write Dynamic Parameters
WR_REC	SFC 58	Write a Data Record
WR_USMSG	SFC 52	Write a User-Defined Diagnostic Event to the Diagnostic Buffer

Short Name	No.	Function
WRIT_DBL	SFC 84	Writing from a Data Block in Load Memory
X_ABORT	SFC 69	Abort an Existing Connection to a Communication Partner outside the Local S7 Station
X_GET	SFC 67	Read Data from a Communication Partner outside the Local S7 Station
X_PUT	SFC 68	Write Data to a Communication Partner outside the Local S7 Station
X_RCV	SFC 66	Receive Data from a Communication Partner outside the Local S7 Station
X_SEND	SFC 65	Send Data to a Communication Partner outside the Local S7 Station

* SFC 63 "AB_CALL" only exists for CPU 614. For a detailed description, refer to the corresponding manual.

33.3 List of SFBs, Sorted Numerically

No.	Short Name	Function
SFB 0	CTU	Count Up
SFB 1	CTD	Count Down
SFB 2	CTUD	Count Up/Down
SFB 3	TP	Generate a Pulse
SFB 4	TON	Generate an On Delay
SFB 5	TOF	Generate an Off Delay
SFB 8	USEND	Uncoordinated Sending of Data
SFB 9	URCV	Uncoordinated Receiving of Data
SFB 12	BSEND	Sending Segmented Data
SFB 13	BRCV	Receiving Segmented Data
SFB 14	GET	Read Data from a Remote CPU
SFB 15	PUT	Write Data to a Remote CPU
SFB 16	PRINT	Send Data to Printer
SFB 19	START	Initiate a Warm or Cold Restart on a Remote Device
SFB 20	STOP	Changing a Remote Device to the STOP State
SFB 21	RESUME	Initiate a Hot Restart on a Remote Device
SFB 22	STATUS	Query the Status of a Remote Partner
SFB 23	USTATUS	Receive the Status of a Remote Device
SFB 29	HS_COUNT	Counter (high-speed counter, integrated function)
SFB 30	FREQ_MES	Frequency Meter (frequency meter, integrated function
SFB 32	DRUM	Implement a Sequencer
SFB 33	ALARM	Generate Block-Related Messages with Acknowledgment Display
SFB 34	ALARM_8	Generate Block-Related Messages without Values for 8 Signals
SFB 35	ALARM_8P	Generate Block-Related Messages with Values for 8 Signals
SFB 36	NOTIFY	Generate Block-Related Messages without Acknowledgment Display
SFB 37	AR_SEND	Send Archive Data
SFB 38	HSC_A_B	Counter A/B (integrated function)
SFB 39	POS	Position (integrated function)
SFB 41	CONT_C [1)]	Continuous Control
SFB 42	CONT_S [1)]	Step Control
SFB 43	PULSEGEN [1)]	Pulse Generation
SFB 44	ANALOG [2)]	Positioning with Analog Output
SFB 46	DIGITAL [2)]	Positioning with Digital Output
SFB 47	COUNT [2)]	Controlling the Counter
SFB 48	FREQUENC [2)]	Controlling the Frequency Measurement
SFB 49	PULSE [2)]	Controlling Pulse Width Modulation
SFB 52	RDREC	Reading a Data Record from a DP Slave
SFB 53	WRREC	Writing a Data Record in a DP Slave
SFB 54	RALRM	Receiving an Interrupt from a DP Slave
SFB 60	SEND_PTP [2)]	Sending Data (ASCII, 3964(R))

No.	Short Name	Function
SFB 61	RECV_PTP 2)	Receiving Data (ASCII, 3964(R))
SFB 62	RES_RECV 2)	Deleting the Receive Buffer (ASCII, 3964(R))
SFB 63	SEND_RK 2)	Sending Data (RK 512)
SFB 64	FETCH_RK 2)	Fetching Data (RK 512)
SFB 65	SERVE_RK 2)	Receiving and Providing Data (RK 512)

* SFB 29 "HS_COUNT" and SFB 30 "FREQ_MES" only exist on the CPU 312 IFM and CPU 314 IFM. SFBs 38 "HSC_A_B" and 39 "POS" only exist on the CPU 314 IFM. For a detailed description, refer to /73/.

