SIEMENS

SIMATIC S5

S5-135U
Programmable Controller
CPU 921, CPU 922

Manual

Order No. 6ES5998-0UL22
Release 01

Contents

Warnings Information Suggestions/Corrections	C79000-R8576-C216	
Product Summary of CPU 921, CPU 922	C79000-T8576-C216-01	1
Installation Guide	C79000-B8576-C452-04	2
Central Controller S5-135U Instructions	C79000-B8576-C395-01	3
S Processor 921 Instructions R Processor 922 Instructions	C79000-B8576-C262-06 C79000-B8576-C348-05	4
EPROM Submodule 376 RAM Submodule 377 Instructions	C79000-B8576-C615-03	5
923A Coordinator Instructions	C79000-B8576-C263-08	6
923C Coordinator Instructions	C79000-B8576-C049-01	7
Multiprocessor Operation Instructions	C79000-B8576-C048-01	8
Multiprocessor Communication User's Guide	C79000-B8576-C468-05	9
S Processor Programming Guide	C79000-B8576-C264-03	10
R Processor Programming Guide	C79000-B8576-C364-02	11
Space for Pocket Guide S Processor	C79000-B8576-C054-01	12
Space for Pocket Guide CPU 922, CPU 928, CPU 928B	C79000-B8576-C054-01	13
Appendix	C79000-A8576-C260-01	14
		15
		16
		17
		18
		19
		20

The Pocket Guides, *S Processor* (C79000-B8576-C060-03), and *CPU 922, CPU 928 CPU 928B* (6ES5 997-3UA21, Rel. 01) are included in this manual.

Summary of Product Number and Corresponding Documentation for the S5-135U Programmable Controller

The following table shows the relationship between the individual products and the documentation in this S5-135U (CPU 921 and CPU 922) manual with the order no. 6ES5 998-0UL12, release 01.

The names CPU 921 and CPU 922 are not yet used throughout all the instructions.

The following applies: CPU 921 = S processor
CPU 922 = R processor

Note:
The renovated central controller frame described in this manual has the order no. 6ES5 135-3UAxx

Product		Product Documentation
Order No.	Title	
6ES5 135-3UA11 -3UA21 -3UA31 -3UA41 -3UA51	**S5-135U** **Central Controller**	**Instructions:** **(Section 3):** S5-135U Central Controller C79000-B8576-C395-01 **(Section 8):** Multiprocessor Operation in the S5-135U/155U Programmable Controllers C79000-B8576-C500-03 **User's Guide** **(Section 9):** Multiprocessor Communication S5-135U Programmable Controller CPU 922, CPU 928 and CPU 928B S5-155U Programmable Controller CPU 946/947 C79000-B8576-C468-05 **Installation Guide** **(Section 2):** Programmable Controllers of the U Series C79000-B8576-C452-04
6ES5 921-3UA11 -3UA12	**CPU 921**	**Product Summary** **(Section 1):** CPU 921, CPU 922 C79000-T8576-C216-01 **Instructions** **(Section 4):** S Processor 921 C79000-B8500-C262-06 **Programming Guide** **(Section 10):** S5-135U Programmable Controller, S Processor C79000-B8576-C264-03 **Pocket Guide** **(Section 12):** S5-135U Programmable Controller, S Processor C79000-B8576-C060-03

Product		Product Documentation
Order No.	Title	
6ES5 922-3UA11	**CPU 922**	**Instructions** **(Section 4):** R Processor 922 C79000-B8576-C348-05 **Programming Guide** **(Section 11):** S5-135U Programmable Controller, R Processor C79000-B8576-C364-02 **Pocket Guide** **(Section 13):** CPU 922, CPU 928, CPU 928B 6ES5 997-3UA21, Rel. 01
6ES5 376-0AA11 -0AA21 -0AA31 6ES5 377-0AA11 -0AA21 -0AA32 -0AA31	**376 EPROM Submodule** **377 RAM Submodule** **(with back-up)**	**Instructions** **(Section 5):** 376 EPROM Submodule 377 RAM Submodule C79000-B8576-C615-03
6ES5 923-3UA11	**923A Coordinator**	**Instructions** **(Section 6):** 923A Coordinator C79000-B8576-C263-08
6ES5 923-3UC11	**923C Coordinator**	Instructions **(Section 7):** 923C Coordinator C79000-B8576-C349-07

Warning

Risks involved in the use of so-called SIMATIC-compatible modules of non-Siemens manufacture

"The manufacturer of a product (SIMATIC in this case) is under the general obligation to give warning of possible risks attached to his product. This obligation has been extended in recent court rulings to include parts supplied by other vendors. Accordingly, the manufacturer is obliged to observe and recognize such hazards as may arise when a product is combined with products of other manufacture.

For this reason, we feel obliged to warn our customers who use SIMATIC products not to install so-called SIMATIC-compatible modules of other manufacture in the form of replacement or add-on modules in SIMATIC systems.

Our products undergo a strict quality assurance procedure. We have no knowledge as to whether outside manufacturers of so-called SIMATIC-compatible modules have any quality assurance at all or one that is nearly equivalent to ours. These so-called SIMATIC- compatible modules are not marketed in agreement with Siemens; we have never recommended the use of so-called SIMATIC-compatible modules of other manufacture. The advertising of these other manufacturers for so-called SIMATIC-compatible modules wrongly creates the impression that the subject advertised in periodicals, catalogues or at exhibitions had been agreed with us. Where so-called SIMATIC-compatible modules of non-Siemens manufacture are combined with our SIMATIC automation systems, we have a case of our product being used contrary to recommendations. Because of the variety of applications of our SIMATIC automation systems and the large number of these products marketed worldwide, we cannot give a concrete description specifically analyzing the hazards created by these so-called SIMATIC-compatible modules. It is beyond the manufacturer's capabilities to have all these so-called SIMATICcompatible modules checked for their effect on our SIMATIC products. If the use of so-called SIMATIC-compatible modules leads to defects in a SIMATIC automation system, no warranty for such systems will be given by Siemens.

In the event of product liability damages due to the use of so-called SIMATIC-compatible modules, Siemens are not liable since we took timely action in warning users of the potential hazards involved in so-called SIMATIC-compatible modules."

Safety-Related Guidelines for the User

1 General

This manual provides the information required for the intended use of the particular product. The documentation is written for technically qualified personnel such as engineers, programmers or maintenance specialists who have been specially trained and who have the specialized knowledge required in the field of instrumentation and control.

A knowledge of the safety instructions and warnings contained in this manual and their appropriate application are prerequisites for safe installation and commissioning as well as safety in operation and maintenance of the product described. Only qualified personnel as defined in section 2 have the specialized knowledge that is necessary to correctly interpret the general guidelines relating to the safety instructions and warnings and implement them in each particular case.

This manual is an inherent part of the scope of supply even if, for logistic reasons, it has to be ordered separately. For the sake of clarity, not all details of all versions of the product are described in the documentation, nor can it cover all conceivable cases regarding installation, operation and maintenance. Should you require further information or face special problems that have not been dealt with in sufficient detail in this documentation, please contact your local Siemens office.

We would also point out that the contents of this product documentation shall not become a part of or modify any prior or existing agreement, commitment or legal relationship. The Purchase Agreement contains the complete and exclusive obligations of Siemens. Any statements contained in this documentation do not create new warranties or restrict the existing warranty.

2 Qualified Personnel

Persons who are **not qualified** should not be allowed to handle the equipment/system. Non-compliance with the warnings contained in this manual or appearing on the equipment itself can result in severe personal injury or damage to property. Only **qualified personnel** should be allowed to work on this equipment/system.

Qualified persons as referred to in the safety guidelines in this manual as well as on the product itself are defined as follows:

- System planning and design engineers who are familiar with the safety concepts of automation equipment;
- Operating personnel who have been trained to work with automation equipment and are conversant with the contents of the manual in as far as it is connected with the actual operation of the plant;
- Commissioning and service personnel who are trained to repair such automation equipment and who are authorized to energize, deenergize, clear, ground and tag circuits, equipment and systems in accordance with established safety practices.

3 Danger Notices

The notices and guidelines that follow are intended to ensure personal safety, as well as protecting the product and connected equipment against damage.

The safety notices and warnings for protection against loss of life (the users or service personnel) or for protection against damage to property are highlighted in this manual by the terms and pictograms defined here. The terms used in this manual and marked on the equipment itself have the following significance:

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 or property damage can result if proper precautions are not taken.

Note

is an important information about the product, its operation or a part of the manual to which special attention is drawn.

Important

If in this manual "Important" should appear in bold type, drawing attention to any particularly information, the definition corresponds to that of "Warning", "Caution" or "Note".

4 Proper Usage

- The equipment/system or the system components may only be used for the applications described in the catalog or the technical description, and only in combination with the equipment, components and devices of other manufacturers as far as this is recommended or permitted by Siemens.
- The product described has been developed, manufactured, tested and the documentation compiled in keeping with the relevant safety standards. Consequently, if the described handling instructions and safety guidelines described for planning, installation, proper operation and maintenance are adhered to, the product, under normal conditions, will not be a source of danger to property or life.

Warning

- After opening the housing or the protective cover or after opening the system cabinet, certain parts of this equipment/system will be accessible, which could have a dangerously high voltage level.
- Only suitably qualified personnel should be allowed access to this equipment/system.
- These persons must be fully conversant with any potential sources of danger and maintenance measures as set out in this manual.
- It is assumed that this product be transported, stored and installed as intended, and maintained and operated with care to ensure that the product functions correctly and safely.

© Siemens AG C79000-D8076-C338-03

5 Guidelines for the Planning and Installation of the Product

The product generally forms a part of larger systems or plants. These guidelines are intended to help integrate the product into its environment without it constituting a source of danger.

The following facts require particular attention:

Note

Even when a high degree of safety has been designed into an item of automation equipment by means of multichannel configuration, it is still imperative that the instructions contained in this manual be exactly adhered to. Incorrect handling can render ineffective the preventive measures incorporated into the system to protect it against dangerous faults, and even create new sources of danger.

The following advice regarding installation and commissioning of the product should - in specific cases - also be noted.

Warning

- Follow strictly the safety and accident prevention rules that apply in each particular case.
- Units which are designed as built-in units may only be operated as such, and table-mounted or portable equipment only with its casing closed.
- In the case of equipment with a permanent power connection which is not provided with an isolating switch and/or fuses which disconnect all poles, a suitable isolating switch or fuses must be provided in the building wiring system (distribution board). Furthermore, the equipment must be connected to a protective ground (PE) conductor.
- For equipment or systems with a fixed connecting cable but no isolating switch which disconnects all poles, the power socket with the grounding pin must be installed close to the unit and must be easily accessible.
- Before switching on the equipment, make sure that the voltage range setting on the equipment corresponds to the local power system voltage.
- In the case of equipment operating on 24 V DC, make sure that proper electrical isolation is provided between the mains supply and the 24 V supply. Only use power supply units to IEC 364-4-41 or HD 384.04.41 (VDE 0100 Part 410).
- Fluctuations or deviations of the power supply voltage from the rated value should not exceed the tolerances specified in the technical specifications. Otherwise, functional failures or dangerous conditions can occur in the electronic modules/equipment.
- Suitable measures must be taken to make sure that programs that are interrupted by a voltage dip or power supply failure resume proper operation when the power supply is restored. Care must be taken to ensure that dangerous operating conditions do not occur even momentarily. If necessary, the equipment must be forced into the "emergency off" state.
- Emergency tripping devices in accordance with EN 60204/IEC 204 (VDE 0113) must be effective in all operating modes of the automation equipment. Resetting the emergency off device must not result in any uncontrolled or undefined restart of the equipment.

Caution

- Install the power supply and signal cables in such a manner as to prevent inductive and capacitive interference voltages from affecting the automation functions.
- Automation equipment and its operating elements must be installed in such a manner as to prevent unintentional operation.
- Automation equipment can assume an undefined state in the case of a wire break in the signal lines. To prevent this, suitable hardware and software measures must be taken when interfacing the inputs and outputs of the automation equipment.

6 Active and Passive Faults in Automation Equipment

- Depending on the particular task for which the electronic automation equipment is used, both **active** as well as **passive** faults can result in a **dangerous** situation. For example, in drive control, an active fault is generally dangerous because it can result in an unauthorized startup of the drive. On the other hand, a passive fault in a signalling function can result in a dangerous operating state not being reported to the operator.
- This differentiation of the possible faults and their classification into dangerous and non-dangerous faults, depending on the particular task, is important for all safety considerations in respect of the product supplied.

Warning

In all cases where a fault in an automation equipment can result in severe personal injury or substantial damage to property, ie. where a dangerous fault can occur, additional external measures must be taken or equipment provided to ensure or force safe operating conditions even in the event of a fault (e.g. by means of independent limit monitors, mechanical interlocks etc.).

7 Procedures for Maintenance and Repair

If measurement or testing work is to be carried out on an active unit, the rules and regulations contained in the "VBG 4.0 Accident prevention regulations" of the German employers liability assurance association (Berufsgenossenschaften) must be observed. Particular attention is drawn to paragraph 8 "Permissible exceptions when working on live parts". Use only suitable electrical tools.

Warning

- Repairs to an item of automation equipment may only be carried out by **Siemens service personnel** or an **authorized Siemens repair center.** For replacement purposes, use only parts or components that are contained in the spare parts list or listed in the "Spare parts" section of this manual. Unauthorized opening of equipment and improper repairs can result in loss of life or severe personal injury as well as substantial property damage
- Before opening the equipment, always remove the power plug or open the disconnecting switch.
- Only use the fuse types specified in the technical specifications or the maintenance instructions of this manual.
- Do not throw batteries into an open fire and do not carry out any soldering work on batteries (danger of explosion). Maximum ambient temperature 100°C. Lithium batteries or batteries containing mercury should not be opened or recharged. Make sure that the same type is used when replacing batteries.
- Batteries and accumulators must be disposed of as classified waste.
- The following points require attention when using monitors:
  Improper handling, especially the readjustment of the high voltage or fitting of another tube type can result in excessive X-ray radiation from the unit. The license to operate such a modified unit automatically lapses and the unit must not be operated at all.

The information in this manual is checked regularly for updating and correctness and may be modified without prior notice. The information contained in this manual is protected by copyright. Photocopying and translation into other languages is not permitted without express permission from Siemens.

© Siemens AG C79000-D8076-C338-03

Guidelines for Handling Electrostatically Sensitive Devices (ESD)

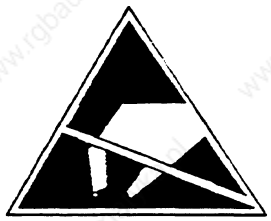
1 What is ESD?

VSLI chips (MOS technology) are used in practically all SIMATIC S5 and TELEPERM M modules. These VLSI components are, by their nature, very sensitive to overvoltages and thus to electrostatic discharge:

They are therefore defined as
"Electrostatically Sensitive Devices"

"ESD" is the abbreviation used internationally.

The following warning label on the cabinets, subracks and packing indicates that electrostatically sensitive components have been used and that the modules concerned are susceptible to touch:



ESDs can be destroyed by voltage and energy levels which are far below the level perceptible to human beings. Such voltages already occur when a component or a module is touched by a person who has not been electrostatically discharged. Components which have been subjected to such overvoltages cannot, in most cases, be immediately detected as faulty; the fault occurs only after a long period in operation.

An electrostatic discharge
- of 3500 V can be felt
- of 4500 V can be heard
- must take place at a minimum of 5000 V to be seen.

But just a fraction of this voltage can already damage or destroy an electronic component.

The typical data of a component can suffer due to damage, overstressing or weakening caused by electrostatic discharge; this can result in temporary fault behavior, e.g. in the case of

- temperature variations,
- mechanical shocks,
- vibrations,
- change of load.

Only the consequent use of protective equipment and careful observance of the precautions for handling such components can effectively prevent functional disturbances and failures of ESD modules.

© Siemens AG C79000-D8076-C333-01

2 When is a Static Charge Formed?

One can never be sure whether the human body or the material and tools which one is using are not electrostatically charged.

Small charges of 100 V are very common; these can, however, very quickly rise up to 35 000 V.

Examples of static charge:

– Walking on a carpet	up to 35 000 V
– Walking on a PVC flooring	up to 12 000 V
– Sitting on a cushioned chair	up to 18 000 V
– Plastic desoldering unit	up to 8 000 V
– Plastic coffee cup	up to 5 000 V
– Plastic bags	up to 5 000 V
– Books, etc. with a plastic binding	up to 8 000 V

3 Important Protective Measures against Static Charge

- Most plastic materials are highly susceptible to static charge and must therefore be kept as far away as possible from ESDs.
- Personnel who handle ESDs, the work table and the packing must all be carefully grounded.

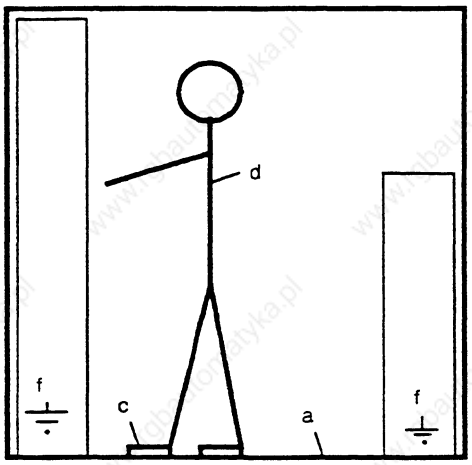
4 Handling of ESD Modules

- One basic rule to be observed is that electronic modules should be touched by hand only if this is necessary for any work required to be done on them. Do not touch the component pins or the conductors.
- Touch components only if
  - the person is grounded at all times by means of a wrist strap

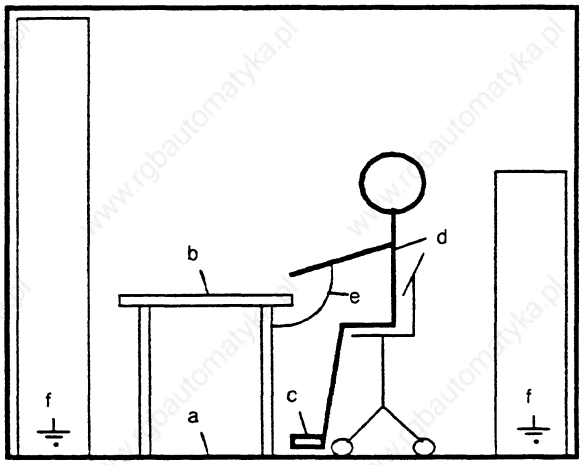
  or
  - the person is wearing special anti-static shoes or shoes with a grounding strip.
- Before touching an electronic module, the person concerned must ensure that (s)he is not carrying any static charge. The simplest way is to touch a conductive, grounded item of equipment (e.g. a blank metallic cabinet part, water pipe, etc.) before touching the module.
- Modules should not be brought into contact with insulating materials or materials which take up a static charge, e.g. plastic foil, insulating table tops, synthetic clothing, etc.
- Modules should only be placed on conductive surfaces (table with anti-static table top, conductive foam material, anti-static plastic bag, anti-static transport container).
- Modules should not be placed in the vicinity of monitors, TV sets (minimum distance from screen > 10 cm).

© Siemens AG C79000-D8076-C333-01

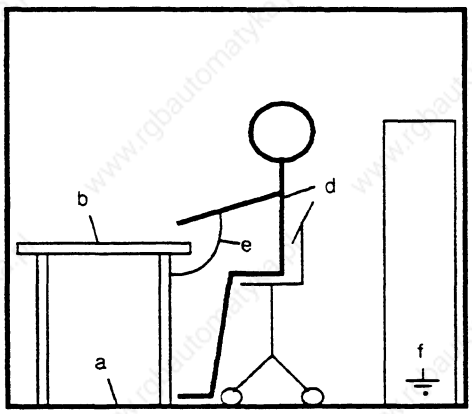
The diagram below shows the required protective measures against electrostatic discharge.



Standing position



Standing/sitting position



Sitting position

a Conductive flooring
b Anti-static table
c Anti-static shoes
d Anti-static coat
e Grounding wrist strap
f Grounding connection of the cabinets

5 Measurements and Modification to ESD Modules

- Measurements on modules may only be carried out under the following conditions:
  - The measuring equipment is grounded (e.g. via the PE conductor of the power supply system) or
  - when electrically isolated measuring equipment is used, the probe must be discharged (e.g. by touching the metallic casing of the equipment) before beginning measurements.
- Only grounded soldering irons may be used.

6 Shipping of ESD Modules

Anti-static packing material must always be used for modules and components, e.g. metalized plastic boxes, metal boxes, etc. for storing and dispatch of modules and components.

If the container itself is not conductive, the modules must be wrapped in a conductive material such as conductive foam, anti-static plastic bag, aluminium foil or paper. Normal plastic bags or foils should not be used under any circumstances.

For modules with built-in batteries ensure that the conductive packing does not touch or short-circuit the battery connections; if necessary cover the connections with insulating tape or material.

SIEMENS

SIMATIC S5

Product Summary for CPU 921 and CPU 922 C79000-T8576-C216-01

www.rgbautomatyka.pl

Product Summary of CPU 921 and CPU 922

The functions and characteristic features of the CPU 921 and CPU 922 are summarized in this section. General performance features and fields of application are described briefly; furthermore, the programming languages, which can be used, are listed.

Performance features

In the S5-135U programmable controller the CPU 921 can be used in single as well as in multiprocessor operation. The CPU 922 can be used in the S5-135U and S5-155U programmable controllers in single or multiprocessor operation.

The CPU 921 (S processor) is designed for binary signal processing (open-loop control tasks).

The CPU 922 (R processor) is designed for word processing (computing and closed-loop control). Binary signal processing is also possible.

The most outstanding features of both CPUs are:

- a plug-in RAM or EPROM submodule with a maximum of 64 kbytes for instructions and data of the user program (for CPU 921 and CPU 922).
- a 7.4 kbyte integrated data memory (DB-RAM) for data of the user program (for CPU 921),

  or

  a 22 kbyte integrated data memory (DB-RAM) for data of the user program (for CPU 922).
- - 2048 flags
  - 128 timers
  - 128 counters
- Processing from up to 4096 binary inputs/outputs each
  192 analog inputs/outputs each

Programming language and program processing

Use the STEP 5 programming language /1/ to program the CPU 921 and the CPU 922. The types of representation of this programming language are the ladder diagram (LAD), control system flowchart (CSF) and statement list (STL). Furthermore GRAPH 5 is available which is used to program sequential controls (step sequences).

The CPU 921 and the CPU 922 allow user programs which are

- cyclic
- alarm-controlled
- time-controlled (in 100 ms clock pulse)

The average processing time of 1 K instructions is:

	CPU 921	CPU 922
– Binary instructions	1.3 ms	20 ms
– 1K word commands (load - transfer)	45 ms	20 ms

You will find detailed information on programming the CPU 921 and CPU 922 in the Programming Instructions in Part 10/11 of this manual. All operations and their running times are described in the Pocket Guides included with this manual.