1) SFBs 41 "CONT_C," 42 "CONT_S" and 43 "PULSEGEN" only exist on the CPU 314 IFM.

2) SFBs 44 to 49 and 60 to 65 only exist on the S7-300C CPUs.

33.4 List of SFBs, Sorted Alphabetically

Short Name	No.	Function
ALARM	SFB 33	Generate Block-Related Messages with Acknowledgment
ALARM_8	SFB 34	Generate Block-Related Messages without Values for 8 Signals
ALARM_8P	SFB 35	Generate Block-Related Messages with Values for 8 Signals
ANALOG	SFB 44	Positioning with Analog Output
AR_SEND	SFB 37	Send Archive Data
BRCV	SFB 13	Receiving Segmented Data
BSEND	SFB 12	Sending Segmented Data
CONT_C [1)]	SFB 41	Continuous Control
CONT_S [1)]	SFB 42	Step Control
COUNT	SFB 47	Controlling the Counter
CTD	SFB 1	Count Down
CTU	SFB 0	Count Up
CTUD	SFB 2	Count Up/Down
DIGITAL	SFB 46	Positioning With Digital Output
DRUM	SFB 32	Implement a Sequencer
FETCH_RK	SFB 64	Receiving Data (RK 512)
FREQ_MES	SFB 30	Frequency Meter (frequency meter, integrated function)
FREQUENC	SFB 48	Controlling the Frequency Measurement
GET	SFB 14	Read Data from a Remote CPU
HSC_A_B	SFB 38	Counter A/B (integrated function)
HS_COUNT	SFB 29	Counter (high-speed counter, integrated function)
NOTIFY	SFB 36	Generate Block-Related Messages without Acknowledgment Display
POS	SFB 39	Position (integrated function)
PRINT	SFB 16	Send Data to Printer
PULSE	SFB 49	Controlling Pulse Width Modulation
PULSEGEN [1)]	SFB 43	Pulse Generation
PUT	SFB 15	Write Data to a Remote CPU
RALRM	SFB 54	Receiving an Interrupt from a DP Slave
RDREC	SFB 52	Reading a Data Record from a DP Slave
RECV_PTP	SFB 61	Receiving Data (ASCII, 3964(R))
RES_RECV	SFB 62 __	Deleting the Receive Buffer (ASCII, 3964(R))
RESUME	SFB 21	Initiate a Hot Restart on a Remote Device
SEND_PTP	SFB 60	Sending Data (ASCII, 3964(R))
SEND_RK	SFB 63	Sending Data (RK 512)
SERVE_RK	SFB 65	Receiving and Providing Data (RK 512)
START	SFB 19	Initiate a Warm or Cold Restart on a Remote Device
STATUS	SFB 22	Query the Status of a Remote Partner
STOP	SFB 20	Changing a Remote Device to the STOP State
TOF	SFB 5	Generate an Off Delay

Short Name	No.	Function
TON	SFB 4	Generate an On Delay
TP	SFB 3	Generate a Pulse
URCV	SFB 9	Uncoordinated Receiving of Data
USEND	SFB 8	Uncoordinated Sending of Data
USTATUS	SFB 23	Receive the Status of a Remote Device

* SFB 29 "HS_COUNT" and SFB 30 "FREQ_MES" only exist for CPU 312 IFM and CPU 314 IFM. SFBs 38 "HSC_A_B" and 39 "POS" only exist on the CPU 314 IFM. For more information please refer to /73/.