SIEMENS

SIMATIC S5

Programmable Controllers of the U Series

Installation Guide C79000-B8576-C452-04

Page

1 Introduction to Use of Installation Guidelines ... 3

2 Fundamentals of EMC ... 4

3 Selection and Design of Cabinets ... 6

3.1 Selection Criteria ... 6

3.2 Types of Cabinets ... 6

3.3 Specifications When Designing a Cabinet ... 8

3.4 Power Loss ... 10

3.4.1 Power Loss in Cabinet and Cabinet Cooling ... 10
3.4.2 Example of Calculating Cabinet Type ... 11

3.5 Example of a Cabinet Design ... 12

4 Design and Connection of Power Supplies ... 15

4.1 Internal Power Supply for Central Controllers and Expansion Units ... 15

4.2 Load Power Supply ... 16

4.3 Electrical Design with Process Peripherals ... 17

4.3.1 Power Supply for CCs, EUs and Process Peripherals from Grounded Battery or Grounded Power Supply Units ... 18
4.3.2 Power Supply for CCs, EUs and Process Peripherals from Centrally Grounded Battery or Centrally Grounded Power Supply Units ... 19
4.3.3 Power Supply for CCs, EUs and Process Peripherals from Non-Grounded Battery or Non-Grounded Power Supply Units ... 20

4.4 Load Power Supply from Two Power Supply Units ... 21

4.4.1 Non-floating Modules ... 21
4.4.2 Floating Modules ... 22

5 Wiring Layout ... 23

5.1 Wiring Layout Inside a Cabinet ... 23

5.2 Wiring Layout Outside Cabinets ... 24

5.3 Wiring Layout Outside Buildings ... 25

5.4 Equipotential Bonding ... 25

6	Cabinet Wiring and Design with Respect to EMC	26
6.1	Grounding of Inactive Metal Components	27
6.2	Shielding of Devices and Cables	28
6.3	Use of Special Noise Suppression Measures	30
6.4	Example of an EMC-compatible Cabinet Design	32
6.5	Checklist for EMC-compatible Cabinet Design	34
7	Framework and Wall Mounting of SIMATIC S5 Controllers	36
8	Lightning Protection Measures	38
9	Safety Measures	39
9.1	Protection against Indirect Contact	39

1 Introduction to Use of Installation Guidelines

This document is intended for planning, installation and commissioning engineers.

The installation guidelines are divided into the following sections:

- **Chapter 1** Introduction to Use of Installation Guidelines
- **Chapter 2** Fundamentals of EMC
  This section provides a summary of the rules you must observe to ensure electromagnetic compatibility.
- **Chapter 3** Selection and Design of Cabinets
  This section lists criteria which must be considered when selecting the cabinet. The conditions resulting from the power loss of the modules used and the ambient temperature are considered in particular. The power losses of SIMATIC modules are listed.
- **Chapter 4** Design and Connection of Power Supplies
  This section provides information you must observe for the electrical connection of the power supply to CCs, EUs and process peripherals.
- **Chapter 5** Wiring Layout
  This section describes how you can achieve a high interference-resistance of your programmable controller by using a correct wiring layout.
- **Chapter 6** Cabinet Wiring and Design with Respect to EMC
  This section describes the measures required to ensure EMC of your programmable controller. It shows how you can prevent fundamental errors when designing and wiring cabinets. A checklist is provided to check the EMC-compatible cabinet design.
- **Chapter 7** Framework and Wall Mounting
  This section describes what you must observe if you fit your SIMATIC controller in a framework or on a wall.
- **Chapter 8** Lightning Protection Measures
  This section provides information about the measures you should take to protect outdoor cables and lines for SIMATIC devices from lightning strikes.
- **Chapter 9** Safety Measures
  This section provides a summary of the measures you must always take when planning the use of programmable controllers in order to prevent danger during operation. The regulations CENELEC HD 384.4.41 (IEC 364-4-41) (VDE 0100) and EN 60 204 (IEC 204-1) (VDE 0113) must be applied in order to carry out these measures.

We recommend that users who are using a SIMATIC S5 controller for the first time follow the installation guidelines right from the beginning when planning the control system.
We strongly recommend that all users particularly observe the sections and paragraphs concerned with preventing danger (especially Chapter 9) and protection from sources of error (especially Chapter 6). Even if you are an experienced user, check your design using the checklist in Chapter 6.

2 Fundamentals of EMC

Definition of EMC

Electromagnetic compatability (EMC) means that an electrical device is able to function correctly in a defined electromagnetic environment without disturbing other devices in its vicinity.

It is frequently sufficient to observe a few elementary rules to achieve electromagnetic compatibility (EMC). It is essential for you to observe the following four rules when installing your programmable controller.

Rule 1: Make sure there is a perfectly functioning reference ground (central grounding point)

- Connect the central controller and expansion units to the central grounding point in a star-shaped configuration without loops.
- Protect the PLC from external influences by installing it in a cabinet or housing. Incorporate the cabinet or housing into the ground system.
- Shield electromagnetic fields resulting from inductors (transformers, motors, contactor coils) from the PLC using barriers (steel, highly permeable material).
- Use metal plug housings (not plastic) for screened data transmission lines.

Rule 2: Use a large-area ground.

- Connect all inactive metal components with a large-area contact and a low impedance.
- Establish a central connection between the inactive metal components and the central grounding point.
- The screw connections on inactive, painted metal components should be made using NOMEL contact washers [1].
- Do not forget to incorporate the screen bar into the ground system. This means that the screen bar itself must be connected to ground via a large-area contact.
- Aluminium components are unsuitable for grounding.

[1] Contact washer Siemens standard 70093 available from
- Siemens ANL A443 Werkzeug 8520 Erlangen
- Teckentrup GmbH und Co. KG, Postfach 120, D-5974 Herscheid 2,
- NOMEL S.A. Tour Franklin, Cedex 11, F-92081 Paris.
- or from your local Siemens representative

Rule 3: Plan the wiring layout and ensure that the plan is kept to

- Divide the cables into groups and route them separately.
  (power cables, power supply cables, signal lines, data lines)
- Always route power cables and signal cables in separate ducts or bundles.
- All the cables should only be fed into the cabinet from one side.
- Route the signal cables as close as possible to grounded components (e.g. cabinet members).
- We recommend the twisting of the forward and return lines of individually routed cables.

Rule 4: Ensure that your cables are well shielded

- Data transmission cables should be screened and connected at both ends.
- Analog cables should be screened and the screen connected at one or both ends.
- The cable screens must be connected at the cabinet inlet to the screen bar using a large-area contact and secured with clamps.
- Route the screen up to the module without interruptions.

3 Selection and Design of Cabinets

3.1 Selection Criteria

The following criteria must be observed when selecting and dimensioning a cabinet:

(i) Ambient conditions

(ii) Quantity and type of power supplies and subracks to be used

(iii) Total power loss of components present in the cabinet.

The ambient conditions present where the cabinet is located (temperature, humidity, dust, chemical influences) define the required degree of protection of the cabinet (IP XX) as shown in Fig. 1. Further information on degrees of protection can be found in IEC 529 and DIN 40050.

The required design of the cabinet is described in Section 3.3. Make sure that the maximum ambient temperature for the modules is not exceeded.
This involves both the ambient temperature outside the cabinet and the power loss in the cabinet.
It may be necessary to provide a fan or heat exchanger if the power loss is too high. A typical cabinet design is shown using an example at the end of the section.

3.2 Types of Cabinets

The following diagram shows a summary of the most common types of cabinet. It also shows the principle of heat dissipation used, the maximum achievable heat dissipation and the degree of protection.

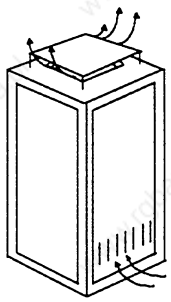
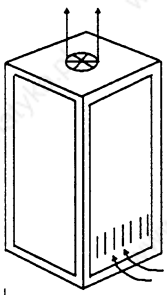
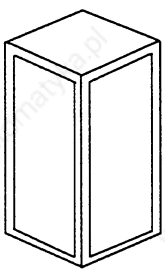
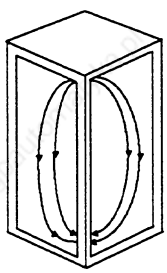
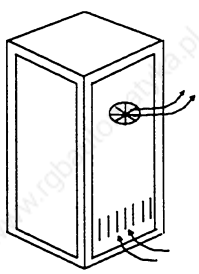
Open cabinets			**Closed cabinets**	
Through-ventilation by natural convection	Increased through-ventilation using external fan (with filter)	Natural convection	Forced circulation using fan assembly, improvement of natural convection	Forced circulation using heat exchanger[2], external ventilation inside and outside
				
Heat dissipation primarily by natural thermal convection, small portion via the cabinet wall.	Increased heat dissipation through increased air movement.	Heat dissipation only through the cabinet wall; only low power loss permissible. Heat accumulation usually occurs in the top of the cabinet.	Heat dissipation only through the cabinet wall. Forced ventilation of the internal air results in improved heat dissipation and prevention of heat accumulation.	Heat dissipation through exchange between heated internal air and cool outside air. The increased surface area of the heat exchanger and the forced circulation between the inside and outside air permit good heat dissipation.
Temperature difference between ambient temperature and cabinet temperature (measured at top in the cabinet): 20 °C[4]				
Power loss P[3] with cabinet dimensions of 2200 mm x 600 mm x 600 mm				
Installation as single unit:				
Up to approx. 700 W	Up to approx. 2700 W (approx. 1400 W with very fine filter)	Up to approx. 260 W	Up to approx. 360 W	Up to approx. 1700 W
Degree of protection IP 20[1]	Degree of protection IP 20[1]	Degree of protection IP 54[1]	Degree of protection IP 54[1]	Degree of protection IP 54[1]

Fig. 1 Types of cabinets

1) The location and the ambient conditions present there are decisive for selection of the type of cabinet protection (see IEC 529 and DIN 40050).
2) See Catalog NV21 for heat exchangers.
3) The values only apply if the guidelines for installation are adhered to (for further details refer to the following section).
4) If other temperature differences are present, refer to the temperature characteristics of the cabinet manufacturer.

3.3 Specifications When Designing a Cabinet

You must first define the components to be fitted in the cabinet. Then calculate the total power loss of the individual components. The following specifications must be observed:

- The expansion units can be accommodated together with the respective central controller in one cabinet, or also in several cabinets (centralized or distributed). See Section 2 for the installation dimensions of the subracks.

- As a result of the required spacing between devices and the maximum permissible installation height for control elements, a maximum of three U-type devices can be arranged one above the other (see Fig. 2).

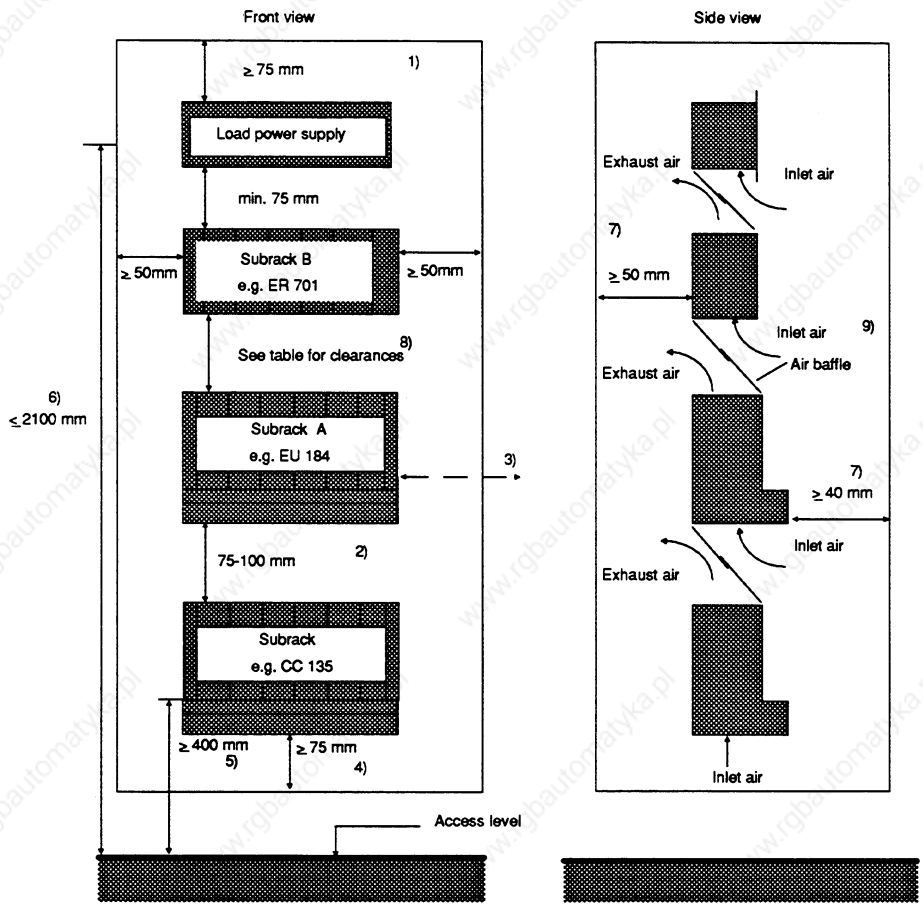


Fig. 2 Installation dimensions for SIMATIC controllers in cabinet

1) Min. 75 mm with closed cabinet roof. Smaller distances are possible with a perforated cabinet roof and an additional, separate ventilation roof.
2) Min. 75 mm space for inlet air and exhaust air, max. 100 mm because of cable length between the CC/EU interface modules.
3) Max. spacing of 400 mm possible (min. 50 mm) when connecting devices next to one another (with IM 312).
4) Min. 75 mm from obstructions (large equipment) in the inlet air area.
5) Min. installation height above access level 400 mm for control elements, 200 mm for connections.
6) Max. installation height for control elements: 2100 mm to VDE 0106, Part 100, 2000 mm to EN 60 204 (IEC 204-1) (VDE 0113).
7) Space for air circulation (400 mm deep cabinets are sufficient).
8) See Table 2-1 for the distances between subracks A and B.
9) The installation of air baffles is recommended to provide a better air supply.

Note:

The expansion unit with the largest power loss should be positioned as the top unit.

- If subracks are combined (CC and EU), the clearances listed in Table 1 must be observed.

Subrack A [1]	Subrack B [1]	Minimum clearance	Maximum clearance
S5-135U/155U or S5-115U or S5-100U	S5-135U	75 mm	Limited by the length of the connection cables to the interface modules
	S5-115U with fan	60 mm	
	S5-115U without fan	100 mm	
	S5-100U	75 mm	

Table 1 Required clearance between subracks

[1] See Fig. 2, Installation dimensions for SIMATIC controllers in cabinet

Note:

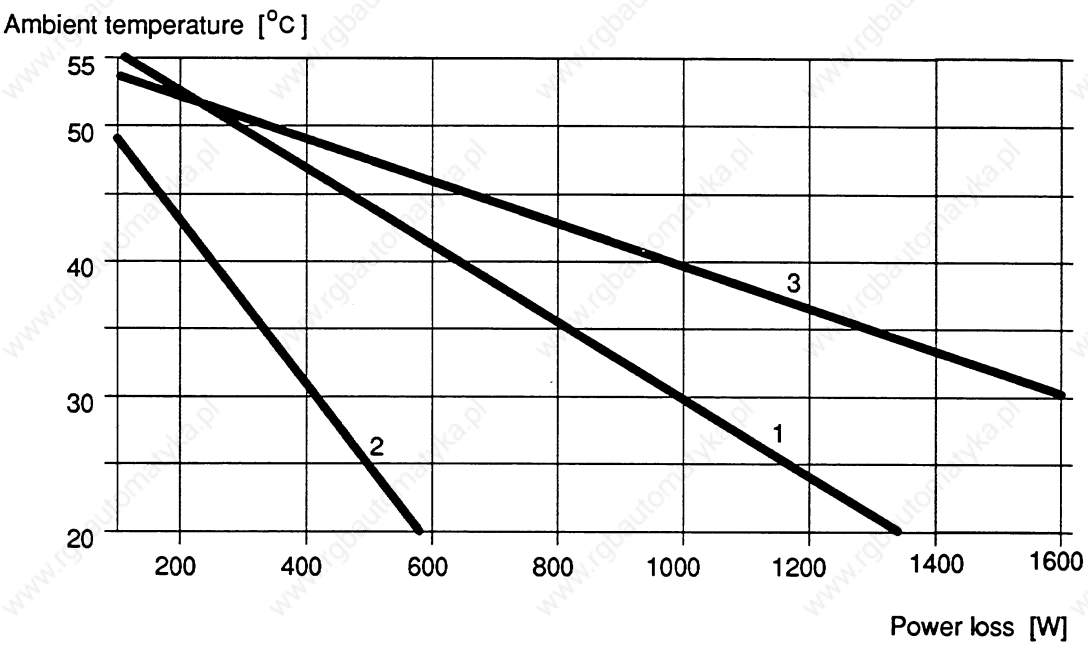
If subracks from the S5-135U/155U series are used together with subracks of the S5-115U in the same cabinet, ensure that the rear panels of the subracks have the same clearance to the rear panel of the cabinet. This results in improved air circulation.

3.4 Power Loss

3.4.1 Power Loss in Cabinet and Cabinet Cooling

The power loss that can be dissipated from a cabinet depends on the cabinet design, its ambient temperature and the arrangement of units in the cabinet.

Fig. 3 shows the permissible ambient temperature of a cabinet with dimensions of 600 mm x 600 mm x 2200 mm depending on the power loss. The values indicated only apply to the arrangement of units in the cabinet as shown in Fig. 2. You can obtain more information from Catalogs NV21 and ET1.



Curve 1 Cabinet (open) with through-ventilation by natural convection.
Curve 2 Closed cabinet with natural convection and internal forced circulation using fan.
Curve 3 Closed cabinet with heat exchanger. Heat exchanger size 11/6 (920 mm x 460 mm x 111 mm).

Fig. 3 Maximum cabinet environment temperature depending on the power loss

Note:

When fitting the subracks of the S5-135U/155U series, the maximum power loss which can be dissipated by the fans must not be exceeded.
The max. dissipated power loss per unit with an inlet temperature of 55 °C is 250 W. This value is increased by 20 W for each reduction in the inlet temperature by 1 °C.

Caution:

Modules with a hard disk drive can only be used up to an ambient temperature of 50 °C.

3.4.2 Example of Calculating Cabinet Type

The following example shows the maximum permissible ambient temperature for different types of cabinet with the same power loss.

Example:

The following configuration is present:

1 central controller	200 W
2 expansion units, each with 250 W power loss	500 W
1 load power supply, 24 V/40 A, 6EV1 362-5BK00 (full load)	200 W
Total power loss:	900 W

Fig. 3 shows the max. ambient temperatures for a total power loss of 900 W:

Cabinet design (see Fig. 1)	Max. ambient temperature
Closed, with natural convection and forced circulation	(use not possible)
Open with through-ventilation	Approx. 33 °C
Closed, with heat exchanger	Approx. 42 °C
Framework/wall	Max. 55 °C

The power losses of the modules can be found in the technical data in the catalogs or in the manuals.
If these values are not listed in the technical data, they can be calculated easily from the power consumption. To do this, multiply the value of the power consuption by the appropriate voltage.

Examples:

- CPU 928B: power consumption 4 A/5 V ⟶ power loss = 20 W
- CP 143: power consumption 4 A/5 V, 0.5 A/15 V, 0.04 A/24 V ⟶ power loss approx. 21 V
- IM 304 power consumption 1.5 A/5 V ⟶ power consumption = 7.5 W

3.5 Example of a Cabinet Design

Figs. 4 and 5 show a design using the example of a metric 8MF cabinet (2200 mm x 600 mm x 600 mm). This design has a number of advantages:

- Universal application
- Independent of the cabinet width (550 to 1200 mm possible)
- The units can be installed asymmetrically; you thus gain more space on one side for routing signal cables.
- All devices can be installed and removed from the front, even after initial installation. The M6 screws must be premounted on the 19-inch cabinet member at the correct mounting height. You can then hook in the subrack and tighten the screws (one-man installation)
- The separate cable routing for analog, digital and power supply lines in cable ducts increases the resistance to mutual interferences between the signals.

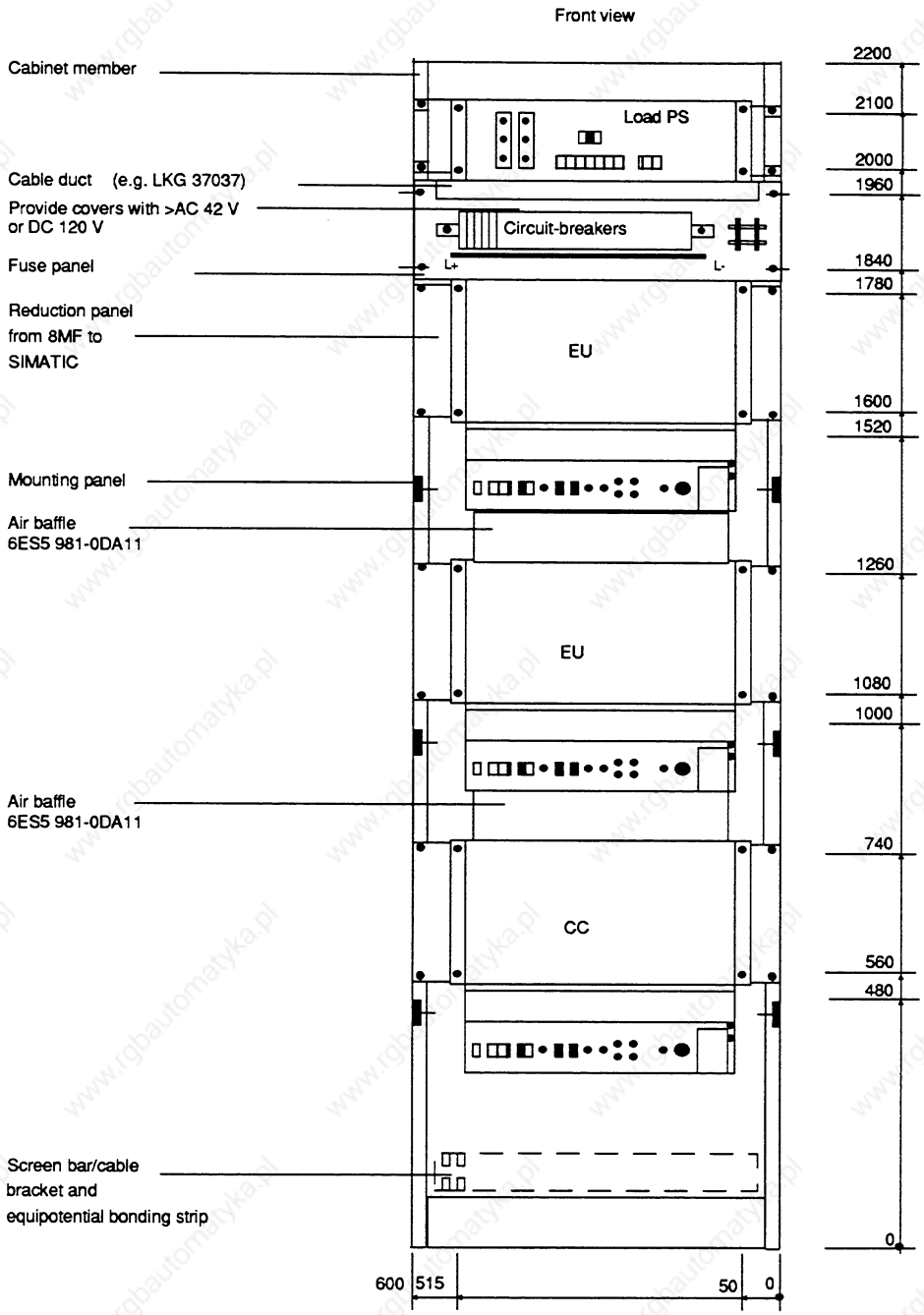


Fig. 4 Front view of 8MF cabinet

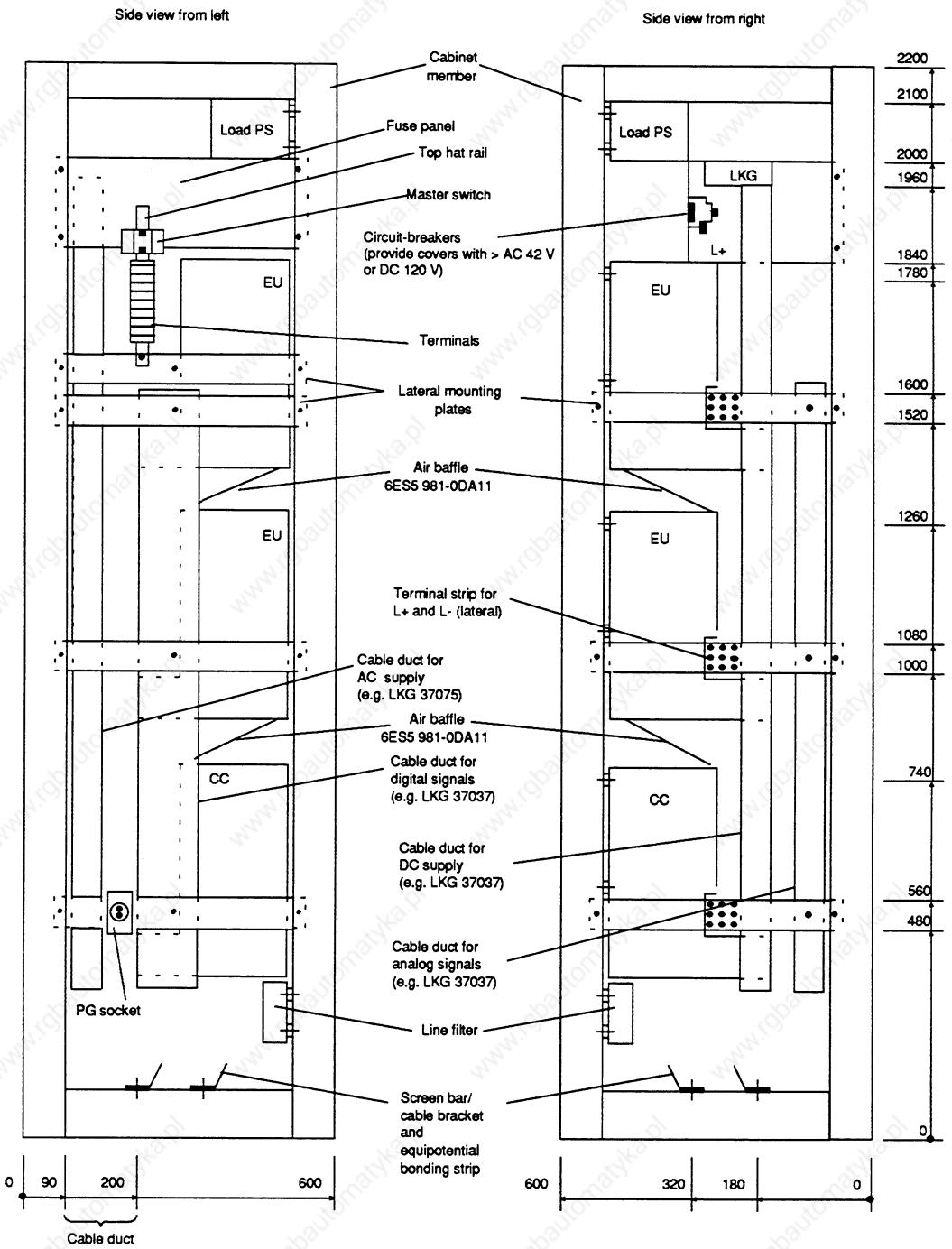


Fig. 5 Side views of 8MF cabinet

4 Design and Connection of Power Supplies

The following section provides information you must observe for the electrical connection of the power supply.

Different power supplies are required for SIMATIC S5 systems:

- Internal power supply for central controllers and expansion units

  The internal power supply of the SIMATIC modules is obtained from power supply units in the form of plug-ins. These power supply units are fixed components of the CC and the EU. You can find their technical data in the Instructions of the respective units and in the catalogs.

- Load power supply for the I/O modules as well as sensors and actuators.

4.1 Internal Power Supply for Central Controllers and Expansion Units

The power supplies fitted in the CCs and EUs deliver the internal DC voltages of 5 V, 15 V and 24 V from the input voltage of 120/230 V AC or 24 V DC.

When equipping the CCs and EUs, ensure that the rated current of the respective power supply is not exceeded. You can find the current consumption of the individual modules with the 5 V supply e.g. in the catalogs and the Instructions of the respective module (Technical Data).

Floating and non-floating power supplies are available for the input voltage of 24 V DC.

The permissible input voltage for power supplies with a rated input voltage of 24 V DC is:

- Static DC 20 to 30 V.

The permissible input voltage is as follows for power supplies with a rated input voltage of 230/120 V AC:

- With rated voltage 230 V: 187 to 253 V AC
- With rated voltage 120 V: 93 to 127 V AC.

4.2 Load Power Supply

The series 6EV 13.. power supply units from Siemens (output currents 20 and 40 A) can be used to supply the I/O modules as well as the CC and EU power supplies with the input voltage of 24 V DC. Detailed information can be found in Catalog ET1.

The following must be observed when dimensioning load power supplies for digital output modules (S5-135U/155U series):

- To protect the cables and lines from overcurrents and to protect the modules from short-circuits, additional fuses are present on the modules in addition to the electronic short-circuit protection (in the power supply). The fuses also serve as protection if the power supply connections are reversed.
- The electronic short-circuit protection for digital outputs only responds when 2-3 times the rated current is exceeded. You should therefore make sure that the load power supply can supply the current required to trigger the short-circuit protection of an output.
- Note when selecting the load power supply, and taking into consideration all connected output loads, that two to three times the rated output current can flow briefly at the output in the event of a short-circuit before the pulsed electronic short-circuit protection takes effect. This excess current is generally present with non-regulated load power supply units.
- In the case of regulated load power supply units, especially with small output currents up to 20 A, the rated output current of the load power supply must be dimensioned such that several times the rated current can flow in the event of a short-circuit.

Caution:

Safe electrical isolation according to CENELEC HD 384.4.41 (IEC 364-4-41) Part 4 (VDE 0100) or VDE 0160 must be guaranteed with all power supply units used for SIMATIC S5 devices and modules. All electrically-isolated Siemens power supplies of the 6EV13... series satisfy this condition.

4.3 Electrical Design with Process Peripherals

The following section shows various designs of power supplies for CCs, EUs and process peripherals.

The following are possible:

- Grounded power supply
- Centrally grounded power supply
- Non-grounded power supply.

You must observe the following fundamental points when designing the electrical configuration of the process peripherals:

- A master switch (to VDE 0113)[1] or a disconnection facility (to VDE 0100)[2] must be provided for the CC, EU and load power supply.
- For DC 24 V load circuits you require a load power supply with guaranteed electrical isolation. Non-regulated load power supplies must be provided with a capacitor (dimensioning: 250 µF per 1 A load current). This means you must connect a capacitor in parallel to the output terminals.
- Electrical isolation by means of a transformer (to VDE 0113[1] Section 6.1.1 and VDE 0100[2]) is recommended for load circuits for supplying external control devices with electromagnetic operating coils (e.g. more than 5).
- The circuits for the sensors and actuators can be used in groups.
- To protect against parasitic voltages, the subracks must be connected together with a large-area contact and low impedance.

1) VDE 0113 is equivalent to EN 60 204, IEC 204-1
2) VDE 0100 is equivalent to CENELEC HD 384.4.31 (IEC 364-4-41).

4.3.1 Power Supply for CCs, EUs and Process Peripherals from Grounded Battery or Grounded Power Supply Units

The ground of the internal supply voltages is connected to the subrack housing. Grounded operation provides the best noise immunity.

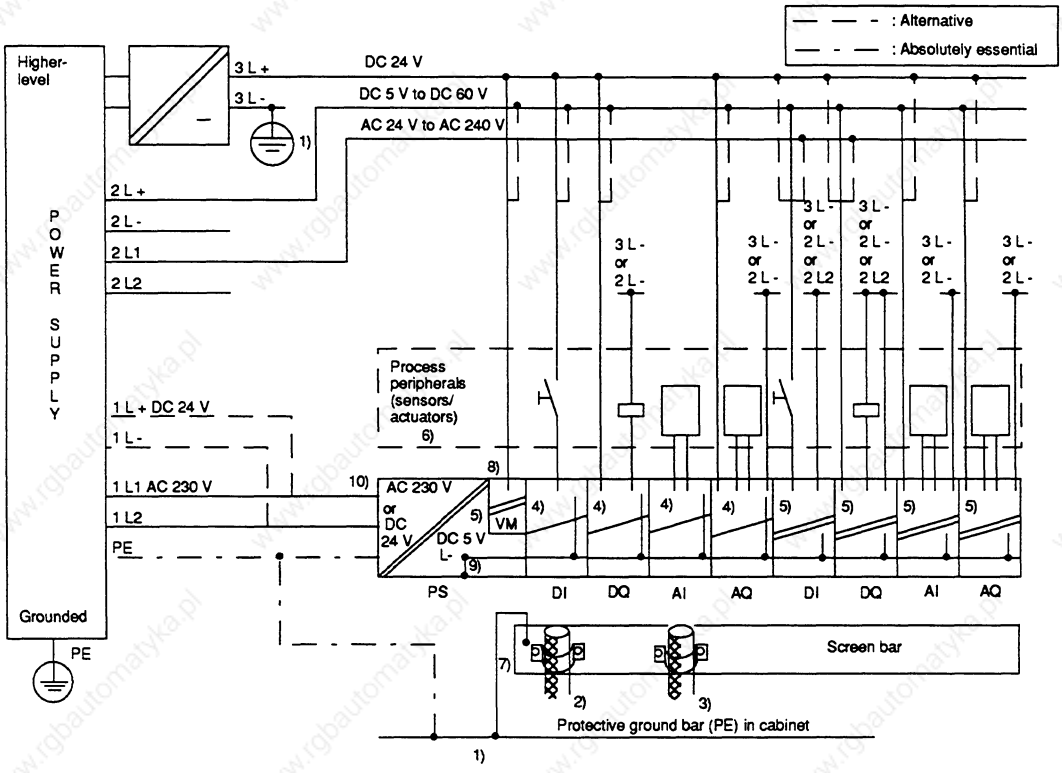


Fig. 6 Possible connections for sensors/actuators of the process peripherals to grounded power supply units

1) Housing potential (cabinet potential) = protective ground conductor.
2) Use cable screen, if available, for digital modules. Provide screening with longer cables; connect at one end to cabinet inlet or connect at both ends.
3) Connect cable screen at one end to cabinet inlet with analog modules or also at both ends; lead on up to module.
4) Non-floating module.
5) Floating module.
6) Protective ground conductor required to housings of sensors and actuators.
7) Connection cable with as large a cross-section as possible (black) > 16 mm^2; if the screen is used as the protective ground conductor (green/yellow), connect at both ends.
8) Only with S5-135U/155U series: monitoring of load voltage L+ (24 V DC).
9) Non-removable connection between the internal ground of the supply voltages and the housing.

Particularly important:

10) Electrical isolation is not available with the power supply unit 24 V/10 A (order no. 6ES5 955-3NA12); operation is only possible without problems on grounded power supply unit.

4.3.2 Power Supply for CCs, EUs and Process Peripherals from Centrally Grounded Battery or Centrally Grounded Power Supply Units

If SIMATIC S5 programmable controllers are to be installed where a central grounding is available, then proceed as shown in Fig. 7. This is, however, not as immune to noise as the grounded system in Fig. 6.

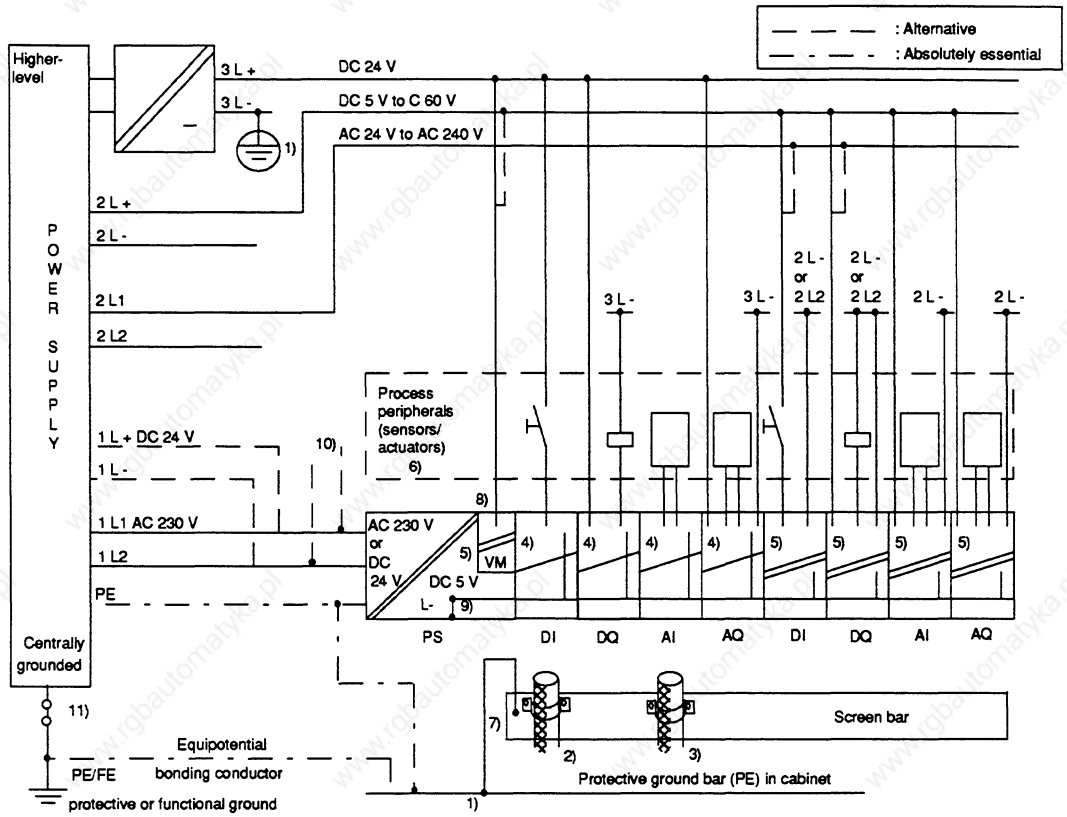


Fig. 7 Possible connections for sensors/actuators of the process peripherals to centrally grounded power supply units

1) Housing potential (cabinet potential) = protective ground conductor.
2) Use cable screen, if available, for digital modules. Provide screening with longer cables; connect at one end to cabinet inlet or connect at both ends.
3) Connect cable screen at one end to cabinet inlet with analog modules or also at both ends; lead on up to module.
4) Non-floating module.
5) Floating module.
6) Protective ground conductor required to housings of sensors and actuators; can be omitted for safely generated functional extra-low voltages.
7) Connection cable with as large a cross-section as possible (black) > 16 mm^2.
8) Only with S5-135U/155U series: monitoring of load voltage L+ (24 V DC).
9) Non-removable connection between the internal ground of the supply voltages and the housing.

Particularly important:

10) Electrical isolation is not available with the power supply unit 24 V/10 A (order no. 6ES5 955-3NA12); operation on a centrally grounded power supply unit is therefore not directly possible. Voltage supply required via 3L+/-.

11) Removable connection for test purposes.

4.3.3 Power Supply for CCs, EUs and Process Peripherals from Non-Grounded Battery or Non-Grounded Power Supply Units

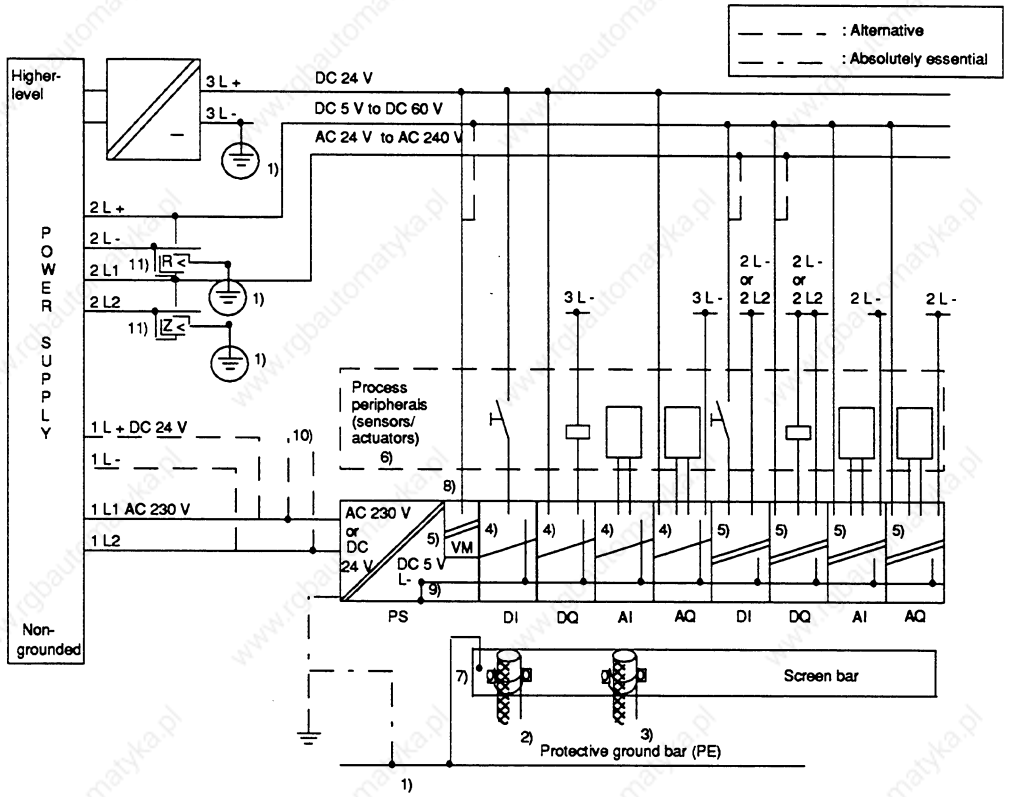


Fig. 8 Possible connections for sensors/actuators of the process peripherals to non-grounded power supply units

1) Housing potential (cabinet potential) = protective ground conductor.
2) Use cable screen, if available, for digital modules. Provide screening with longer cables; connect at one end to cabinet inlet or connect at both ends.
3) Connect cable screen at one end to cabinet inlet with analog modules or also at both ends; lead on up to module.
4) Non-floating module.
5) Floating module.
6) Protective ground conductor required to housings of sensors and actuators; can be omitted for safely generated functional extra-low voltages.
7) Connection cable with as large a cross-section as possible (black); if the screen is used as the protective ground conductor (green/yellow), connect at both ends.
8) Only with S5-135U/155U series: monitoring of load voltage L+ (24 V DC).
9) Non-removable connection between the internal ground of the supply voltages and the housing.

Particularly important:

10) Electrical isolation is not available with the power supply unit 24 V/10 A (order no. 6ES5 955-3NA12); operation on a centrally grounded power supply unit is therefore not directly possible. Voltage supply required via 3L+/-.

11) Insulation monitoring equipment is required if dangerous conditions could result through double faults and/or with voltages > 42 V AC or 120 V DC. Only one insulation monitor is required per supply unit (to VDE 0113 Section 6.2.2).

4.4 Load Power Supply from Two Power Supply Units

The design of the load power supply using two power supply units enables you to specifically disconnect parts of the process peripherals. The inputs and outputs of different modules can be assigned as a group to one power supply unit.

The supply to inputs and outputs of different modules from two power supply units is indicated below using two examples.

4.4.1 Non-floating Modules

In the case of non-floating input/output modules it must be ensured that the negative poles (L-) of the power supply units are connected to the reference potential (SIMATIC device/cabinet housing). This is necessary since the inputs are referred to ground.

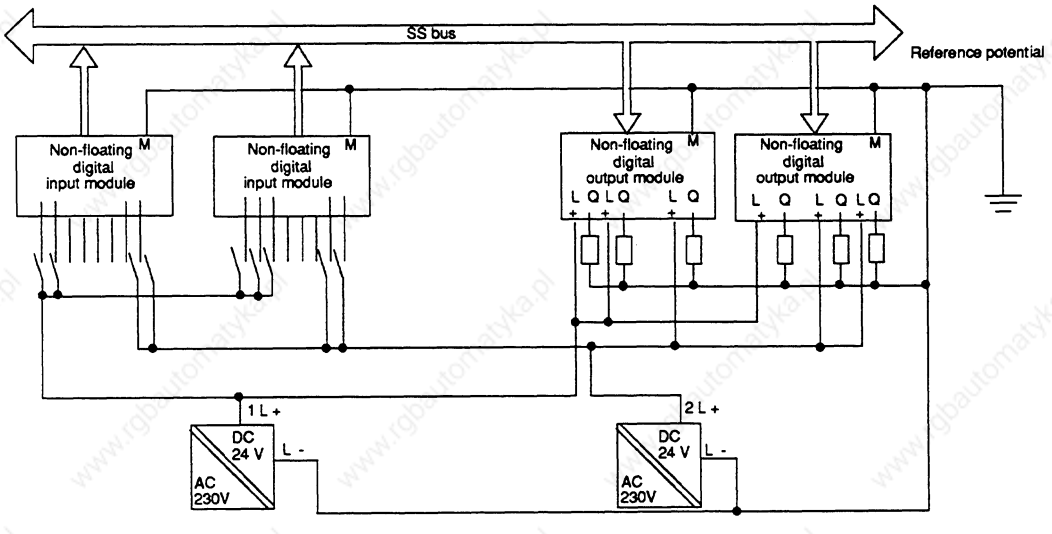


Fig. 9 Example of supply of non-floating peripheral modules from two power supply units

4.4.2 Floating Modules

In the case of floating modules, the inputs or outputs can be supplied from two power supply units by dividing into isolated groups.

Note that electrical isolation between the groups is lost as a result of the connection of inputs or outputs of two floating groups to one power supply unit.

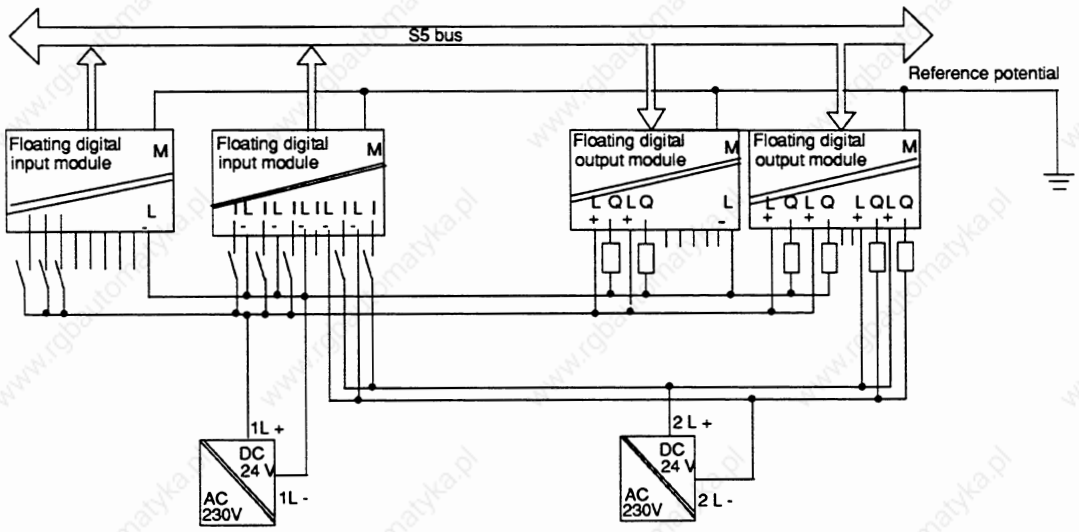


Fig. 10 Supply of floating peripheral modules from two power supply units

When using CCs and EUs with a mains connection, we recommend a Siemens power supply unit from the 6EV13.. series with electrical isolation as the power supply for the process peripherals (load voltage).

5 Wiring Layout

You can achieve a high noise immunity for your programmable controller by using a correct wiring layout. The measures required are described in the following sections.