1) SFBs 41 "CONT_C," 42 "CONT_S" and 43 "PULSEGEN" only exist on the CPU 314 IFM. List of FCs

Bibliography

/30/ Getting Started: Working with STEP 7 V5.1

/70/ Manual: *S7-300 Programmable Controller*
Hardware and Installation

/71/ Reference Manual: *S7-300, M7-300 Programmable Controllers*
Module Specifications

/72/ Instructions List: S7-300 Programmable Controller

/73/ Manual: Programmable Controllers,
Integrated Functions CPU 312 IFM/314 IFM

/100/ Manual: S7-400 and M7-400, Programmable Controllers
Hardware and Installation

/101/ Reference Manual: *S7-400, M7-400 Programmable controllers*
Module Specifications

/102/ Instructions List: S7-400 Programmable Controller

/230/ Converter Manual: From S5 to S7

/231/ Manual: Configuring Hardware and Communication Connections,
STEP 7 V5.1

/232/ Reference Manual: Statement List (STL) for S7-300 and S7-400

/233/ Reference Manual: Ladder Logic (LAD) for S7-300 and S7-400

/234/ Manual: Programming with STEP 7 V5.1

/236/ Reference Manual: Function Block Diagram(FBD) for
S7-300 and S7-400

/250/ Manual: Structured Control Language (SCL) for S7-300
and S7-400 Programming

/251/ Manual: S7-*GRAPH for S7-300 and S7-400*,
Programming Sequential Control Systems

/252/ Manual: S7-*HiGraph for S7-300 and S7-400,*
Programming State Graphs

/254/ Manual: *Continuous Functions Charts (CFC) for S7 and M7*
Programming Continuous Function Charts

/270/ Manual: *S7-PDIAG for S7-300 and S7-400*
"Configuring Process Diagnostics for LAD, STL, and FBD"

/350/ User Manual: SIMATIC 7,
Standard Controller

Glossary

Accompanying Value

A value that can be output along with a message and provided information about the status of a variable or and address at the time the message was generated.

ACCU (Accumulator)

Accumulators are registers in the CPU and serve as buffers for load and transfer operations, as well as for comparison, math, and conversion operations.

Actual Parameter

Actual parameters replace formal parameters when a function block (FB) or function (FC) is called, for example, the formal parameter "REQ" is replaced by the actual parameter "I 3.6."

Address

The address is the identifier given to a memory location or range of memory locations, for example: input I 12.1; bit memory MW25; data block DB3.

Addressing

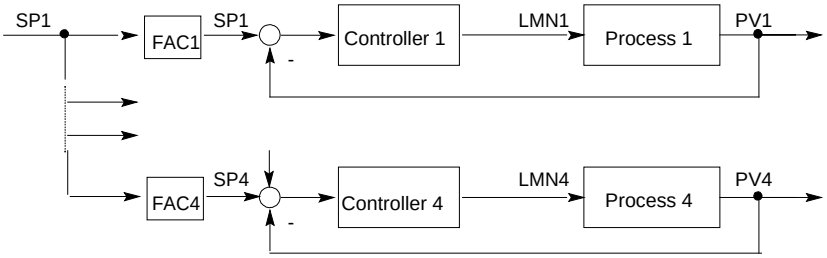
Assigning an address in the user program. Addresses can be assigned to a memory location or range of memory locations (for example: input I 12.1; bit memory MW25).

Bit Memory

This is a 1 bit memory location. Bit memory allows write and read access with STEP 7 basic operations (addressing using bits, bytes, words, and double words). The user can use the bit memory address area to save interim results.

Blending Control

Blending control involves a controller structure in which the setpoint for the total amount SP is converted to percentages of the individual components. The total of the blending factors FAC must be 1 (= 100 %).



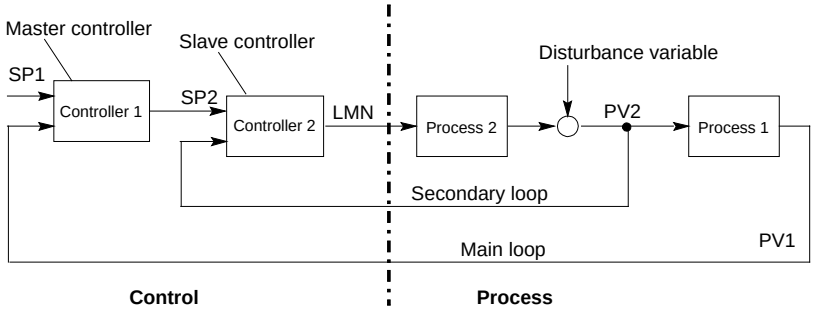
Block-Related Message

A message that is configured for a message-capable block (FB or DB).

Cascade Control

Cascade control involves a series of interconnected controllers, in which the master controller adjusts the setpoint for the secondary (slave) controllers according to the instantaneous error signal of the main process variable.

A cascade control system can be improved by including additional process variables. A secondary process variable PV2 is measured at a suitable point and controlled to the reference setpoint (output of the master controller SP2). The master controller controls the process variable PV1 to the fixed setpoint SP1 and sets SP2 so that the target is achieved as quickly as possible without overshoot.



Closed-Loop Controller

A closed-loop controller is a device in which the error signal is continuously calculated and an actuating signal generated with the aim of eliminating the error signal quickly and without overshoot.

Communication, Bilateral

When using communication SFBs for data exchange, a distinction is made between unilateral and bilateral communication. Communication is bilateral when there is a SFB on the local and the remote module, for example, the communication SFBs "USEND" and "URCV."

Communication SFBs for Configured Connections

Communication SFBs are system function blocks for data exchange and program management.

Examples of data exchange: SEND, RECEIVE, GET.

Examples of program management: setting the CPU of a communication partner to the STOP state, querying the STATUS of the CPU of a communication partner.

Communication SFCs for Non-Configured Connections

Communication SFCs are system functions for data exchange and for aborting existing connections established by the communication SFCs.

Communication, Unilateral

When using communication SFBs for data exchange, a distinction is made between unilateral and bilateral communication. Communication is unilateral when there is a SFB only on the local module, for example, the SFB "GET."

Complete Restart

When a CPU starts up (for example, when the mode selector is moved from STOP to RUN or when power is turned on), before cyclic program processing starts (OB1), either the organization block OB101 (restart; only in the S7-400) or OB100 (complete restart) is processed first. In a complete restart the process-image input table is read in and the STEP 7 user program processed starting with the first statement in OB1.

Constant

"Constants" are token values for constant values in logic blocks. Constants are used to improve the legibility of a program. For example, instead of entering a value directly (for example, 10), the token value "Max_iteration_count" is entered in a function block. The value of the constant (for example, 10) is then entered when the block is called.

Continuous Controller

A continuous controller is a controller in which every change in the error signal produces a change in the manipulated variable. This can adopt any value within the range of the manipulated variable.

Control Device

The entire device used to determine the process variable. It consists of a controller, a controlled device (i.e. actuator), and a sensor (measuring device).

Control Loop

The control loop is the connection between the process output (process variable) and the controller input and between the controller output (manipulated variable) and the process input, so that the controller and process form a closed loop.

Control System Group Message

A group message generated by the CPU operating system when a standard diagnostic event is entered into the diagnostic buffer.

Controller Parameters

Controller parameters are characteristic values for the static and dynamic adaptation of the controller response to the given loop or process characteristics.

CPU Operating System

The CPU operating system organizes all functions and processes of the CPU that are not linked to a special control task.

Data Block (DB)

Data blocks are areas in the user program which contain user data. There are shared data blocks which can be accessed by all logic blocks, and there are instance data blocks which are associated with a particular function block (FB) call.

Diagnostic Buffer

The diagnostic buffer is a memory area in the CPU in which all diagnostic events are stored in the order in which they occurred.

Diagnostic Data

Diagnostic data is information contained in an error message (diagnostic event, time stamp).

Diagnostic Entry

A diagnostic event is described in the diagnostic buffer using a diagnostic entry.

Diagnostic Interrupt

Diagnostic modules report recognized system errors using diagnostic interrupts to the CPU.

Diagnostic Message

The diagnostic message consists of a processed diagnostic event and is sent from the CPU to the display unit.

Diagnostics

Diagnostic functions incorporate all the system diagnostics and include the recognition, interpretation and reporting of errors within the PLC.

Display Device

A device used to display the results of a process.