5.1 Wiring Layout Inside a Cabinet

To ensure a correct layout of wiring inside cabinets, the wiring must be divided into the following groups:

Group A: screened data lines (for PG, OP, SINEC L1, CP 525 etc.)
screened analog lines
screened signal lines for DC and AC voltages ≤ 400 V
non-screened lines for DC and AC voltages ≤ 60 V

Group B: non-screened lines for DC and AC voltages > 60 V and ≤ 400 V

Group C: non-screened lines for DC and AC voltages > 400 V and ≤ 1 kV

Route all wiring groups separately in the cabinet. Separately means that the wiring is routed in

- separate cable ducts
- separate wiring bundles with approx. 10 cm clearance.

When laying screened lines (e.g. analog lines) make a large-area contact of the screen to a cable clamping rail at the inlet to the cabinet and connect the screen further to the final point without an interruption (see Section 6.4).

5.2 Wiring Layout Outside Cabinets

- Route the cables outside cabinets and within buildings on metal cable trays. Make a conductive connection between the ends of two adjacent cable trays and connect these to ground at distances of 20 to 30 m.

  The following may be routed on the same cable trays (cable routes, gutters, channels):

  - cables from group A and
  - cables from group B with approx. 10 cm clearance.

  Route cables in group C on separate cable trays (cable routes, conduit).

- Always screen analog lines.

- Non-screened cables (e.g. signal lines, power supply lines) must be routed with as large a clearance from sources of interference (contactor, transformer, motor, electric welding unit) as possible.

- Signal lines and associated equipotential bonding lines should be routed with the smallest possible distance from one another and on the shortest path.

- Lines between the programmable controller and sensors/load should be installed whenever possible without breaks. If a break in the line is unavoidable, screen the terminal block e.g. with a metal box making large-area contact to a screen bar.

- Route associated single lines (e.g. forward and return lines, power supply cables) as close as possible to one another. If possible these lines should be twisted.

Note:

Signal lines and power cables up to 1 kV must be routed separately but can be routed in parallel. A minimum clearance of 10 cm must be observed. The clearance should be increased proportionally with higher voltages, and the safety regulations must be observed (e.g. IEC 664/664A).

5.3 Wiring Layout Outside Buildings

- If you route cables outside buildings, a double-screened cable must always be used for analog and data signal transmissions.
  The following must be observed when routing double-screened cables:
  - connect the outer screen to ground at both ends
  - only connect the inner screen at one end to the receiver side.

- Ensure that the equipotential bonding is sufficient. Connect an equipotential bonding conductor if necessary.

- The lightning protection and grounding regulations must be observed.

5.4 Equipotential Bonding

Different potentials can occur between different parts of your plant (e.g. different power supplies). These differences can be reduced by laying equipotential bonding lines to ensure the correct functioning of electronic components.

Keep the following points in mind when laying an equipotential bonding line:

- The effectiveness of equipotential bonding is directly related to the impedance of the line (less impedance - greater effectiveness). This means that the connection required for equipotential bonding must have not only a low ohmic resistance but also as small an inductance as possible (achieved by keeping line lengths short).

- If screened signal lines with the screens grounded at both ends are required between parts of the plant, the impedance of the additional equipotential bonding line must not exceed a maximum of 10% of the screen impedance.

- The cross-sectional area of the equipotential bonding line must be selected for the max. equalizing currents.

- The equipotential bonding line must be laid so that loops (e.g. between equipotential bonding line and signal lines) cover as small an area as possible.

- The equipotential bonding line must make large-area contact with ground or chassis (see Section 6.4)

6 Cabinet Wiring and Design with Respect to EMC

EMC: electromagnetic compatibility (EMC) is understood to be the ability of an electric device to function without faults in a defined electromagnetic environment without influencing other devices in the environment.

Measures to guarantee EMC must already be made when designing and wiring the individual components in cabinets. The interfering environment must not be ignored if fault-free functioning of the programmable controller and wiring is to be obtained.

The measures required to guarantee EMC, as well as an example of a cabinet design as concerns EMC, are described in the following sections. The check list at the end of this section serves as an aid for checking the EMC-compatible design of your cabinet.

The following section as well as Section 5.1, Wiring Layout Inside a Cabinet, must be observed when designing your cabinet to guarantee EMC. These sections handle the subjects:

- Grounding of all inactive metal components
- Wiring layout in the cabinet
- Shielding of devices and cables
- Use of special interference-suppression measures.

6.1 Grounding of Inactive Metal Components

An important factor which contributes towards interference-free operation is consistent grounding. Grounding is understood to be the electrical connection of all inactive metal components (VDE 0160). Large-area grounding must always be used.

Large-area grounding means:

- Ground all conducting parts.
  These include subracks, cabinet members, cabinet panels, cabinet doors, screen bars, filter housings.

Measures to be observed when grounding:

- Make all ground connections with a low impedance.
- Connect all metal parts with a large-area contact.
- Use ground straps for the connection. Metallic wire mesh made of tin-plated copper strands is suitable as the ground strap. It should be kept as short as possible. The surface area of the ground straps is decisive, and not the cross-section, because of the high-frequency noise pulses discharged.
- Make the screw connections using NOMEL contact washers[1)].

NOMEL contact washers
Assemble contact washers such that the teeth bite into the part to be screwed and thus generate a metallic contact.

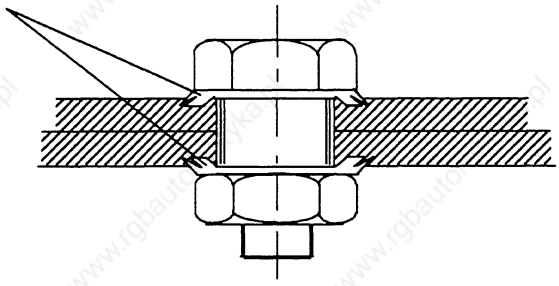


Fig. 11 NOMEL contact washers

1) Contact washer Siemens standard 70093 available from
- Siemens ANL A443 Werkzeug 8520 Erlangen
- Teckentrup GmbH und Co. KG, Postfach 120, D-5974 Herscheid 2,
- NOMEL S.A. Tour Franklin, Cedex 11, F-92081 Paris.
- or from your local Siemens representative

6.2 Shielding of Devices and Cables

Shielding is a way of attenuating (dampening) magnetic, electric or electromagnetic interferences. Shielding can be divided into:

Device shielding

Cabinets and housings must be incorporated into the measures for shielding the programmable controllers. The following must be observed:

- Cabinet enclosures such as side panels, rear walls, roof and floor panels must be connected sufficiently often with a low impedance in the case of an overlapping arrangement (connection interval e.g. 50 mm).
- Doors must additionally be connected to the cabinet ground. Use at least 2 ground straps.
- If sources of strong interference are present in the cabinet (transformers, cables to motors etc.), these must be isolated from sensitive electronics areas by metal partitions (steel, highly permeable material, e.g. mu-metal). The panels must be screwed several times to the cabinet ground with a low impedance.

The central grounding point must be connected to the protective ground conductor (grounding bar) with a low impedance and a Cu conductor $\geq 16\ mm^2$ as short as possible.

Cable screening

Screened cables must be connected at both ends to the grounding bar with a large-area contact and if possible directly at the cabinet inlet. Good attenuation of all conducted frequencies can only be achieved by connecting at both ends.

The following must be observed when handling the screen:

- Use metal cable clamps to secure the braided screens with a large-area contact.
- Avoid the use of cables with foil screens since the foil can be easily damaged by tension or pressure when fitting, thus leading to a poorer screening effect.

Note:

An equalizing current may flow via the screen connected at both ends in the case of variations in the ground potential. Use an additional equipotential bonding conductor in this case (see Section 5.4 Equipotential Bonding).

In certain cases the screen can also be connected at only one end. Only the lower frequencies are then attenuated. Connection of the screen at one end may be more favorable if:

- An equipotential bonding conductor cannot be laid
- Analog signals (several mV or µA) are transmitted.

Interferences on cable screens are discharged to ground via the grounding bar and the equipotential bonding conductor. A low-impedance path to ground for the interfering currents must be provided so that these discharged currents do not produce a source of interference themselves:

- Tightly connect the screws of cable plugs, modules and equipotential bonding conductors.
- Protect the contact surfaces of equipotential bonding conductors and ground lines from corrosion.

6.3 Use of Special Noise Suppression Measures

Connection of inductors

Provide suppression (e.g. using RC elements, varistors or free-wheeling diodes) for inductors installed in the same cabinet (e.g. contactor and relay coils) not activated by SIMATIC S5 modules.

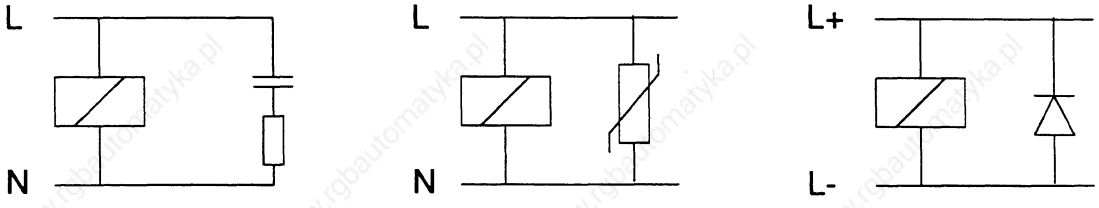


Fig. 12 Wiring inductors (example)

If further contacts are connected in series to SIMATIC outputs, the SIMATIC internal fusing is not effective. In such cases the inductor must be fused directly.

Protection against electrostatic discharge

Use metal housings or cabinets that are closed in at all sides to protect devices and modules against electrostatic discharge. Connect these housings or cabinets to the grounding point where you set them up so as to form a good contact.

Caution:

If you must work on the system with the cabinet open, follow the guidelines to protect electrostatically sensitive devices and modules (ESD).

The interference resistance is always reduced when the cabinet is open.

Mains power connection for programmers

Provide a grounded socket in each cabinet to supply power for a programmer. The sockets should be connected to the distribution board to which the protective ground conductor of the cabinet is also connected.

Cabinet illumination

Do not use fluorescent lamps for the cabinet illumination since these generate interferences. If you must use fluorescent lamps, take the precautions shown in Fig. 13 LINESTRA® lamps are more suitable.

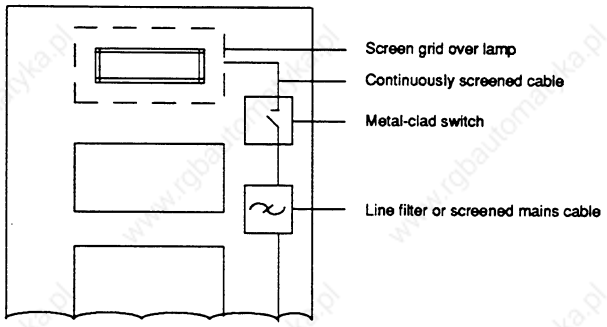


Fig. 13 Measures to suppress noise from fluorescent lamps in a cabinet

6.4 Example of an EMC-compatible Cabinet Design

The example of a cabinet design shown in Fig. 14 - taking into consideration EMC - shows the grounding of all inactive metal components and the connection of screened cables. This example only applies to grounded operation. Observe the points listed in Fig. 14 during installation.

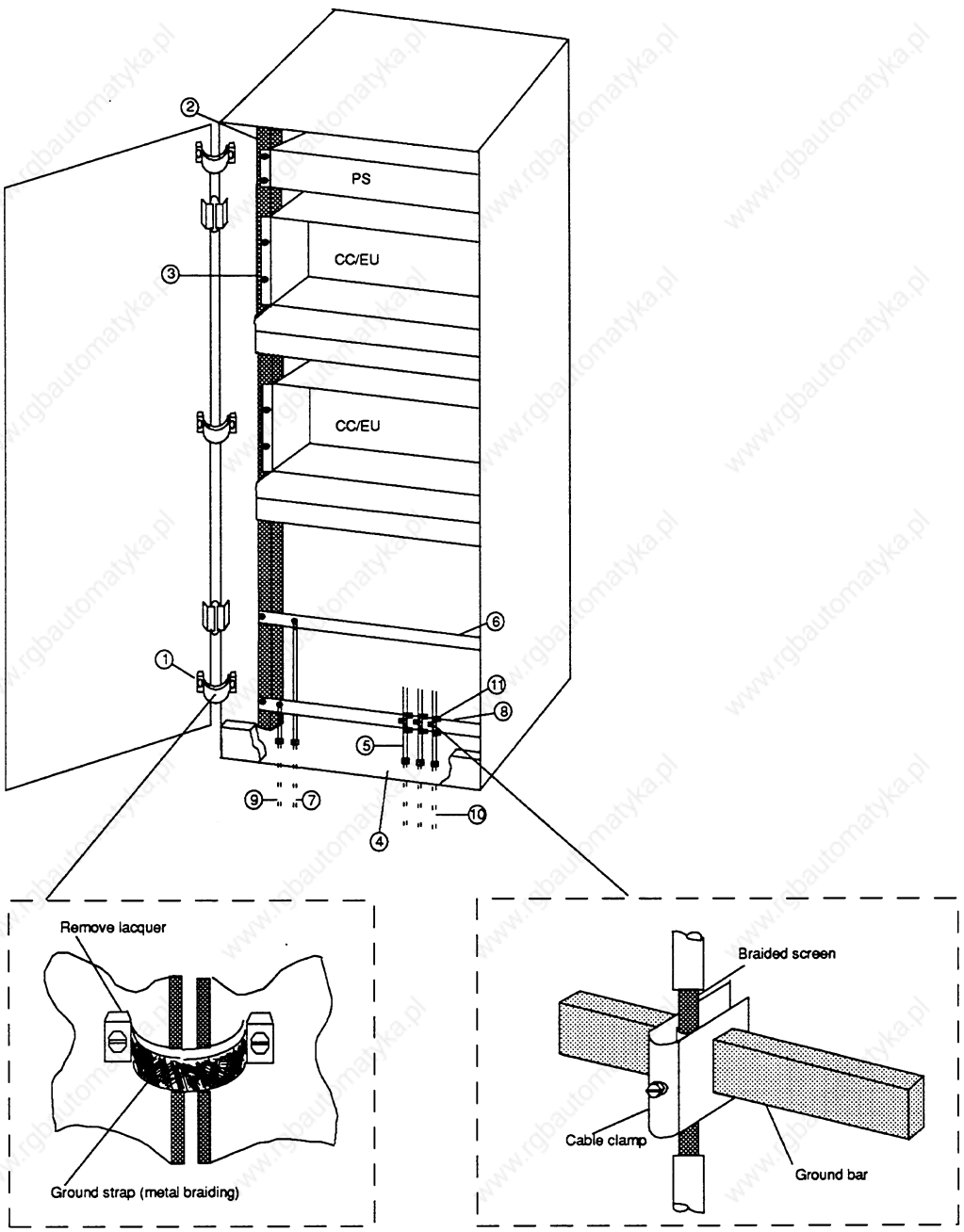


Fig. 14 Example of an EMC-compatible cabinet design

Re 1. **Ground straps**
All inactive metal components (e.g. cabinet doors and supporting panels) must be connected using ground straps if large-area metal-metal connections are not present. Metallic wire mesh made of tin-plated copper strands is suitable as the ground strap. It should be kept as short as possible, with a ratio between the length and width of less than 3 to 1.

Re 2. **Cabinet members**
The cabinet members must be connected to the cabinet housing with a large-area contact (metal-metal connection).

Re 3. **Mounting bracket**
A large-area metal-metal connection must be made between the cabinet member and mounting bracket.

Re 4. **Base panel**
A large-area metal-metal connection to the cabinet housing must be guaranteed.

Re 5. **Cable screwed glands**
Unused cable screwed glands must be closed using blanking plates in the case of closed cabinets with heat exchangers.

Re 6. **Equipotential bonding bar**
The bar must be connected to the cabinet members with a large-area contact (metal-metal connection).

Re 7. **Equipotential bonding conductor**
The conductors must be connected to the equipotential bonding bar.

Re 8. **Ground bar**
This serves as the central grounding point of the cabinet and must be connected to the cabinet members with a large-area contact (metal-metal connection). The ground bars must be connected to the external central grounding point to guarantee discharging of interfering and fault currents. It can additionally be used to connect screened cables.

Re 9. **Cable from central grounding point**
The cable must be connected to the grounding bar with a large-area contact.

Re 10. **Signal cables**
The screen of screened signal cables must be connected to the grounding bar with a large-area contact using cable clamps or to an additional screen bar connected with a large-area contact, and then routed further to the end point (e.g. I/O module) without interruption.

Re 11. **Cable clamp**
The cable clamp must enclose the braided screen over a large area.

6.5 Checklist for EMC-compatible Cabinet Design

EMC measures	Remarks
Connection of inactive components (Section 6.1)	
Are all inactive metal components connected together with a large-area contact and low impedance, and grounded?	
Is there a sufficient connection to the central grounding point?	
Screw connections made using NOMEL contact washers?	
Particularly check the connections to: • Subracks • Cabinet members • Cabinet bar • Filter housing	
Equipotential bonding (Section 5)	
With a spacially separated design, check the routing of the equipotential bonding conductor	
Device screening (Section 6.2)	
All cabinet components provided with contacts at sufficient intervals?	
Doors connected to the cabinet body using ground straps?	
Are only metallic device plugs used?	
Cable screening (Section 6.2)	
Are all analog cables screened? Are screens connected at both ends?	
Are cable screens connected to ground bar or screen bar at cabinet inlet?	
Are cable screens connected via cable clamps with a large-area contact, completely enclosed and with a low impedance?	
Inductors (Section 6.3)	
Are isolating panels used in event of magnetic influences from inductors?	
Are all coils of contactors connected to RC elements?	

Table 2 Checklist for EMC-compatible cabinet design

EMC measures	Remarks
Cable routing (Section 5)	
Cabling divided into groups?	
Power supply cables (230 V) and signal cables routed in separate ducts or bundles?	
Complete cabling introduced into cabinet at one position?	
Signal cables routed close to grounded surface?	
Forward and return lines of individually routed conductors twisted if possible?	

Table 2 Checklist for EMC-compatible cabinet design (continued)

7 Framework and Wall Mounting of SIMATIC S5 Controllers

If you operate your SIMATIC controllers in a environment which is free from interferences to the greatest possible extent, you can fit the central controllers and expansion units on a framework or directly on a wall.

The following must be observed:

- A reference surface made of sheet-steel should be provided to improve the deviation of interfering currents conducted via the inlet cables. This reference surface must be at least 480 mm x 250 mm large and connected to the central grounding point. If you use screening or cable clamping rails, space must be provided for these on the reference surface. In the case of framework mounting, the metal frame serves as the reference surface.

- Fit the screen bar or cable clamping bar to this reference surface or to the framework. Ensure that the connection between the rails and the reference surface or framework is made with a large-area contact and low impedance (metal-metal connection).

- Connect all inactive metal components together with a large-area contact and low impedance. Inactive metal components are: subrack, power supply, reference surface, screen bar, protective ground bar.

- Also observe the points for the wiring layout (see Section 5).

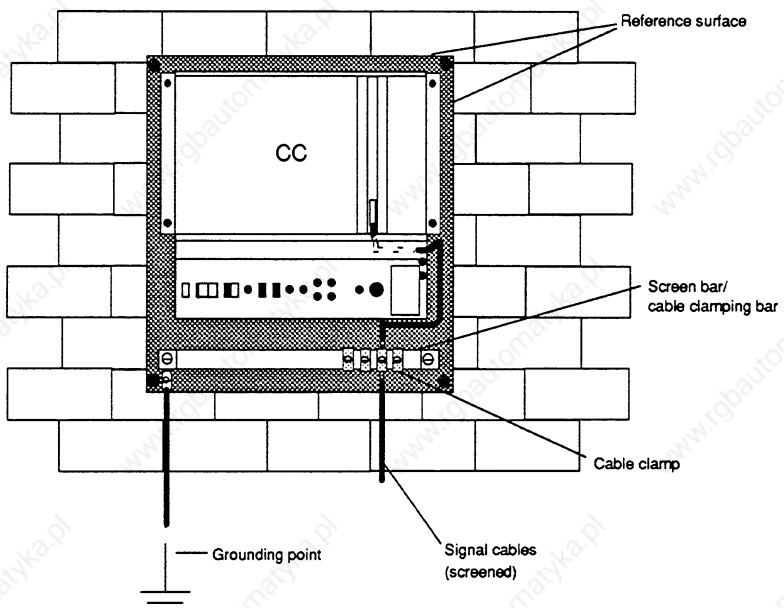


Fig. 15 Wall mounting

Note:

If you mount subracks from the S5-135U/155U series on a wall, provide a minimum clearance from the rear panel of ≥ 30 mm. This is necessary to guarantee heat exhange on the heat sink of the power supply.

Note:

When using radio telephones the field strength must not exceed 3 V/m at the programmable controller.

Owing to unknown values such as output power and frequency range, radio telephones should only be used when a certain safety clearance from PLCs not installed in cabinets is maintained.

8 Lightning Protection Measures

If cables and lines for SIMATIC S5 devices are laid outdoors, the lightning protection regulations must be adhered to.

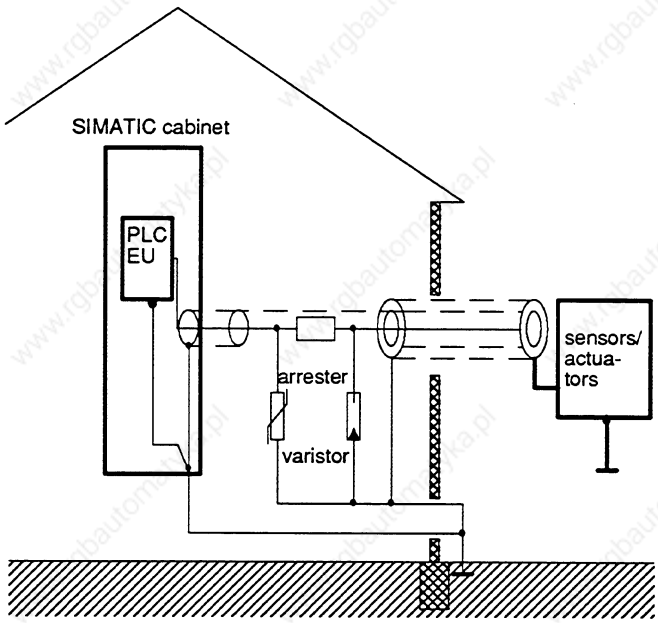


Fig. 16 Arrangement of lightning protection elements

Protect signal lines from overvoltages as follows:

- varistors

  or

- inert gas-filled lightning arresters

Install these protective elements

- as close as possible to the point of entry into the building
- before cables enter the cabinet

Note:

Lightning protection must be adapted to each situation individually. Please contact your local Siemens representative if you require advice.

9 Safety Measures

When configuring systems that have programmable controllers - as is the case with contactor equipment - follow the relevant regulations: CENELEC HD 384.4.41 (IEC 364-4-41) "Electrical installations of buildings" and also EN 60204 (European standard, corresponds to IEC 204-1), "Electrical equipment of industrial machines" (VDE 0113).

Pay special attention to the following points:

- Prevent conditions that could endanger or injure people or which could damage machines and material.
- When power is restored after a power failure or after EMERGENCY STOP units are released, machines must not be able to restart automatically.
- When a programmable controller malfunctions, commands from EMERGENCY STOP units and from safety limit switches must remain effective under all conditions. These safety measures must have a direct effect on the actuators in the power circuit independent of the programmable controller.
- When EMERGENCY STOP units are activated, safety must be guaranteed for people and systems as follows:

Note:

As operator of the system, you are responsible for the measures described in the section "Safety Measures" to avoid dangerous situations.

9.1 Protection against Indirect Contact

Parts which can be touched must not carry dangerous currents even in the event of a fault. They must be incorporated into the protective measures against dangerous currents.