Error, Asynchronous

Asynchronous errors are run time errors which are not assigned to any particular place in the user program (for example, power supply error, scan time overrun). When these errors occur, the operating system calls the corresponding organization blocks in which the user can program a reaction.

Error Handling with OBs

If the system program recognizes a particular error (for example, access error in S7), it will call the designated organization block in which the CPU's response to the error can be set by the user program.

Error OB

Error OBs are organization blocks which the user can use to program the reaction to an error. However, a programmed reaction to an error is only possible if the error does not cause the PLC to stop. There is an error OB for each type of error. (For example, error OB for addressing error, error OB for access error in S7.)

Error Reaction

Reaction to a run-time error. The operating system can react in the following ways: by changing the PLC to the STOP status, by calling an organization block in which the user can program a reaction, or by displaying the error.

Error, Synchronous

Synchronous errors are run-time errors assigned to a particular place in the user program (for example, error accessing an I/O module). When these errors occur, the operating system calls the corresponding organization blocks in which the user can program a reaction.

Error, System Error

System errors are errors which may occur within a PLC (not in the process). System errors can be, for example program errors in the CPU and faults in modules.

Formal Parameter

A formal parameter is a placeholder for the actual parameter in logic blocks that can be assigned parameters. In FBs and FCs, the formal parameters are declared by the user; in SFBs and SFCs, they already exist. When a block is called, an actual parameter is assigned to the formal parameter so that the called block works with the latest value. The formal parameters belong to the local data of the block and are declared as input, output, and in/out parameters.

Group Error

Error message indicated by a LED display on the front panel of modules (only) in S7-300. The LED lights up whenever there is an error in the module concerned (internal errors and external errors).

Hardware Interrupt

A hardware interrupt is triggered by modules with interrupt capability as a result of a specific event in the process. The hardware interrupt is reported to the CPU. The assigned organization block is then processed according to the priority of this interrupt.

Input Parameter

Input parameters only exist in functions and function blocks. With the help of the input parameters, data are transferred to the called block for processing.

Instruction

An instruction (STEP 5 or STEP 7) is the smallest part of a program created in a textual language. It represents a command for the processor.

Integral Component

Integral component of the controller.

After a step change in the process variable (or error signal) the output variable changes with a ramp function over time at a rate of change proportional to the integral-action factor KI (= 1/TI). The integral component in a closed control loop has the effect of correcting the controller output variable until the error signal becomes zero.

Integrated Controller

An integrated controller is a ready programmed controller block available in the operating system and containing the most important functions of a closed-loop control application. The user can select and deselect functions using software switches.

Interrupt

The SIMATIC S7 priority class system recognizes 10 different priority classes, which regulate the processing of the user program. Interrupts belong to these priority classes, for example, hardware interrupts. When an interrupt occurs, the operating system automatically calls an organization block in which the user can program the required reaction (for example, in a function block).

Interrupt, Time-of-Day

The time-of-day interrupt belongs to one of the priority classes in SIMATIC S7 program execution. It is generated at a specific date (or day) and time (for example, 9:50 or every hour or every minute). A corresponding organization block is then executed.

Interrupt, Time-Delay

The time-delay interrupt belongs to one of the priority classes in SIMATIC S7 program execution. It is generated when a timer has expired in the user program. A corresponding organization block is then executed.

Logic Block

In SIMATIC S7, a logic block is a block that contains part of the STEP 7 user program. The other type of block is a data block which contains only data. The following list shows the types of logic blocks:

- Organization block (OB)
- Function block (FB)
- Function (FC)
- System function block (SFB)
- System function (SFC)

Message

The report of the occurrence of a event. A message can be output to a suitably configured display device and contains information such as priority, location, and time of the message event as well as information about the state transition (entering the state/leaving the state).

Message Configuration

Message configuration refers to the creation and editing of message and message templates with their texts and attributes and concerns such messages as block-related messages, symbol-related messages, and diagnostic messages.

Message Number

A unique number assigned to a message and used to identify it, such as for acknowledgement.

Module Parameter

Module parameters are values with which the behavior of the module can be set. Depending on the particular module, some of these parameters can be modified in the user program.

OB1

The organization block OB1 is the user interface for the system program for cyclic program processing.

OB Priority

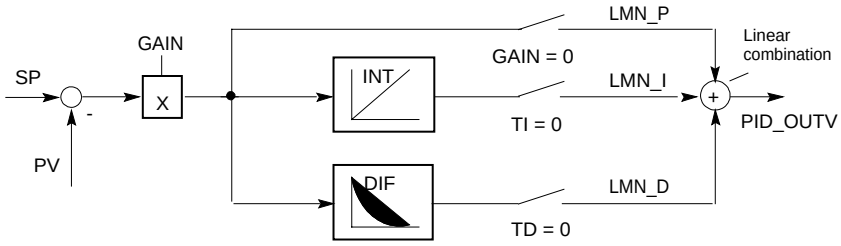
The operating system of the CPU differentiates between various priority classes, for example, cyclic program processing, hardware interrupt- controlled program processing. Organization blocks (OB) are assigned to each priority class, in which the S7 user can program a reaction. The OBs have different priorities, which allow them to be processed in the correct sequence when two occur at the same time and allow OBs with higher priority to interrupt those with lower priority. The S7 user can change the standard priorities.

Organization Block (OB)

Organization blocks form the interface between the CPU operating system and the user program. The sequence in which the user program is processed is specified in the organization blocks.

Parallel Structure

The parallel structure is a special type of signal processing in the controller (mathematical processing). The P, I and D components are calculated parallel to each other with no interaction and then totaled.



Parameter

1. A parameter is a variable of an S7 logic block

   (see block parameter actual parameter formal parameter)

2. A variable for setting the behavior of a module

   (one or more per module)

   Every configurable module has a basic parameter setting when it is supplied from the factory, but this can be changed using STEP 7.

   (one or more per module).

   There are two types of parameter:

   static and dynamic parameters parameter, static/ parameter, dynamic).

Parameter, Dynamic

Dynamic parameters of modules, in contrast to static parameters, can be changed by the user program during operation by calling an SFC, for example, limit values of an analog module.

Parameter, Static

Static parameters of modules, in contrast to dynamic parameters, cannot be changed by the user program, but only using STEP 7, for example, the input delay of a digital input module.

P Algorithm

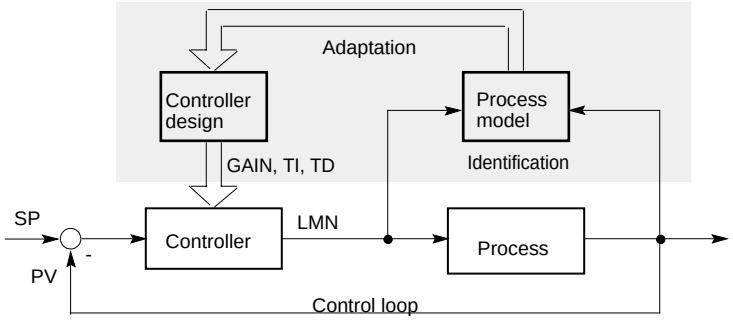
Algorithm for calculating an output signal in which there is a proportional relationship between the error signal and manipulated variable change. Characteristics: steady-state error signal, not to be used with processes including dead time.

PI Algorithm

Algorithm for calculating an output signal in which the change in the manipulated variable is made up of a component proportional to the error signal and an I component proportional to the error signal and time. Characteristics: no steady-state error signal, faster compensation than with an I algorithm, suitable for all processes.

PID Algorithm

Algorithm for calculating an output signal formed by multiplication, integration and differentiation of the error signal. The PID algorithm is a parallel structure. Characteristics: high degree of control quality can be achieved providing the dead time of the process is not greater than the other time constants.



Priority

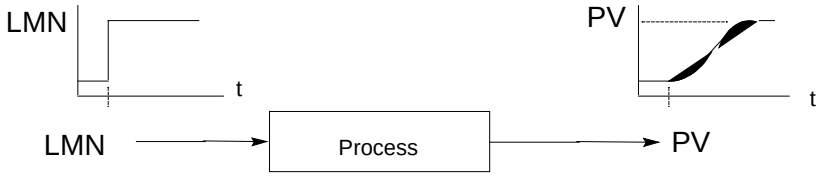
When you assign a priority to an organization block, you determine the interruptability of the currently active user program so that high-priority events interrupt lower-priority events.