This requirement is satisfied if all metal parts which can be touched and which could be dangerous in the event of a fault are safely connected electrically to the protective ground conductor (PE). The max. permissible resistance between the protective ground conductor and the part to be protected is 0.5 Ω.

SIEMENS

SIMATIC S5

S5-135U Central Controller

6ES5 135-3UA11
6ES5 135-3UA21
6ES5 135-3UA31
6ES5 135-3UA41
6ES5 135-3UA51

Instructions

C79000-B8576-C395-01

IMPORTANT!

Note the order no. of your central controller.

There are different possibilities for configuration

between the versions

-3KA . . and -3UA . . !

Preface

This manual provides the hardware description and installation and maintenance procedures for the central controller S5-135U (6ES5 135-3UA11/-3UA21/-3UA31/-3UA41/-3UA51). This manual also provides everything you need to know about operation, maintenance and technical specifications.

This manual is intended for engineers, programmers, and maintenance personnel.

If you have any questions about the S5-135U central controller not answered in this manual, please contact your local Siemens representative.

NOTE

These instructions do not cover all details or variations in equipment or provide for every circumstance that can arise with installation, operation, or maintenance. If you want further information or if particular problems arise that are not covered sufficiently for your purposes, contact your local Siemens sales office.

The contents of this instruction manual shall not become part of or modify any prior or existing agreement, commitment or relationship. The sales contract contains the entire obligation of Siemens. The warranty contained in the contract between the parties is the sole warranty of Siemens. Any statements contained herein do not create new warranties or modify the existing warranty.

How to Use This Book

This section discusses information that may be helpful as you use this book.

The main information you will find in :

- Section 1.2.1, Possible Configurations
- Chapter 2, Installation/Dimensions of the S5-135U Central Controller
- Section 4.5.3, Checklist for Starting Up

The individual chapters offer you the following:

- **Chapter 1: Technical Description**
  This chapter discusses the application of the S5-135U programmable controller and describes its central controller. It includes details on possible configurations with expansion units.

- **Chapter 2: Installation of the S5-135U Central Controller**
  This chapter describes the installation procedure for the central controller, including its integrated power supply unit and 15 V submodule.

- **Chapter 3: Wiring Connections on the Power Supply Unit of the S5-135U Central Controller**
  This chapter describes all connections and explains how to set the installed fan and battery monitoring. It includes recommendations on wiring for fan and temperature monitoring on the central contoller.

- **Chapter 4: Operation of the S5-135U Central Controller**
  This chapter discusses the commissioning and the operation requirements. It explains the LEDs and operating elements on the power supply, jumper locations, and the functions of the alarm relays in the power supply.

- **Chapter 5: Maintenance of the S5-135U Central Controller**
  This chapter explains how to change the modules and the back-up battery. It also provides information on the connector pin assignments of the bus PCB and the pin designations of the interrupt signals.

- **Chapter 6: Technical Data of the S5-135U Central Controller**
  This chapter lists the technical data for the central controller, including its integrated power supply unit and 15 V submodule. It informs you about safety, climatic and mechanical ambient conditions and noise immunity.

- **Index**
  The index lists the key words in alphabetical order in these instructions together with their corresponding page numbers.

Training

Contact your local Siemens representative for information on training courses to aid you in becoming familiar with this product.

Reference Materials

The following books that support the S5-135U are recommended:

- Catalog ST 54.1: S5-135U, S5-155U and S5-155H Programmable Controllers
  (Order No. E86010-K4654-A111-A6-7600)*

Programmer Manuals:

- S5-135U (CPU 928B)
  (Order No. 6ES5 998-2UL22)*
- S5-135U (CPU 928)
  (Order No. 6ES5 998-1UL23)*
- S5-135U (S and R Processor)
  (Order No. 6ES5 998-0UL22)*
- U-Periphery
  (Order No. 6ES5 998-0PC22)*
- PG 685 Programmer
  (Order No. 6ES5 885-0SC21)*
- PG 710 Programmer
  (Order No. 6ES5 814-0SC21)*
- PG 730 Programmer
  (Order No. 6ES5 834-0FC21)*
- PG 750 Programmer
  (Order No. 6ES5 886-0FC21)*
- PG 770 Programmer
  (Order No. 6ES5 887-0FC21)*
- STEP 5 Programming Package for Personal Computers
  (Order No. 6ES5 896-0SC21)*
- You will find an introduction to programming with STEP 5, as well as an explanation of how to work with the S5-135U programmable controller and its I/O modules in the following book:
- Automating with SIMATIC S5-135U
  by Hans Berger
  Siemens AG ISBN 3-8009-1561-8

* available from your local Siemens representative.

Page

1	**Technical Description of the S5-135U Central Controller**	**4**
1.1	**Application**	**4**
1.2	**Design**	**5**
1.2.1	Possible Configurations of the S5-135U Central Controller	7
1.2.2	Addressing the I/O Modules	8
1.3	**Device Configuration S5-135U**	**9**
1.4	**Interfacing Between Central Controllers and Expansion Units**	**9**
1.4.1	Central Interfacing	10
1.4.2	Distributed Interfacing	11
1.4.3	Examples of Interfacing Possibilities	13
1.5	**Mode of Operation**	**14**
1.5.1	Single/Multiprocessor Operation	14
1.5.2	Unit Interfacing	15
2	**Installation of the S5-135U Central Controller**	**17**
2.1	**Installing the Central Controller**	**17**
2.2	**Installing the Power Supply Unit**	**19**
2.2.1	Installing the 15 V Supplementary Module	19
2.3	**Installation of the Modules**	**20**
2.3.1	Connections to the CPUs, CPs and Interface Modules	20
2.3.2	Connections to the I/O Modules	20
3	**Wiring Connections on the Power Supply Unit of the S5-135U Central Controller**	**21**
3.1	**Connections of the Power Supply Units**	**21**
3.2	**Recommended Wiring for Fans and Temperature Monitoring**	**22**
3.2.1	Setting the Fan Monitoring	23
3.2.2	Setting the Battery Monitoring	24
4	**Operation of the S5-135U Central Controller**	**25**
4.1	**General Notes on the Power Supply Unit**	**25**

4.2 Operating and Display Elements of the Power Supply Unit 27

4.3 Functions and Locations of Jumpers on the Power Supply Units 28

4.4 Power Supply Behavior in Event of Faults 31

4.5 Start-Up and Functional Test 32

4.5.1 Restart with Single Processor Operation 33
4.5.2 Restart with Multiprocessor Operation 34
4.5.3 Checklist for Starting Up 35

5 Maintenance of the S5-135U Central Controller 37

5.1 Removing and Inserting S5 Modules 37

5.2 Removing and Inserting Power Supply Units 38

5.3 Replacing the Back-up Battery and the Fans 39

5.4 Pin Assignments of the Power Supply Unit 40

5.5 Pin Assignments of the Bus PCB 42

5.6 Pin Designations of the Interrupt Signals on the Bus PCB 45

6 Technical Data of the S5-135U Central Controller 47

6.1 Power Supply Unit 6ES5 955-3LC14 49

6.2 Power Supply Unit 6ES5 955-3LF12 51

6.3 Power Supply Unit 6ES5 955-3NA12 53

6.4 Power Supply Unit 6ES5 955-3NC13 55

7 Index 57

These instructions apply to the S5-135U programmable controller with the following power supply units:

Order number of the central controller with power supply unit	Power supply unit	Technical data
6ES5 135-3UA11	6ES5 955-3LC14	230V/5V/18A IN: 230V/120V AC OUT: 5V/18A DC 24V/0.8A DC 15V DC . . . [1]
6ES5 135-3UA21	6ES5 955-3LF12	230V/5V/40A IN: 230V/120V AC OUT: 5V/40A DC 24V/2.8A DC 15V DC . . . [1]
6ES5 135-3UA31	6ES5 955-3NC13	24V/5V/18A IN: 24V DC OUT: 5V/18A DC 24V/0.8A DC 15V DC . . . [1]
6ES5 135-3UA41	6ES5 955-3NA12	24V/5V/10A IN: 24V DC OUT: 5V/10A DC 24V/0.8A DC 15V DC . . . [1]
6ES5 135-3UA51	6ES5 955-3NF11	24V/5V/40A IN: 24V DC OUT: 5V/40A DC 24V/2.8A DC 15V DC . . . [1]

IN = INPUT (primary)
OUT = OUTPUT (secondary)

1) A 15 V module can be inserted into all power supply units. This is necessary if you use a CP 535 or CP 143 communications processor. The total current of the 24 V DC and 15 V DC supplies must not exceed the maximum current of 0.8 A or 2.8 A.

Table 1 S5-135U and power supply units

1 Technical Description of the S5-135U Central Controller

1.1 Application

The SIMATIC S5-135U programmable controller is a versatile multiprocessor unit for automation applications in the medium and top performance ranges.

The standardized instrument technology, the modular design of the units and the expansion facilities mean that the S5-135U programmable controller can be readily adapted to the respective automation task. It can be configured according to your requirements. The system provides you with various expansion facilities (e.g. EG-185U expansion unit), communications facilities (e.g. SINEC H1) and a range of operation, monitoring and programming devices of varying performance.

With the S5-135U programmable controller you can solve the following automation tasks simply and economically:

- Open-loop control
- Closed-loop control and computing
- Communication
- Operation and monitoring.

The controller is thus suitable for:

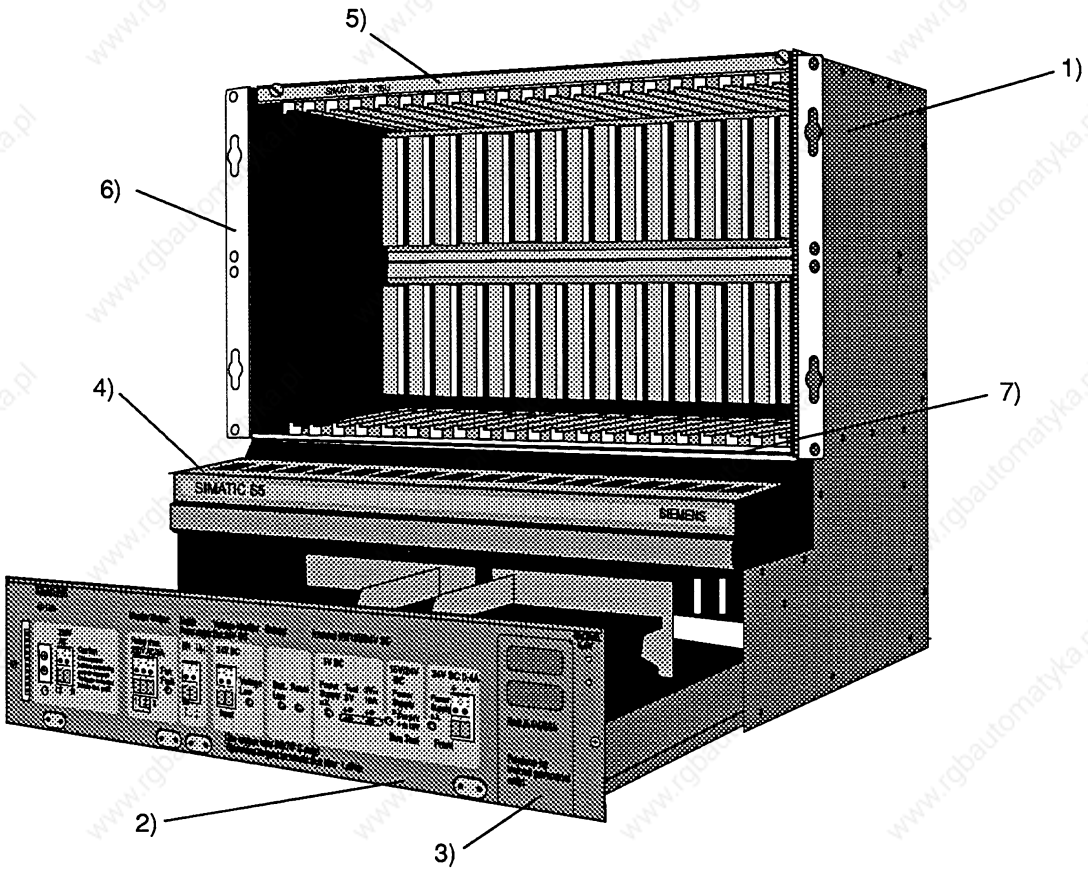
- Machine controls
- Process automation
- Process monitoring.

The programming language is STEP 5 with the following methods of representation:

- Control system flow chart - CSF
- Ladder diagram - LAD
- Statement list - STL
- Higher-level sequence diagram - GRAPH 5.

The CPU 920 (M processor) can be programmed in ASSEMBLER or in one of the programming languages C or BASIC.

1.2 Design



1) Housing with 21 module slots
2) Power supply unit with fans
3) Plug-in for back-up battery
4) Cable duct
5) Locking rail
6) Mounting bracket
7) Rail for individual locking

Fig. 1 S5-135U central controller

The S5-135 U central controller consists of:

- Housing with 21 module slots
- Power supply unit with fans and a plug-in for a back-up battery.

Housing

The housing consists of screwed sheet-steel sections with ventilation openings at the top and bottom, as well as aluminium parts. The sheet-steel sections are chromium-plated, the aluminium locking rails are tin-plated. The housing contains the bus PCB which serves to connect the modules electrically. All the slots have guide rails to ensure correct connection of the modules. At the top of the housing there is a locking bar to lock all modules at once. Modules with individual locking mechanisms can be secured using the bottom rail. A cable duct for incoming and outgoing signal cables is located at the front of the housing.

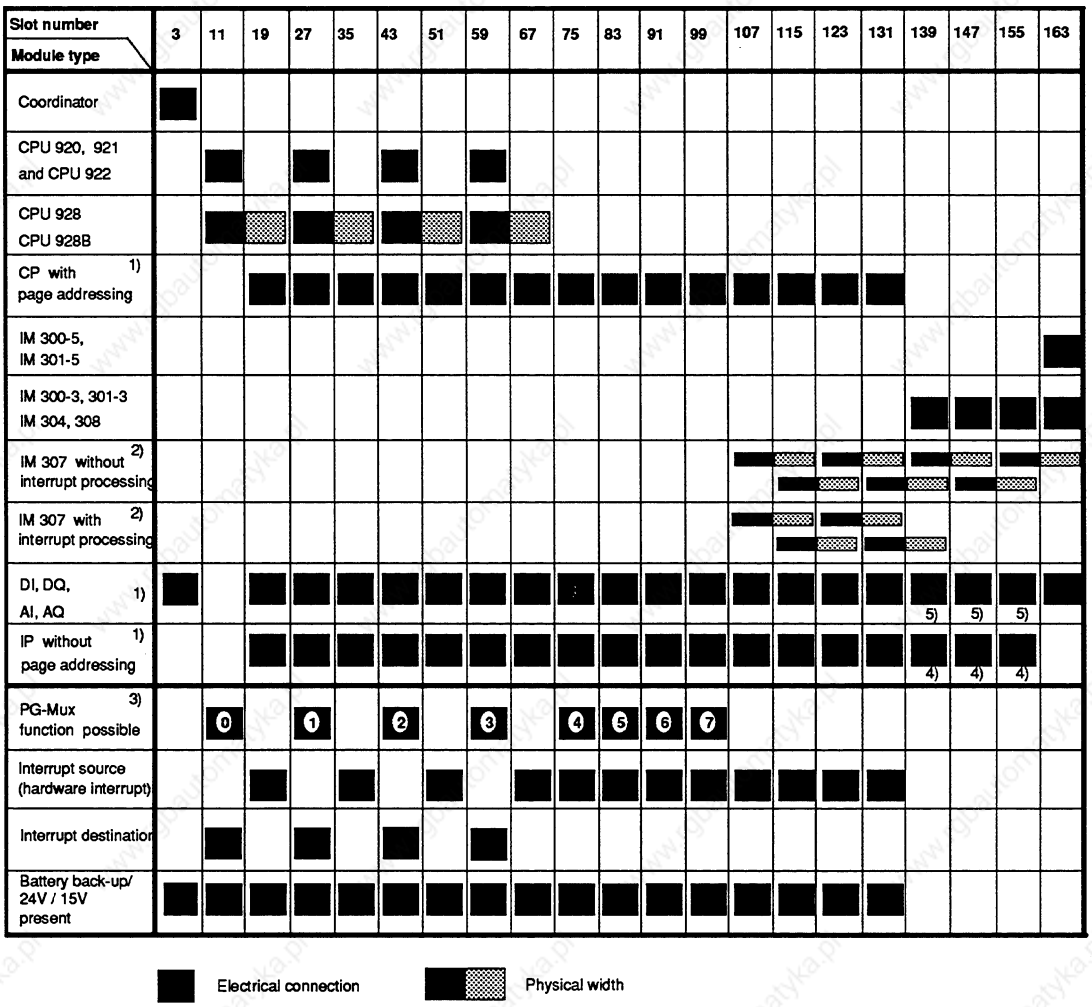
Power supply unit

The power supply unit with its fans is accommodated in a tier at the bottom in the housing. The input voltage is either 24 V DC or 230/120 V AC depending on the type of power pack used. An internal selector is present for adaptation with 230/120 V AC.

To operate the CP 535 and the CP 143, the Ethernet bus interfaces for SINEC H1, a submodule supplying 15 V to the CP slots via the bus board must be installed in the power supply unit.

1.2.1 Possible Configurations of the S5-135U Central Controller

The following table shows which slots modules can be plugged into.



1) Note the functions listed in the lower part of the table
2) Note the jumper setting on the IM 307.
3) The numbers in the table indicate the sub-addresses for PG communication via the PG multiplexer.
4) Not suitable for IP 240, IP 241, IP 241USW, IP 242, IP 242A, IP 243, IP 244.
5) Not suitable for DI 432-4U

Table 2 Possible configurations of the S5-135U central controller

Caution:

Do not insert modules in slots for which they were not intended, otherwise the modules can be irreparably damaged.

1.2.2 Addressing the I/O Modules

I/O modules can be addressed in the P and O (extended) I/O areas.

- I/O modules to be addressed in the O area must be plugged into an expansion unit. Depending on which combination of interface modules you require to communicate, you must set the appropriate address area on the interface module of the expansion unit 300, 301, 307 or on the interface module of the central controller 314, 318.

It is also possible to multiply the O area.

- By setting "O area pages" on the IM 308 interface module a multiplexer function can be inplemented which muliplies the O area by 256. The "O page number" must first be entered in O byte 255 before the operations L/T OY or L/T OW can be used to access the periphery. The "O page numbers" are set on the EPROM of the IM 308.

  If you use this method, the O area, described above, is not available.

Caution:

To avoid double addressing, remember the following point:

When you use an input module in the extended address area (O area) in an expansion unit, there must be no input module in the central controller with the same I/O address. The same applies to output modules.

1.3 Device Configuration S5-135U

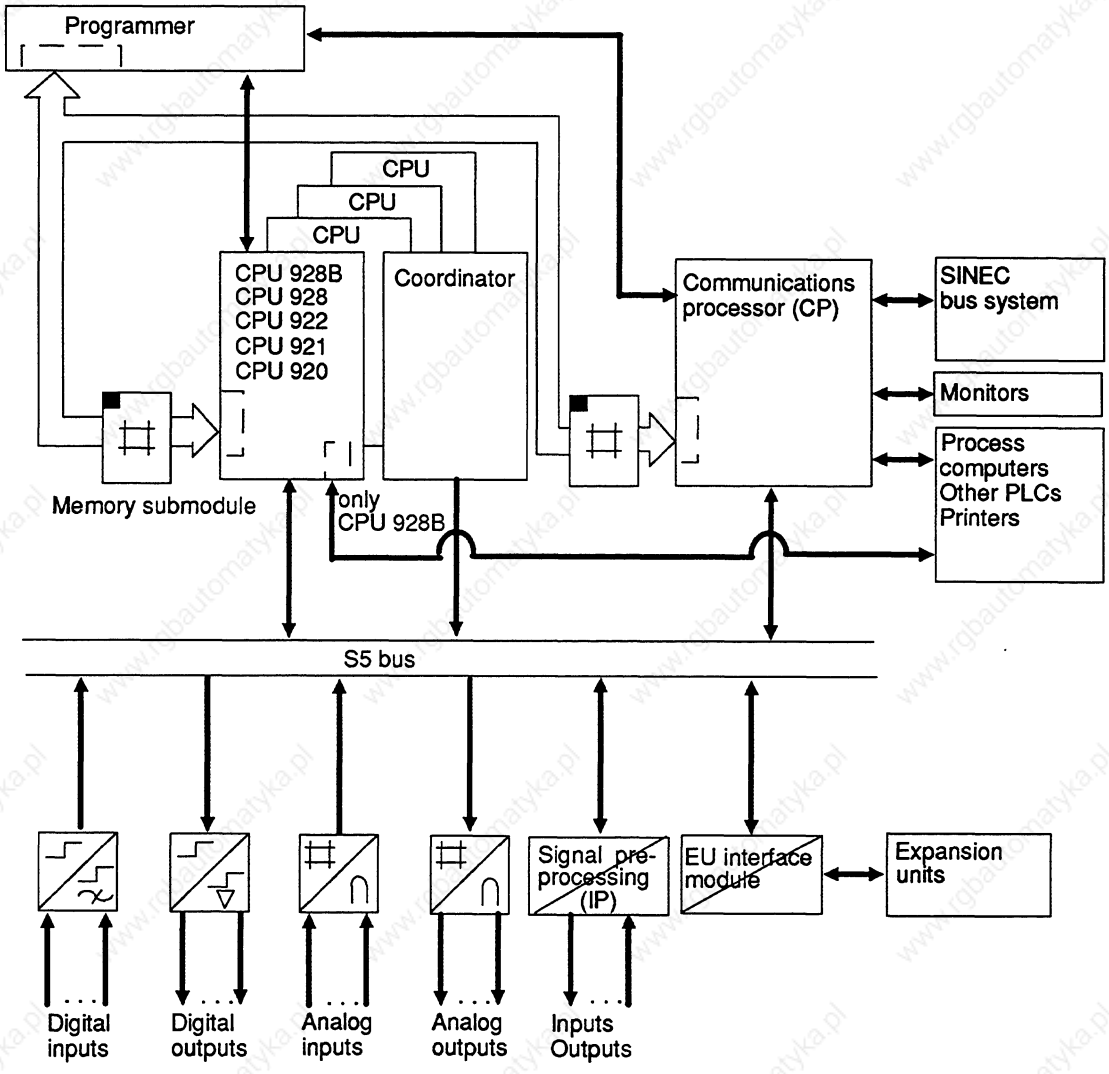


Fig. 2 Device configuration of the S5-135U

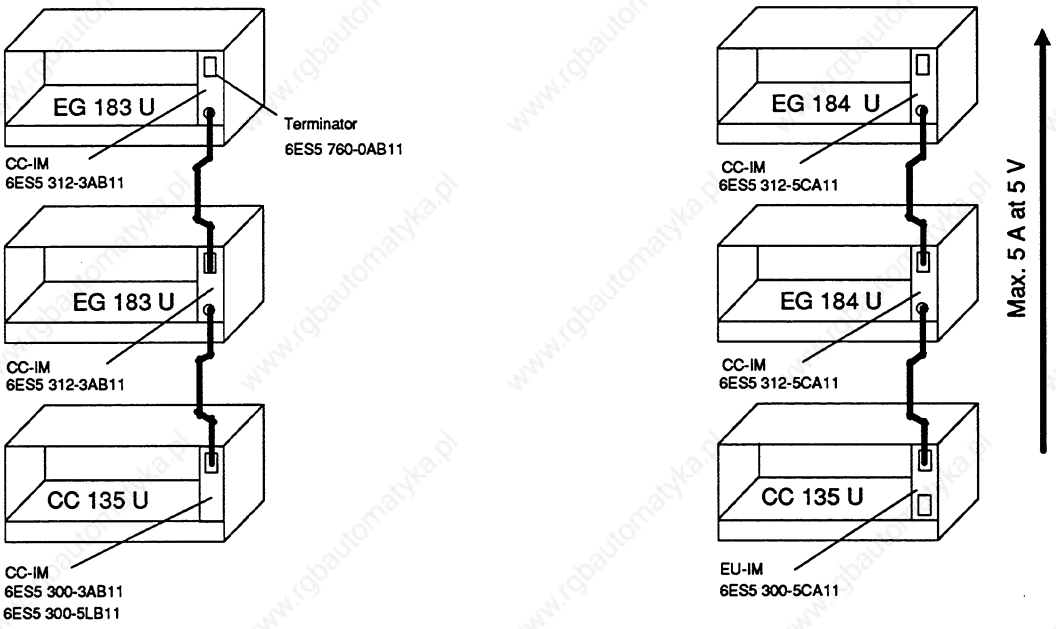
1.4 Interfacing Between Central Controllers and Expansion Units

Expansion units can be connected if the number of slots in the central controller is insufficient. Please refer to the Instructions of the expansion units.