Priority Class

The operating system of a CPU has a maximum of 28 priority classes, to which the various organization blocks are assigned. The priority classes decide which OBs can interrupt other OBs. If a priority class includes more than one OB, these do not interrupt each other but are executed sequentially.

Process

The process is the part of the system in which the process variable is influenced by the manipulated variable (by changing the level of energy or mass). The process can be divided into the actuator and the actual process being controlled.



Program Execution, Event-Controlled

With event-controlled program execution, the running of the cyclic user program is interrupted by start events (priority classes). If a start event occurs, the block currently being executed is interrupted before the next instruction and an assigned organization block called and executed. Cyclic program execution then continues from the point of interruption.

Proportional Actuator

Pulse duration modulation

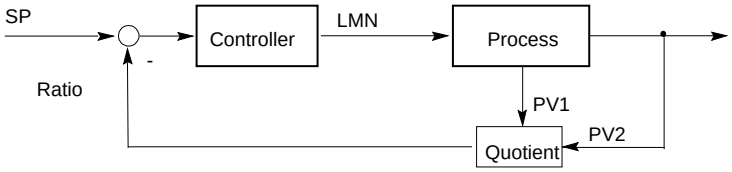
Pulse Duration Modulation

Pulse duration modulation is a method of influencing the manipulated variable at a discontinuous output. The calculated manipulated value as a percentage is converted to a proportional signal pulse time Tp at the manipulated variable output, for example, 100 % Tp = TA or = CYCLE.

Ratio Control

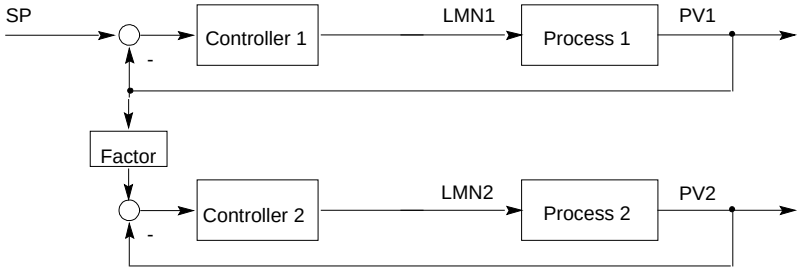
- Single loop ratio controller

  A single loop ratio controller is used when the ratio of two process variables is more important than the absolute values of the variables.



- Multi-loop ratio controller

  In a multi-loop ratio controller, the ratio of the two process variables PV1 and PV2 must be kept constant. To do this, the setpoint of the 2nd control loop is calculated from the process variable of the 1st control loop. Even if the process variable PV1 changes dynamically, the ratio is maintained.



Remote Device

Remote devices are devices, for example, printers or computers that are obtainable on a network. In contrast to local devices, they must be assigned a network address when they are installed.

Restart

When a CPU starts up (for example, when the mode selector is moved from STOP to RUN or when the power is turned on), before cyclic program processing starts (OB1), either the organization block OB100 (complete restart) or the organization block OB101 (restart; only in the S7-400) is processed first. In a restart the process-image input table is read in and the STEP 7 user program processing is restarted at the point where it was interrupted by the last stop (STOP, power off).

Result of Logic Operation (RLO)

The result of logic operation (RLO) is the current signal state in the processor which is used for further binary signal processing. The signal state of the last RLO decides whether or not certain operations are executed.

Run-time Error

Errors which occur during execution of the user program in the PLC (not in the process).

SCAN

An operating system function integrated in the CPU that is used to scan for and detect a signal at set intervals in order to determine if a signal change has occurred.

Standard Function

Standard functions are function blocks available from SIEMENS for implementing complex tasks.

Standard Function Block

Standard function blocks are function blocks available from SIEMENS for implementing complex tasks.