1.4.1 Central Interfacing

Central interfacing means that the expansion units are accommodated together with the central controller in the same cabinet or in an adjacent cabinet. The total cable length from the central controller to the furthest expansion unit must not exceed 2 m.

The example (Fig. 3) shows how the S5-135U CC and the EG-183U/184U expansion unit are connected with the appropriate interface modules.



Expansion unit (EU) with power supply

Expansion unit (EU) without power supply

Fig.3 Example of central design with S5-135U central controllers and EG-183U/184U expansion units

Please note:

- A terminator must be used on the last CC-IM 312-3....
- If you use expansion units without a power supply, the maximum load on the interface cable is 5 A (at 5 V).

For further designs with other interface modules, see Catalogs ST 54.1/ST 52.3 and the manual "U-Periphery" 6ES5998-0PC22.

1.4.2 Distributed Interfacing

Distributed interfacing means that the expansion units are accommodated in a cabinet located further away from the central controller. The total cable length from the CC to the most remote EU must not exceed defined values. These distances depend on the interface module used (see Section 1.4.3, table: Examples of further interfacing possibilities). A distributed configuration up to 600 m is shown in the example (Fig. 4).

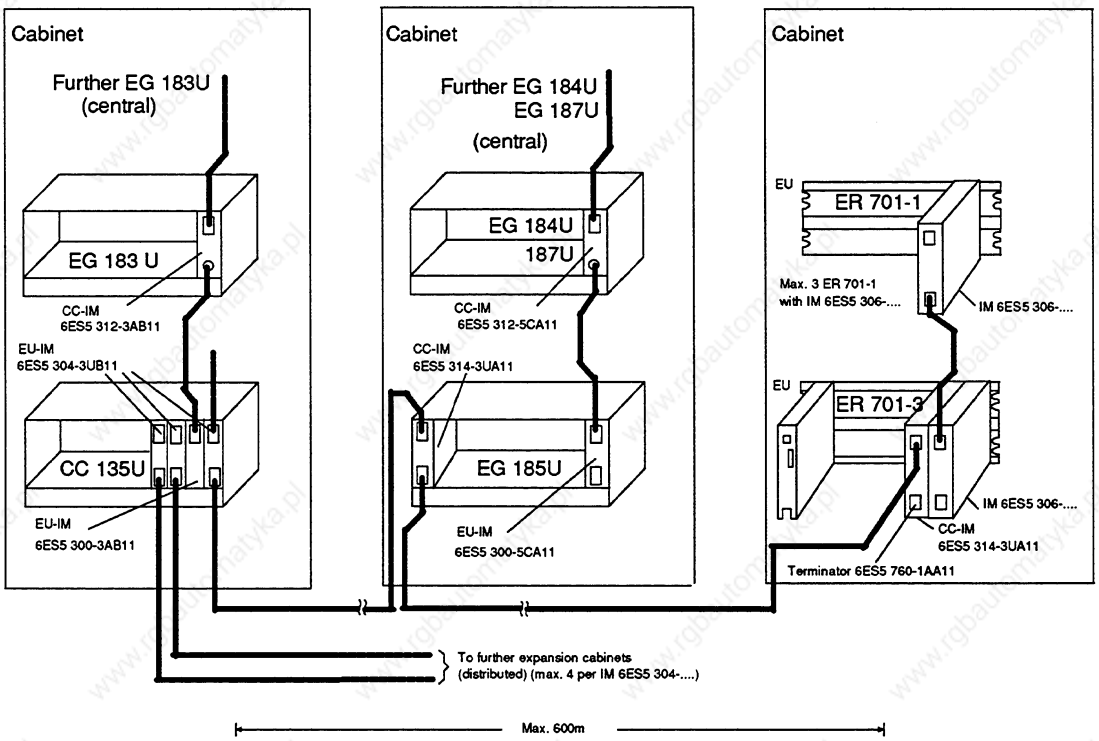


Fig. 4 Example of a distributed configuration up to max. 600 m with S5-135U central controller and EG-183U/184U/187U expansion units, ER 701-1/701-3 subracks

Please note:

- A distributed connection of up to 2 lines with 4 subracks (EG-183U, EG-185U, EG-186U expansion units, ER 701-2 and ER 701-3 subracks) can be made to a CC using an EU-IM 304 via the CC-IM 314.
- The total length (cable connector 721) from the CC up to the last EU can be up to 600 m per line.
- Further subracks can be connected centrally to the EG-185U expansion units and ER 701-3 subracks connected with a distributed configuration.
- Terminators must be inserted in the last CC-IM 314 of each line and in the last central connection of the CC-IM 312-3.

For further designs with other interface modules, see Catalogs ST 54.1/ST 52.3 and the manual "U-Periphery" 6ES5998-0PC22.

Caution:

Only original cable connectors must be used to connect the interface modules to one another.
The screen of these cables is connected at both ends (do not isolate!).
The screen connection to the CC/EU subrack must be guaranteed via the springs on the metal front panel of the interface module or - in the case of plastic front panels - via the springs in the guide rails of the subrack.
Ensure that these important contact springs are not bent or dirty and are not interrupted at any point.

1.4.3 Examples of Interfacing Possibilities

The following table shows which interface modules and cable connectors can be used to connect the various expansion units to the central controller.

Type of design	Interface module in central controller	Expansion unit	Interface module in expansion unit	Cable connector
Central	6ES5 300-3AB11 6ES5 301-3AB13	EG 183U	6ES5 312-3AB11 6ES5 312-3AB31	Fixed on module 312 "
	6ES5 300-5CA11 6ES5 301-5CA12	EG 184U EG 187U	6ES5 312-5CA11 6ES5 312-5CA21	" "
	6ES5 300-5LB11	ER 701-1	6ES5 306-7LA11	6ES5 705-0xxxx
Distributed up to 200 m	6ES5 301-3AB13	ER 701-2	6ES5 310-3AB11	6ES5 721-0xxxx
	6ES5 301-5CA12 6ES5 301-3AB13	EG 183U		
Distributed up to 600 m	6ES5 304-3UB11	ER 701-2 ER 701-3 EG 183U EG 185U EG 186U	6ES5 314-3UA11	6ES5 721-0xxxx
Distributed up to 3000 m	6ES5 308-3UA12	ER 701-2 ER 701-3 EG 183U EG 185U EG 186U	6ES5 318-3UA11	Screened, twisted 2-wire cable
		ET 100U (Catalog ST 52.1)	6ES5 318-8MA12	
		ICM 560	-	
Distributed up to 1500 m	6ES5 307-3UA11	ER 701-2 ER 701-3 EG 183U EG 185U EG 186U	6ES5 317-3UA11	6ES5 722-2xxxx (fiber-optic cable)

Table 3 Examples of interfacing possibilities

Note:

An equipotential bonding conductor ≥ 10 mm^2 must be provided for a distributed connection of central controllers and expansion units via the interface module 301/310 (up to 200 m) or 304/314 (up to 600 m) if a potential difference of more than 7 V can occur (see IEC 364-4-41, VDE 0100). The interface modules 301/310 and 304/314 are non-floating.

Connection via fibre-optic interface module IM 307/317:
The fibre-optic interface module provides an isolated connection that can also transfer interrupts from an expansion unit (EG 186U or ER 701-3) to a central controller.

1.5 Mode of Operation

The S5-135U programmable controller belongs to the SIMATIC S5 range. The controller can be used as a single processor as well as for multiprocessing with up to four CPUs.

1.5.1 Single/Multiprocessor Operation

The user program is executed cyclically. Access to the input and output modules is possible at all times via the S5 bus. If the 923A/923C coordinator is present, you can configure the S5-135U programmable controller with more than one CPU.
The S5-135U programmable controller is a multiprocessing device with processors for specific tasks which can be combined in a variety of ways:

- **CPU 928B:** Designed for multiple tasks; provides very fast binary signal processing (open-loop control tasks) as well as very fast word processing (computing and closed-loop control)
- **CPU 928:** Designed for multiple tasks; provides fast binary signal processing (open-loop control tasks) as well as fast word processing (computing and closed-loop control)
- **CPU 922 (R processor):** Mainly for fast word processing (computing and closed-loop control). Binary signal processing is also possible
- **CPU 920 (M processor):** For processing of measured values, arithmetic and statistics. Programming is carried out in BASIC, C or Assembler
- **CPU 921 (S processor):** For binary signal processing (open-loop control tasks).

The automation task can be clearly organized by using several CPUs. Each CPU executes its program independent of the others. This increases the overall processing speed. Each processor can be started independent of the others. Up to four processors can be operated in the programmable controller, and the user program can be divided amongst several CPUs for specific tasks. All data to be exchanged between CPUs is transferred via the S5 bus.

In multiprocessor operation, you must specify address lists in data block DB1 of each CPU to allow the input/output modules to be assigned to the individual CPUs. DB1 must not be programmed as a standard DB. In single processor operation, DB1 can be used to increase the processing speed.

The coordinator assigns each processor access to the S5 bus cyclically. Information is exchanged between the processors via the coordinator, which has a memory for this purpose. The coordinators 923A and 923C have an interprocessor communication flag area. The 923C also has a memory for multiprocessor communication and a central programmer connection. Via this, up to 8 modules (CPU, CP, IP) can be accessed without reconnecting the cable to the PG or SINEC-CP.

1.5.2 Unit Interfacing

The S5-135U central controller can be connected to:

- **Programmers or operator panels (OP)**

  The programmers can be directly connected to the processors or the 923C coordinator for programming or system start-up.
  PG multiplexer function: with an electronic switch on the 923C coordinator you can access up to 8 modules in the central controller from the PG. OPs can only be operated with a direct connection to the CPU.

- **Standard peripheral devices and computers**

  The communications processors (CP) can handle data traffic independently with

  - standard peripheral devices such as printers, keyboards, monitors,
  - computers or
  - other programmable controllers.

  The data required for texts and displays can be programmed for each communications processor in a RAM or EPROM submodule.

  The CPU 928B can handle data traffic independently via its second serial interface with

  - standard peripheral devices such as printers, keyboards,
  - computers or
  - other programmable controllers.

- **SINEC buses**

  The communications processors (CP) can handle data traffic independently with

  - PGs, computers or
  - other programmable controllers.

  You can also use CPs for operation and monitoring, to display and/or modify process data. CPs are also available to display diagnostic messages and there are also CPs with mass memories.

2 Installation of the S5-135U Central Controller

2.1 Installing the Central Controller

The S5-135U central controller is designed for installation in cabinets, on frameworks and on walls (see Part 2, Section 3.3; Specifications When Installing a Cabinet and Chapter 7; Framework and Wall Mounting of SIMATIC S5 Controllers).

The S5-135U central controller need only be accessible from the front for carrying out connections and maintenance.

The following Figs. show you the dimensions relevant to installation of the central controller.

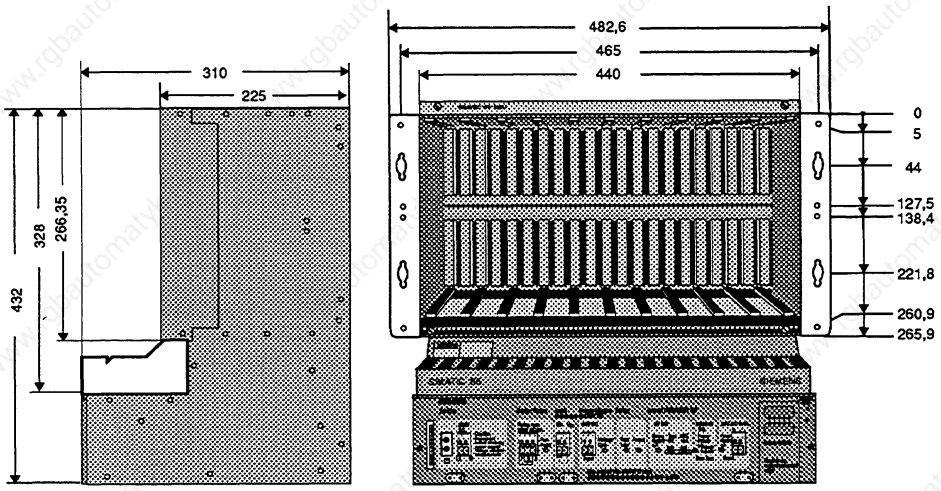


Fig. 5 Device dimensions of S5-135U central controller (in mm)

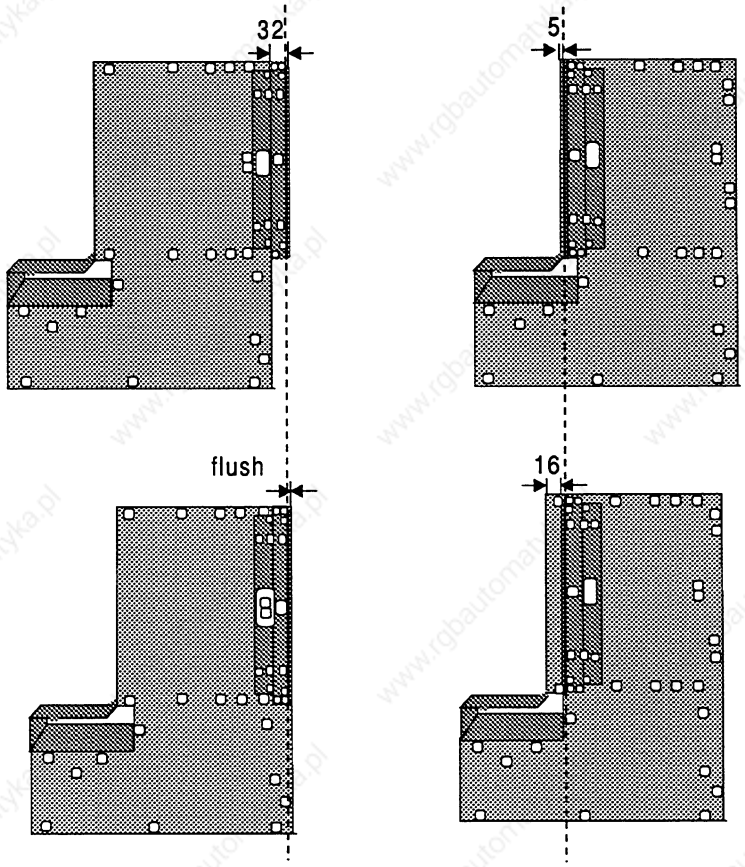


Fig. 6 Different ways of fixing mounting brackets

Note:

The same installation dimensions apply to the EG-183U/184U/185U and 186U expansion units as to the central controller.

2.2 Installing the Power Supply Unit

Push the power supply unit to the back. Push it firmly against the limit stop until the the front panel is flush with the device frame. The spring action of the contacts must be overcome. Then tighten both screws in the frame at the right and left of the front panel. The protective jumper at the left must be sucurely connected to the front panel terminal and the rack of the central controller.

2.2.1 Installing the 15 V Supplementary Module

The supplementary module [1] must only be inserted when no voltage is applied.
Remove the power supply unit as described in Sections 2.2 and 5.2, and insert the 15 V supplementary module in the space shown in Fig. 7.

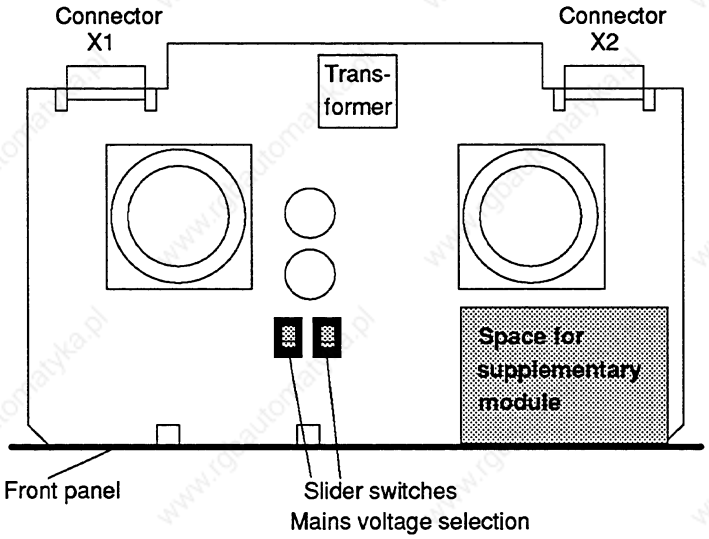


Fig. 7 Mounting location of 15V supplementary module

[1] See Appendix for Order No.

2.3 Installation of the Modules

The dimensions of the S5-135U programmable controller modules correspond to the double-height Europa format (w x h x d: 20.32 mm x 233.4 mm x 160 mm).

There are modules with different widths:

Slots occupied	Standard slots [1]	Front panel width	Module
1	$1^1/_3$	20.32 mm	e.g. CPU 922
2	$2^2/_3$	40.64 mm	e.g. CPU 928 CPU 928B
4	$5^1/_3$	81.28 mm	e.g. CP 580

1) Standard slot in ES 902 = 15.24 mm.

Table 4 Examples of widths of modules

Observe the following when installing the modules:

- Insert each module into the subrack as far as possible
- Lock the modules with individual locking mechanisms
- Screw the top locking rail for modules onto the central controller.

To improve the ventilation and the degree of protection, you can cover vacant slots using dummy front panels. See "Ordering Information" in the Appendix.

2.3.1 Connections to the CPUs, CPs and Interface Modules

Connect the cables from CPUs, communications processors and expansion unit interface modules using front plugs.

1. Metal front plugs with a sliding locking mechanism are locked by pushing the sliding bracket down.
2. Metal front plugs with a thumb wheel screw are screwed to the device.

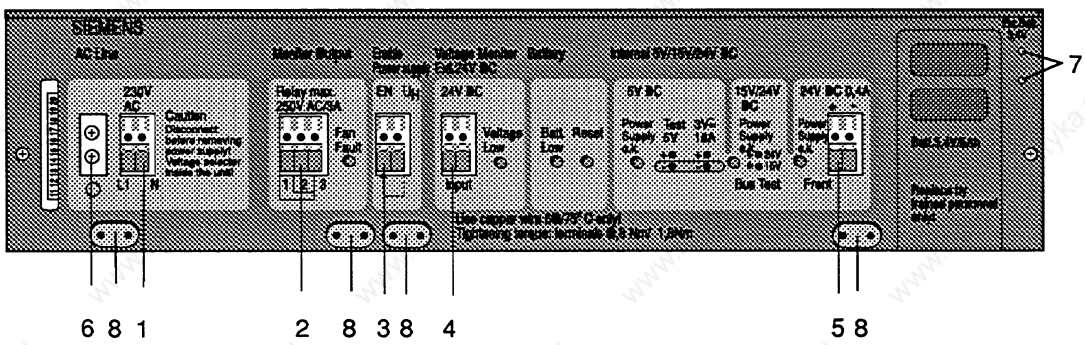
Take care to assign the plugs to the correct modules as damage could otherwise result.

2.3.2 Connections to the I/O Modules

Information on how to connect the cables to the I/O modules can be found in the U-Periphery Manual (Order No. 6ES5998-0PC22) and in the ST catalogs.

3 Wiring Connections on the Power Supply Unit of the S5-135U Central Controller

3.1 Connections of the Power Supply Units



1 AC line:
230 V/120 V selectable input voltage (a 24 V DC line is also possible depending on the type of power supply unit).

2 Monitor output:
external signalling on LED and relay contact if one or both fans stops; this results in the output voltages being switched off (function selectable using jumper F-R of the power supply unit; in this case only relay signal and LED display).
In addition, failure of the back-up battery can be signalled on the power supply unit with the jumper RR-LL closed.
- See Chapter 4.3 for the functions and locations of the jumpers on the power supply units.
- See Chapter 3.2 for recommended wiring.

3 Enable power supply:
the power supply unit is switched off if no voltage is present at the EN input. Not more than 7 EN inputs (front terminal) can be set with an U_H output.
- See Chapter 3.2 for recommended wiring.

4 Voltage monitor:
24 V load voltage monitor input, must be connected or switched inactive by means of jumper BA-EX in the power supply unit. This does not apply to the power supply unit 6ES5 955-3NA12.

5 24 V DC, 0.4 A output:
this output can be used to supply the enable inputs of the U periphery.

6 Protective ground conductor connection:
connection between power supply unit and housing.

7 Connection sockets for external back-up voltage of 3.4 to 3.6 V

8 Cable detensioners for connection cables with metal contact surface for cable screens.

Fig. 8 Connections of the power supply unit

Caution:

The appropriate safety regulations must be adhered to, especially IEC 364-4-41. The terminals at the front are suitable for cables with a cross-section up to 4 mm^2. The 230/120 V AC power connection must have a core cross-section of at least 0.75 mm^2. Ensure that the tension on the cables is relieved sufficiently.

3.2 Recommended Wiring for Fans and Temperature Monitoring

If several devices with fan subassemblies are to be monitored together, connect the terminals EN and U_H of the power supply unit according to the following diagram. **All devices are switched off if the fan fails in one device.** The jumper settings required on the power supply units for this purpose are described in the following sections.

If you have installed the device in a cabinet with heat exchanger, the maximum internal temperature could be exceeded even with the fans working.
It is advisable to install a temperature monitoring device or thermostat in the top of the cabinet (in the exhaust air flow) (see Catalog NV 21). If a fault develops, the signal from the fan or temperature monitoring should be transferred immediately to the CPU to activate a programmed response. The actual shut-down should be triggered by a (customer-installed) time-delay relay after a maximum of 60 seconds, by interrupting the U_H-EN loop.

Up to 7 EN inputs can be controlled using one U_H output.