Start Event

Start events are defined events such as errors or interrupts which prompt the operating system to call the appropriate organization block.

Start Event Information

The start event information is part of an organization block (OB). Start event information provides the S7 user with detailed information about the event which triggered the call for the OB. The start event information contains the event number (consisting of event classes and event IDs), an event time stamp, and additional information (for example, the address of the interrupt-activating signal module).

Start Information

When the operating system calls an organization block, the operating system transfers start information which can be interpreted in the user program.

Startup OB

Depending on the setting of the startup mode selector (only S7-400), the reason for the startup (return of power after outage, manual switch from STOP to RUN with the mode selector or command from the programming device) either the startup organization block "Complete restart" or "Restart" (only exists on the S7-400) is called by the operating system. In the startup OB, the SIMATIC S7 user can, for example, program how the system will start up again after a power outage.

An instruction (STEP 5 or STEP 7) is the smallest part of a program created in a textual language. It represents a command for the processor.

Statement List

The Statement List is the assembly language of STEP 7. When a program is processed in STL, the individual instructions correspond to the sequence with which the CPU processes the program.

STEP 7

Programming software for creating user programs for SIMATIC S7 controllers.

STEP 7 Programming Language

Programming language for SIMATIC S7 controllers. The S7 programmer can use STEP 7 in different representation types: a) Statement List, b) Control System Flowchart, c) Ladder Logic.

Step Controller

A step controller is a quasi continuous controller with a discontinuous output (and motor-driven actuator with an I action). The actuator has a three-step response, for example, up - stop - down (or open - hold – close).

(Three-step controller).

STL

Statement List.

Subnumber

The number of the signal to be monitored if a message block can monitor more than one signal.

Symbol-Related Message

A message used in configuring messages for a symbol (input, output, bit memory, data block) in the symbol table. During configuration, the time interval must be set for the SCAN function used to monitor the signal.

Symbolic Programming

The STEP 7 programming language allows the use of symbolic names instead of STEP 7 addresses. For example, a STEP 7 address "Q 1.1" can be replaced with "Valve 17."

The symbol list in STEP 7 also creates the link between the address and the assigned symbolic name.

System Diagnostics

The detection and evaluation of system diagnostic events.

System Diagnostic Event

An entry which is made in the diagnostic buffer of the CPU and is used to initialize operating system.

System Function (SFC)

A system function (SFC) is a function which is integrated in the CPU operating system and can be called in the STEP 7 user program as required.

System Function Block (SFB)

A system function block (SFB) is a function block integrated in the CPU operating system which can be called in the STEP 7 user program when required.

Three-Step Controller

A controller that can only adopt three discrete states; for example, "heat - off cool" or "right - stop - left" (step controller).

Time-Delay Interrupt

The time-delay interrupt belongs to one of the priority classes used in SIMATIC S7 program processing. This interrupt is generated in the user program after a specified time has elapsed and is processed in the associated organization block.

Tool

A software feature used for configuring and programming.

Two-step Controller

A two-step controller is a controller that can only set two states for the manipulated variable (for example, on - off).

User-Defined Diagnostics

The detection and evaluation of user-defined diagnostic events.

User-Defined Diagnostic Event

A diagnostic event detected by the user which can be placed into the diagnostic buffer (with SFC 52).

User-Defined Diagnostic Message

A message reporting the occurrence of a user-defined diagnostic event.

User Program

The user program contains all the statements and declarations and the data for signal processing with which a system or process can be controlled. It is assigned to a programmable module (module, programmable) and can be structured in smaller units known as blocks.

User Program Error

Errors which may occur during the processing of the user program in a SIMATIC S7 PLC (in contrast to process errors). The operating system handles errors using error OBs (priority class system), the status word and output parameters from system functions.

Variable

A variable defines a data with variable contents that can be used in the STEP 7 user program. A variable consists of an address (for example, M 3.1) and a data type (for example, BOOL) and is represented by a symbol (for example, MOTOR_ON).

Variable Declaration

The variable declaration incorporates the entry of a symbolic name, a data type and possibly a default value, address and comment.

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W

X