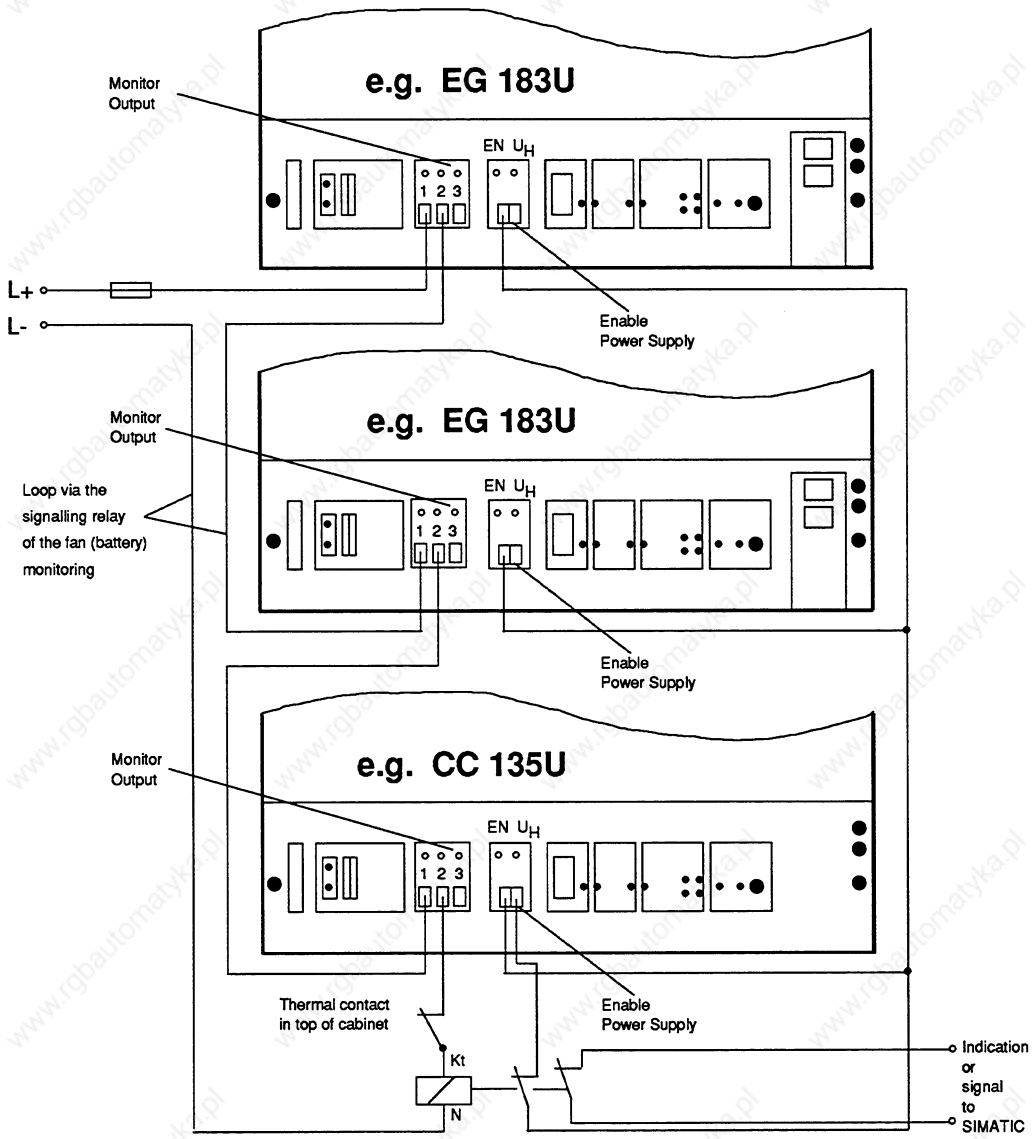


Fig. 9 Recommended connections for fan/temperature monitoring when using S5-135U CC and EG-183U EUs in one cabinet

Caution:

Modules with a Winchester disk drive can only be used at an ambient temperature of up to 50 °C. The permissible switching point of the thermal contact in the top of the cabinet must be adapted. If necessary, use an adjustable thermostat (see Catalog NV21).

3.2.1 Setting the Fan Monitoring

You can use the jumper F-R on the power supply units to select whether the internal supply voltages U_A (5 V) are to be switched off or not in the event of a fan failure:

- Jumper F-R closed: U_A switched off (signal via contact)
- Jumper F-R open: U_A not switched off (signal via contact).

The signalling relay ("Monitor Output") is activated if one or both fans stops. The LED "Fan Fault" lights up at the same time.

- Relay contact 2-1 closed: fan running.
- Relay contact 2-3 closed: fan failure.

Relay contact 2-3 closed is also the normal position, i.e. position when the power is off (intrinsic safety). The position of the jumpers on the power supply unit is shown in Section 4.3.

Caution:

Jumper F-R must be opened if switching-off cannot take place immediately. In this case you must ensure that the power supply is switched off at the latest after 60 s. This can be achieved e.g. using a time-delay relay. This prevents modules being damaged by overheating.

3.2.2 Setting the Battery Monitoring

Using jumper RR-LL on the power supply unit you can select whether the signalling relay ("Monitor Output") is to be switched not only upon a fan failure but also upon a battery failure:

- Jumper RR-LL open (factory setting): Relay only signals fan failure.
- Jumper RR-LL closed: Relay signals fan and battery failures.

The signalling relay ("Monitor Output") is switched in the event of a battery failure or standstill of one or both fans. The LED "Batt Low" lights up at the same time.

- Relay contact 2-1 closed: battery OK or (optionally) fan running.
- Relay contact 2-3 closed: battery faulty or (optionally) fan failure.

The position of the jumper on the power supply unit is shown in Section 4.3.

Note:

The signalling relay is activated if a fan or the battery fails. The signalling relay must be wired by the user to adapt it to both faults.
If the signalling relay is triggered in the event of a battery failure, and as a result the PLC is switched off, the program in the user memory is lost. This can be avoided by having an external back-up voltage applied to the sockets on the front panel of the power supply unit while the PLC is being switched off (see Fig. 8).

4 Operation of the S5-135U Central Controller

Before you start up the programmable controller, read the notes in the following section. These notes explain the requirements for operating a PLC and contain useful information about starting up and operating the S5-135U system.

The S5-135U programmable controller of type 6ES5 135-3UA11/-3UA21 is set at the factory for operation with 230 V AC. If you want to operate the PLC with 120 V AC remove the power supply unit and change the slider switch settings to 120 V (see Fig. 7). Before refitting the unit please stick the label "120 V" over the printed label "230 V". When you refit the power supply unit make sure that the short protective ground cable is reconnected to the power supply unit and housing.

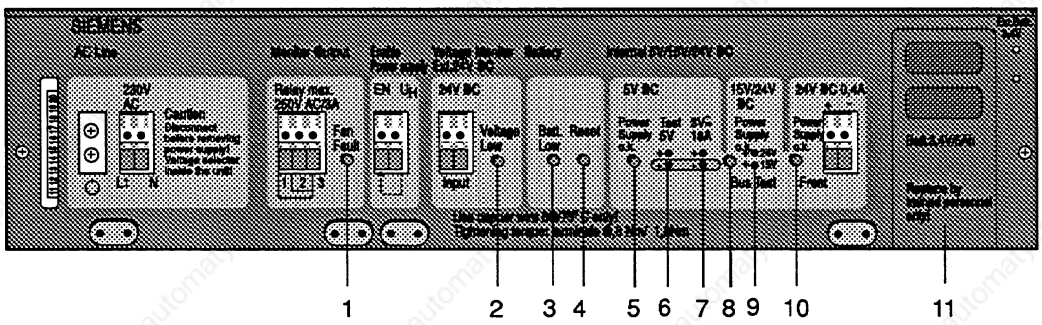
4.1 General Notes on the Power Supply Unit

- No voltages > 50 V must occur between the power supply connections and the protective ground conductor of the power supply unit.
- The protective ground conductor must always be connected as well as the jumper between the CC rack and the front panel of the power supply.
- If there is an overvoltage at the internal DC supply voltages U_{o1} = +5 V and U_{o3} = +15 V, the power supply unit is switched off without loss of data. A voltage ≤ 0.5 V (see Chapter 6, Technical Data, for overvoltage switch-off) is present at U_{o1} and U_{o3} in the switched-off condition.
  The power supply unit can be started up again by switching the external power supply off and then on again to reset the memory flip-flop, provided that the overvoltage was not caused by an internal fault.
- The power supply unit will only function correctly if the +5 V side has a load of at least 1 A (2 A with power supply unit -3LF12).
- An air filter [1] can be installed in the base of the power supply unit housing.
- Make sure the voltage level and polarity are correct when using an external back-up voltage.
- **The back-up battery is supplied loose and must be fitted before you start up. Without the battery, the PLC will not start when the power is switched on. The battery must first be inserted. Then press the RESET button. Then perform an overall reset.**
- The wire jumper for "enable power supply" from U_H to EN enables the power supply. By suitable wiring of the monitoring outputs with the EN inputs, you can block the PLC in the event of a fault (see Section 3.2).

1) See Catalog ST 54.1 and Appendix for order nos.

- Undervoltage or the absence of voltage at the input terminals "voltage monitor" for 24 V DC activates the signal BASP in the CC and connected expansion units so that all digital outputs are disabled (the function can be disabled with jumper BA-EX). In this situation the CPUs do not recognize that the BASP signal is active. If you want the CPUs to recognize this, a digital output must be set constantly to 1 and be connected to an input that is scanned at least once per cycle.

4.2 Operating and Display Elements of the Power Supply Unit



1 LED "Fan Fault"
The red LED lights up if a fan fault has occurred. The power supply unit then switches off (jumper F-R closed) with a delay of approx. 6 to 10 s. The jumper F-R must be opened if the programmable controller cannot be switched off immediately for technical reasons. The programmable controller must always be switched off within 60 s (overheating of module).

2 LED "Voltage Low"
The red LED lights up if an undervoltage is present at the load voltage monitor input
(omitted with power supply unit -3NA12).

3 LED "Batt. Low"
The yellow LED lights up if the battery voltage has dropped below 2.7 V; the data buffered in the RAM are lost following "power off/on".

4 Key "Reset"
If the programmable controller is in the power-off state, the battery must be replaced following mains-on and with the LED "Batt. Low" lit. Press this key after replacing the battery, the LED "Batt. Low" then goes out.

5 LED "Power Supply O.K."
The green LED lights up when the output voltage of 5 V is present.

6 Test sockets "Test 5 V"
For checking the output voltage V_{o1} (standard setting: 5.1 V DC ± 0.5 %)

7 Test sockets "3 V $\triangleq$ 18 A"
For checking the output current I_{o1} (3 V $\triangleq$ max. output current of respective power supply unit).

8 LED "Power Supply O.K." (Bus)
The green LED lights up when the output voltage of 15 V (if the 15 V supplementary module is used) **and** the output voltage of 24 V are present.

9) Test sockets "15 V/24 V DC" (Bus)
a) For checking the output voltage V_{o2} (24 V DC +25%/-17%)
b) For checking the output voltage V_{o3} (15 V DC ± 5% provided the 15 V supplementary module is plugged in)

10 LED "Power Supply O.K." (terminal)
The green LED lights up when the output voltage for the enable supply is present at the terminal "24 V DC".

11 Battery plug-in

Fig. 10 Operating and display elements of power supply units

4.3 Functions and Locations of Jumpers on the Power Supply Units

Function	Jumpers
Battery monitoring ($\overline{BAU}$) on Battery monitoring (BAU) off	NN-MM closed [1] NN-MM open
Switching-off of power supply following fan fault Without switching-off of power supply following fan fault (only LED signal, relay)	**F-R closed** [1] F-R open
Operation with load voltage monitoring Operation without load voltage monitoring	**BA-EX open** [1] BA-EX closed
Setting the signal relay (relay contact 2-3 closed)	
• **By fan fault always**	Independent of jumpers
• **In addition by battery fault message** Battery undervoltage (V_{BATT} < 2.7 V) leads to a battery fault message (can be switched off using jumper MM-NN). In addition to display "Batt.Low" and output of the signal $\overline{BAU}$, the signal relay can be activated via the jumper RR-LL if the following power supplies are used: 6ES5 955-3NA11 from version 9 onwards 6ES5 955-3LF12 from version 5 onwards 6ES5 955-3LC14 from version 7 onwards The jumper RR-LL is irrelevant with the other power supplies.	RR-LL closed
• **Without battery fault message**	**RR-LL open** [1]
• **In addition by undervoltage message** The signal $\overline{BASPA}$ = Low is output with an undervoltage at the monitoring input (V_M < 20 V -25%; can be switched off with jumper BA-EX) or with an undervoltage at the output (V_o < 4.75 V)	BB-AA closed
• **Without undervoltage message**	**BB-AA open** [1]

[1] As supplied

Table 5 Jumper settings of the power supply unit

Jumper locations

The jumper settings shown correspond to the factory settings.

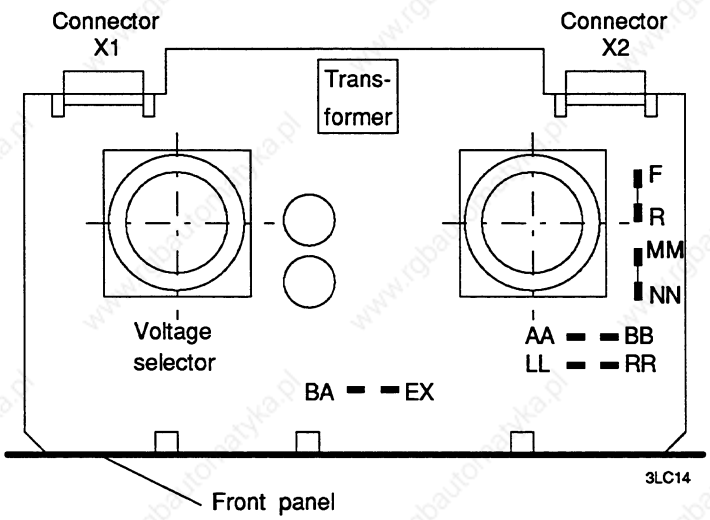


Fig. 11 Power supply unit 6ES5 955-3LC14

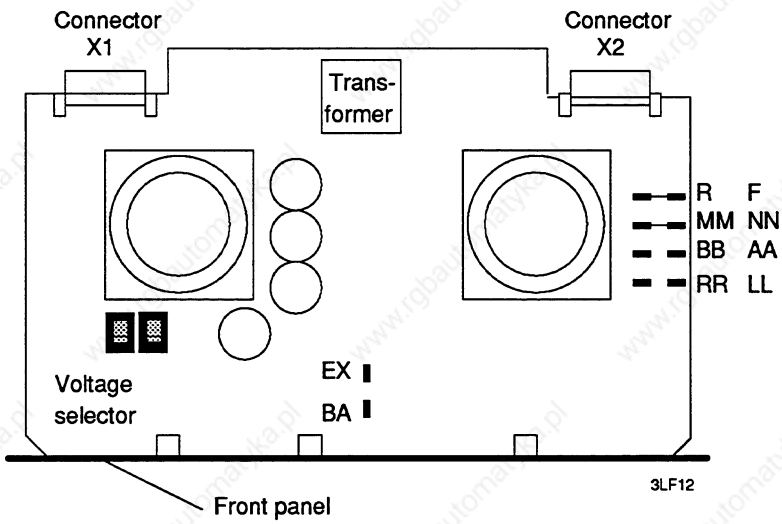


Fig 12 Power supply unit 6ES5 955-3NC13

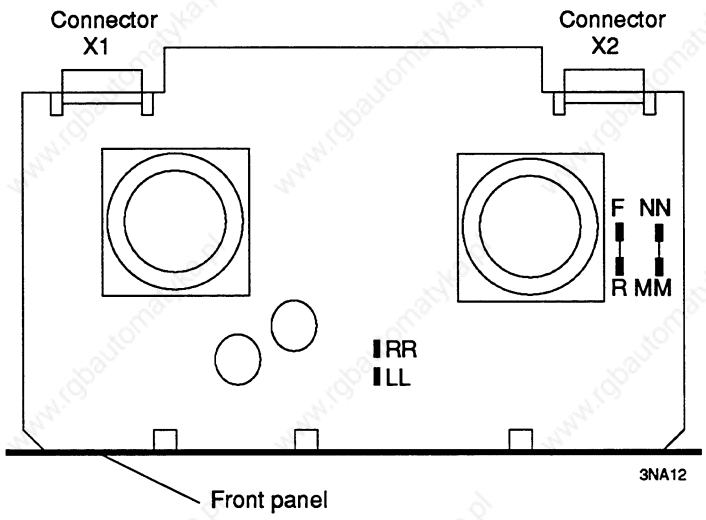


Fig. 13 Power supply unit 6ES5 955-3NA12

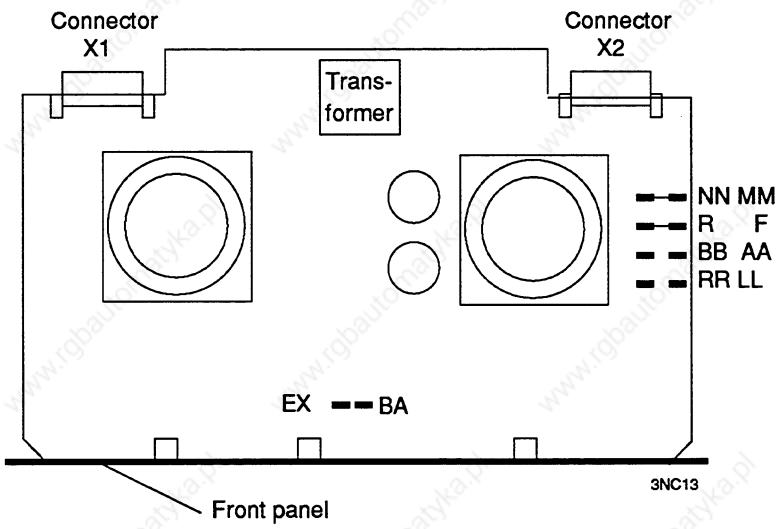


Fig. 14 Power supply unit 6ES5 955-3NC13

4.4 Power Supply Behavior in Event of Faults

If the power supply is switched off, relay contact 2-3 is closed and relay contact 1-2 open.
If there are no faults, relay contact 1-2 is closed and relay contact 2-3 open.
In addition to fan faults, other faults (see jumper description) can set the signal relay to the normal position (relay contact 2-3 closed) by means of appropriate jumper settings.

The following table shows the response of the power supply in the event of faults (condition: jumper MM-NN closed, jumper BA-EX open).

Fault and jumper settings / Reaction				Message: LED "Fan fault"	Message: Closed relay contact	Output voltage (5 V) switched off
Enable (jumper EN-UH) present	No fan fault	No battery fault and no undervoltage message		Dark	1-2	No
		Battery fault	RR-LL closed		2-3	
			RR-LL open		1-2	
		Under-voltage message	BB-AA closed		2-3	
			BB-AA open		1-2	
	Fan fault	F-R open		Lights up	2-3	
		F-R closed				Yes
No enable (jumper EN-UH)	No fan fault	No battery fault		Dark	1-2	
		Battery fault	RR-LL closed		2-3	
			RR-LL open		1-2	
		Under-voltage message	BB-AA closed		2-3	
			BB-AA open		1-2	

Table 6 Fault message and reaction of the power supply unit

4.5 Start-Up and Functional Test

Requirement: 1 S5-135U with one processor (CPU) and 1 RAM submodule.

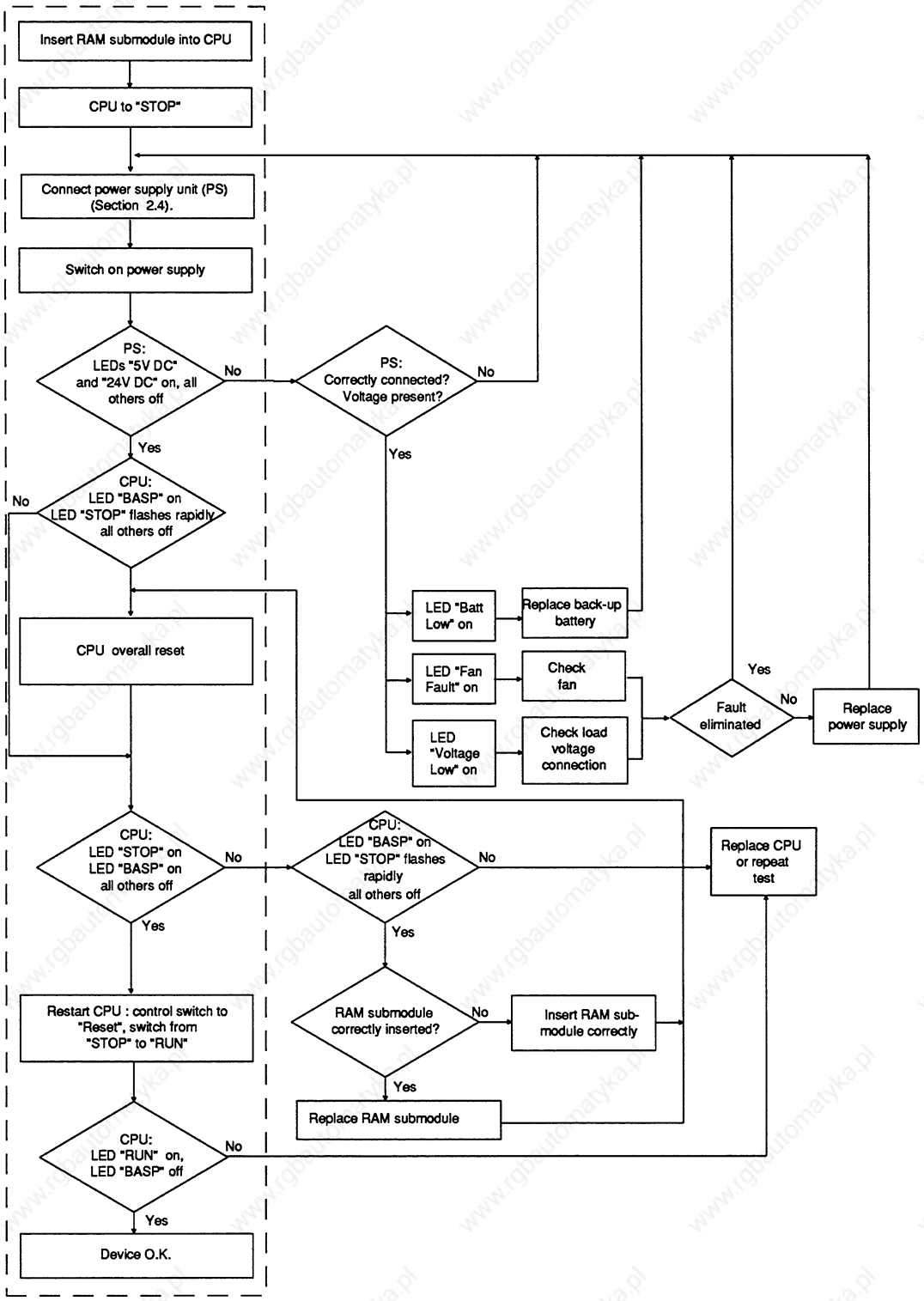
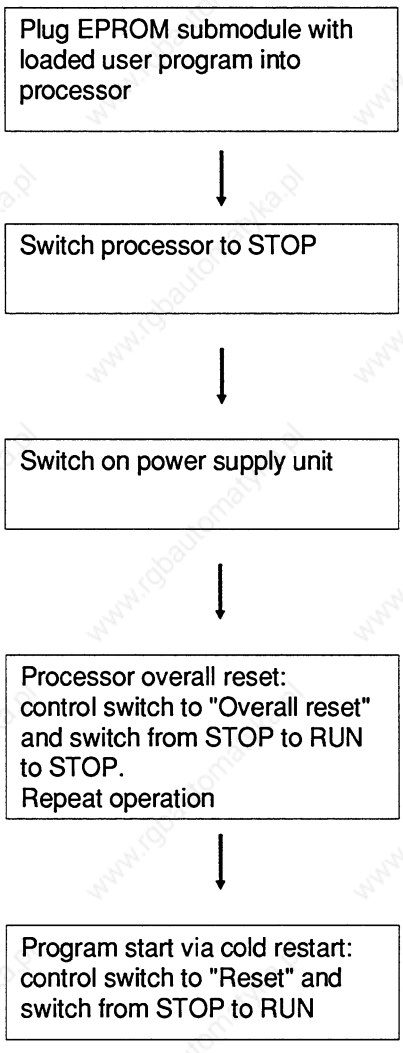


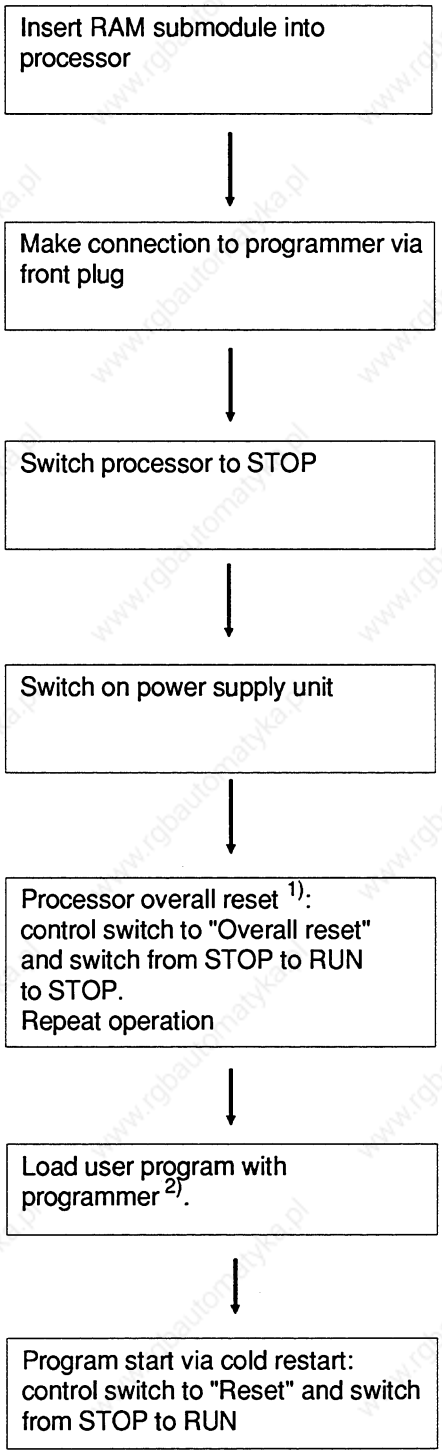
Fig. 15 Starting up

4.5.1 Restart with Single Processor Operation

Operation with EPROM submodule:



Operation with RAM submodule:

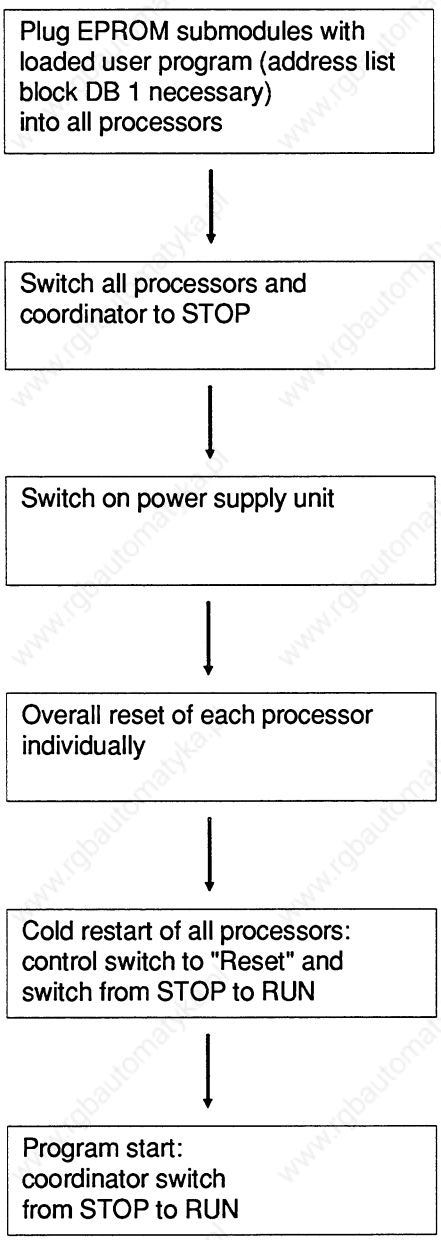


1)If you use a battery-backed RAM submodule whose contents you wish to retain, carry out the overall reset using a different RAM submodule. Then switch off the programmable controller and replace the RAM submodule by the battery-backed RAM submodule.

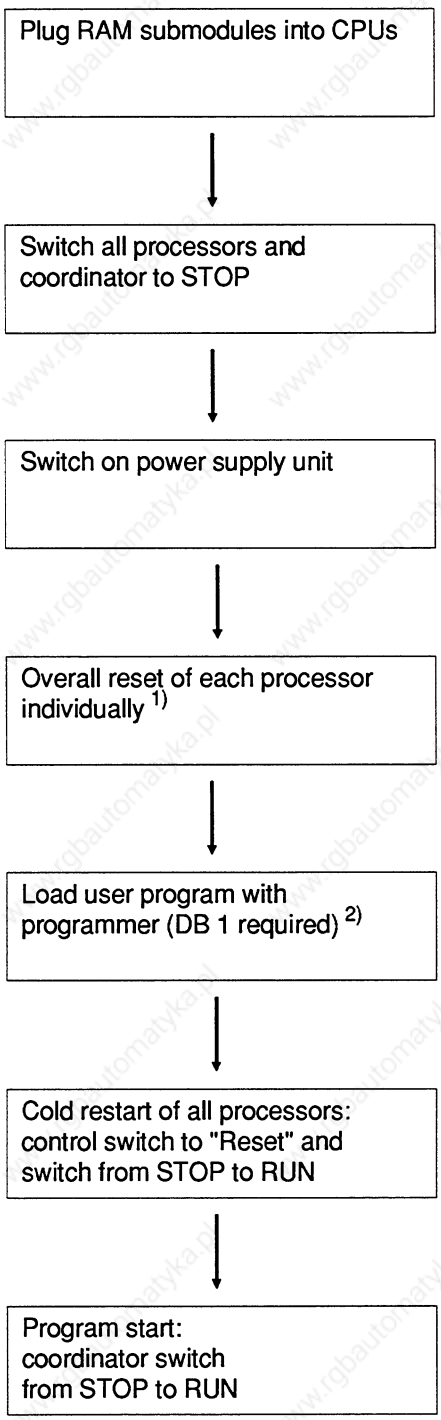
2)This step is omitted if you use a battery-backed RAM submodule in which a program is already loaded.

4.5.2 Restart with Multiprocessor Operation

Operation with EPROM submodules:



Operation with RAM submodules:



1)If you use a battery-backed RAM submodule whose contents you wish to retain, carry out the overall reset using a different RAM submodule. Then switch off the programmable controller and replace the RAM submodule by the battery-backed RAM submodule.

2)This step is omitted if you use a battery-backed RAM submodule into which a program is already loaded.

4.5.3 Checklist for Starting Up

Start up the PLC in the order described here. This brings you to the first trial run with the CPU.

The chapters of the publications describing the steps in detail are shown in brackets.

To avoid the start up being too complex at this stage, you should simply begin with one CPU and without any expansion units.

1) Install the PLC so that air can circulate freely. If you are using several devices (PLC and EU) in one cabinet, make sure the required clearances are maintained and if necessary install air baffles (Installation Guidelines, Part 2, Section 3.3)

2) Fit the back-up battery (Section 5.3).

3) Install the CPU and set the mode selector to STOP.

4) Connect the power supply to the monitoring input and when using the 230/120 V AC supply the 24 V load voltage.

5) Switch on the power and, if present, the 24 V supply: green LED "Power supply o.k." in the "DC 5 V" and "DC15/24 V" fields and the yellow LED "Batt. Low" light up.

6) Press the RESET button on the power supply unit. The yellow LED "Batt. Low" goes out (Section 4.2)

7) Hold the CPU button in the "overall reset" position and switch the mode selector from STOP to RUN: the "STOP" LED flashes quickly.
Repeat this step: the LED "STOP" is lit constantly.
Hold the button in the "RESET" position and switch the selector from STOP to RUN: the green LED "RUN" lights up, the small yellow LED "BASP" goes out (Section 4.5).

The CPU is now running through an empty cycle. After switching off the power supply, you can now plug the other devices into the rack and connect the expansion units. Remember to plug in modules only in the permitted slots (Section 1.2.1), the addressing possibilities with the P and O areas and the settings on the interface modules (U Periphery) and the installation instructions (Part 2).

When starting up with more than one CPU remember that the address lists must be programmed in DB 1 of all CPUs. Refer to the "Multiprocessor Operation Instructions" for detailed information about multiprocessing.

5 Maintenance of the S5-135U Central Controller

If measurements or tests are required on active devices, the accident prevention regulations (VBG 4.0) must be observed and suitable tools used.

Warning:

Repairs on automation equipment must only be carried out by the **Siemens servicing department** or by **qualified personnel** (see above). When replacing parts or components, only use those listed in the Catalog ST 54.1 or in the Appendix in Part 12 of this manual. Unauthorized opening and improper repairs may lead to death, severe personal injury or extensive damage to property.

Always remove the mains plug or open the isolating switch before opening the device.

Only use replacement fuses of the same type.

5.1 Removing and Inserting S5 Modules

Warning:

I/O modules with an active enable circuit may only be removed during operation if the enable voltage is switched off. This is achieved when the front plug is removed.
All inputs of this module are read into the process image of the inputs as "zero" if the enable voltage is missing, if the front plug is disconnected or if the module is removed. With a direct access to the process peripherals, the signal statuses are also written into ACCU 1 as "zero".
With all other modules you must first switch off the device before removing or inserting a module. Removing or inserting these modules while voltage is applied can damage the module and lead to undefined system statuses.

You must ensure in the organization block in which the acknowledgement delay is checked that a hazardous condition in the process or on the machine cannot occur if a fault occurs or when a module is replaced.

The CPU 921 (S processor) of the S5-135U programmable controller goes into the STOP mode with a timeout signal when the front connector of an I/O module is removed (because the enable voltage is removed).

Caution:

Dangerous voltages may be present at the front sockets of I/O modules 435, 436, 455 and 456 if the front plugs are removed and inserted during operation. The modules must only be replaced when under voltage by electricians or trained personnel.

If it is not necessary to remove and insert modules during operation, the wiring of the enable circuit (F_+/F_-) can be omitted on certain I/O modules. The jumper for switching over the enable mode must then be removed.
After removing the jumper for the enable mode, the module can be addressed via the S5 I/O bus irrespective of how the enable circuit F_+/F_- is connected. Connection of the enable voltage is no longer necessary. You must never remove or insert I/O modules connected to voltage if the enable circuit is not activated since this can damage the module and lead to undefined system statuses.

5.2 Removing and Inserting Power Supply Units

Power supply units must only be removed when no voltage is applied. The connection between the back-up battery and the backplane bus is retained when the power supply unit is removed, thus ensuring that the user program is still backed-up.

5.3 Replacing the Back-up Battery and the Fans

The back-up battery can be changed without losing any data in the memory if the power supply unit is switched on or if an external voltage (3.4 V) is applied to the sockets "Ext. Batt.". The back-up battery should be replaced every three years regardless of the memory configuration or the extent to which it had been used.

Proceed as follows to replace the battery:

- Pull down the cover.
- Pull the battery module to the front and remove it.
- Replace the battery.
- Make sure the polarity is correct.
- Once the new battery is fitted and the power is on, press the RESET button on the power supply module (see Fig. 10).

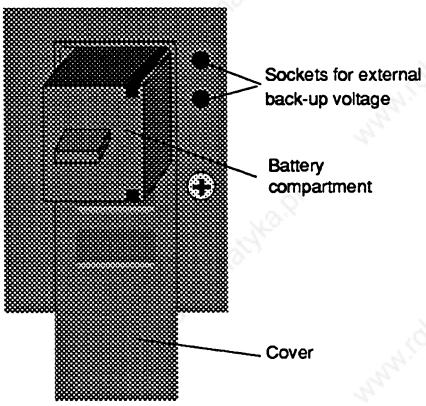


Fig. 16 Battery compartment

Caution:

Ensure correct polarity when fitting a battery or applying an external back-up voltage.

Caution: LITHIUM THIONYL CHLORIDE BATTERY!

Do not dispose of batteries in fire and do not solder on cell body - danger of explosion (max. temperature 100 °C) and do not attempt to recharge them. Do not open batteries. Only replace by batteries of the same type. Order replacement batteries only from Siemens using the order numbers listed in the Appendix. You can then be sure that you are using a short-circuit proof battery.
Old batteries with some charge remaining should be discharged with a 10 Ω resistor or torch bulb until no further no-load voltage can be measured.
Completely discharged batteries no longer contain thionyl chloride and are therefore non-toxic and can be disposed of with normal garbage.
Charged lithium thionyl chloride batteries must otherwise be treated as toxic waste.

Replacing fans

The service life of the fans (see Section 6.1 "Technical Data") depends on the operating time, ambient temperature and ambient conditions. Resulting damage, e.g. on modules, can be avoided in the event of a fan failure during operation if the fan monitoring is switched on (jumper F-R closed); the power supply unit is then switched off.
In particular circumstances, it may be advisable to replace the fans at corresponding maintenance intervals as a preventive measure.

To replace the fans, proceed as follows:

- Switch off the voltage to the power supply.
- Disassemble the power supply.
- Loosen the fixing screws of the fans.
- Disconnect the plug contacts for the fan power supply.

Insert the fans in the reverse order.

The order nos. of the back-up battery and the various fans can be found in the Appendix.

5.4 Pin Assignments of the Power Supply Unit

- The connections of the power supply lines between the power supply unit and the bus PCB are on an 8-pin plug (subminiature plug, 8-pin, fitted with 8 power contacts, series D to MIL-C24308).

For -3LC14; -3NA12; -3NC13:

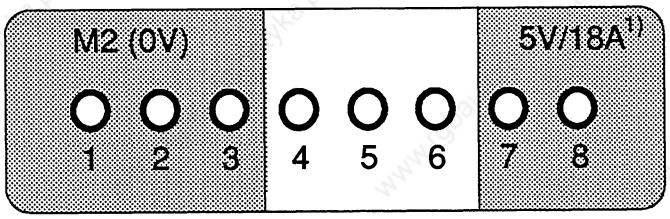


Fig. 17 Plug X1, view from rear of device

[1)] 5 V/10 A with power supply unit -3NA12.

For -3LF12:

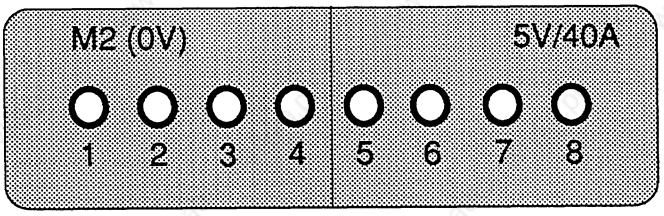


Fig. 18 Plug X1, view from rear of device

- The signal connections on the power supply unit are on a 37-pin plug (subminiature plug connector, 37-pin, series D to MIL-C24308).

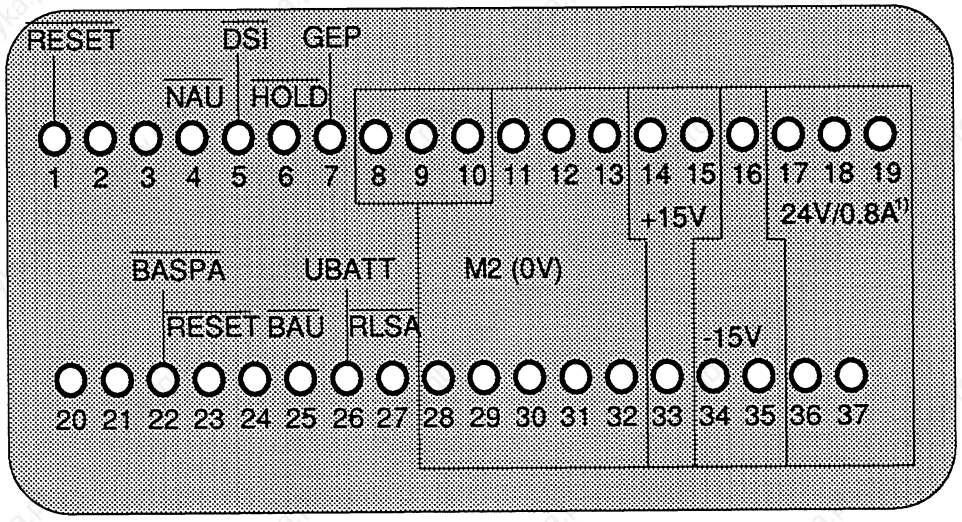


Fig. 19 Plug X2, view from rear of device

[1)] 2.8 A with power supply unit 6ES5 955-3LF12.

5.5 Pin Assignments of the Bus PCB

Slot 3 COR, I/O				
Backplane connector 1	Pin No.	Pin row z	b	d
	2	+5 V	M 5 V	
	4	PL	PESPI	UBAT
	6	RESET	ADB 0	ADB 12
	8	MEMR	ADB 1	ADB 13
	10	MEMW	ADB 2	ADB 14
	12	RDY	ADB 3	ADB 15
	14	DB 0	ADB 4	PL
	16	DB 1	ADB 5	PL
	18	DB 2	ADB 6	PL
	20	DB 3	ADB 7	PL
	22	DB 4	ADB 8	PL
	24	DB 5	ADB 9	PL
	26	DB 6	ADB 10	PL
	28	DB 7	ADB 11	DSI
	30		BASP	PL
	32	PL	M 5 V	BASPA
Backplane connector 2	2	+ 5 V	M 5 V	PL
	4	DB 12	DB 8	PL
	6	DB 13	DB 9	PL
	8	DB 14	DB 10	PL
	10	DB 15	DB 11	PL
	12		PL	PL
	14	NAU	PL	PL
	16	BAU	PL	PL
	18	RESETA	PL	PL
	20		PL	PL
	22	PEU	PL	PL
	24	GEP	PL	PL
	26		PL	PL
	28	PL	PL	PL
	30	M	M	M
		24/15 V	24/15 V	24/15 V
	32	+ 15 V	+ 15 V	+ 15 V

Slots 11, 27, 43, 59 CPU, IRZ, IP-K, CP, I/O, IP			
Pin No.	Pin row z	b	d
2	+5 V	M 5 V	
4	PL	PESPI	UBAT
6	RESET	ADB 0	ADB 12
8	MEMR	ADB 1	ADB 13
10	MEMW	ADB 2	ADB 14
12	RDY	ADB 3	ADB 15
14	DB 0	ADB 4	IRx
16	DB 1	ADB 5	
18	DB 2	ADB 6	
20	DB 3	ADB 7	
22	DB 4	ADB 8	IRE
24	DB 5	ADB 9	IRF
26	DB 6	ADB 10	IRG
28	DB 7	ADB 11	DSI
30	PL	BASP	PL
32	PL	M 5 V	BASPA
2	+ 5 V	M 5 V	
4	DB 12	DB 8	
6	DB 13	DB 9	M 5 V
8	DB 14	DB 10	
10	DB 15	DB 11	
12	M 5 V		
14	NAU		
16	BAU		
18		M 5 V	
20		PL	
22	PEU	PL	PL
24	GEP	M 5 V	
26		PL	PL
28	PL	PL	PL
30	M	M	M
	24/15 V	24/15 V	24/15 V
32	+ 15 V	+ 15 V	+ 15 V

Abbreviations:
COR - Coordinator module
I/O - Input/output
CPU() - CPU mod. 1 . . . 4
IRZ - Interrupt sink *)
IRQ - Interrupt source *)
IP - Intelligent I/O
IP-K - ditto with page address
IM307 - Slot with full 307 functions
IM - Interface module slot
IM-V - IM slot with extra power supply output
PL - Private line
*) Interrupt sink means: module which takes on the interrupt
Interrupt source means: module which generates the interrupt

Table 7 (1ff) Pin assignments of the backplane bus

Slots 19, 35, 51, 75, 83, 91, 99 CP, IP-K, IP, I/O, IRQ				
	Pin No.	Pin row		
		z	b	d
Backplane connector 1	2	+5 V	M 5 V	
	4	PL	PESPI	UBAT
	6	RESET	ADB 0	ADB 12
	8	MEMR	ADB 1	ADB 13
	10	MEMW	ADB 2	ADB 14
	12	RDY	ADB 3	ADB 15
	14	DB 0	ADB 4	IRA
	16	DB 1	ADB 5	IRB
	18	DB 2	ADB 6	IRC
	20	DB 3	ADB 7	IRD
	22	DB 4	ADB 8	IRE
	24	DB 5	ADB 9	IRF
	26	DB 6	ADB 10	IRG
	28	DB 7	ADB 11	DSI
	30		BASP	
	32	PL	M 5 V	BASPA
Backplane connector 2	2	+ 5 V	M 5 V	
	4	DB 12	DB 8	
	6	DB 13	DB 9	
	8	DB 14	DB 10	
	10	DB 15	DB 11	
	12			
	14	NAU	PL	
	16	BAU	PL	
	18			
	20		PL	PL *
	22		PL	
	24	GEP		
	26		PL *	PL
	28	PL	PL	PL
	30	M	M 5 V	M
		24/15 V		24/15 V
	32	+ 24 V		+ 15 V

Slots107, 115, 123, 131 CP, IP-K, IP, I/O, IRQ, IM307			
Pin No.	Pin row		
	z	b	d
2	+5 V	M 5 V	
4	PL	PESPI	UBAT
6	RESET	ADB 0	ADB 12
8	MEMR	ADB 1	ADB 13
10	MEMW	ADB 2	ADB 14
12	RDY	ADB 3	ADB 15
14	DB 0	ADB 4	IRA
16	DB 1	ADB 5	IRB
18	DB 2	ADB 6	IRC
20	DB 3	ADB 7	IRD
22	DB 4	ADB 8	IRE
24	DB 5	ADB 9	IRF
26	DB 6	ADB 10	IRG
28	DB 7	ADB 11	DSI
30		BASP	
32	PL	M 5 V	BASPA
2	+ 5 V	M 5 V	
4	DB 12	DB 8	
6	DB 13	DB 9	
8	DB 14	DB 10	
10	DB 15	DB 11	
12			
14	NAU		
16	BAU		
18		PEU	
20		PL	
22		PL	
24	GEP		
26			PL
28	PL	PL	PL
30	M	M	M
	24/15 V	24/15 V	24/15 V
32	+ 24 V	+ 15 V	+ 15 V

PL* = only in slots 75, 83, 91 and 99

Table 7 (2ff) Pin assignments of the backplane bus

Slots 139, 147, 155 IP, I/O, IM				
Back-plane con-nector 1	Pin No.	Pin row		
		z	b	d
	2	+5 V	M 5 V	
	4	PL	PESPI	
	6	RESET	ADB 0	ADB 12
	8	MEMR	ADB 1	ADB 13
	10	MEMW	ADB 2	ADB 14
	12	RDY	ADB 3	ADB 15
	14	DB 0	ADB 4	
	16	DB 1	ADB 5	
	18	DB 2	ADB 6	M 5 V
	20	DB 3	ADB 7	M 5 V
	22	DB 4	ADB 8	M 5 V
	24	DB 5	ADB 9	M 5 V
	26	DB 6	ADB 10	M 5 V
	28	DB 7	ADB 11	M 5 V
	30		BASP	M 5 V
	32		M 5 V	BASPA
Back-plane con-ector 2	2	+ 5 V	M 5 V	
	4	DB 12	DB 8	
	6	DB 13	DB 9	
	8	DB 14	DB 10	
	10	DB 15	DB 11	
	12			
	14			
	16			
	18	RESET	PEU	
	20			
	22	M 5 V	M 5 V	
	24	M 5 V	M 5 V	
	26	M 5 V	M 5 V	
	28	M 5 V	M 5 V	
	30	M 5 V	M 5 V	
		M 5 V	M 5 V	
	32	M 5 V	M 5 V	

Slot 163 I/O, IM, IM-V			
Pin No.	Pin row		
	z	b	d
2	+5 V	M 5 V	
4	PL	PESPI	+ 5 V
6	RESET	ADB 0	ADB 12
8	MEMR	ADB 1	ADB 13
10	MEMW	ADB 2	ADB 14
12	RDY	ADB 3	ADB 15
14	DB 0	ADB 4	+ 5 V
16	DB 1	ADB 5	+ 5 V
18	DB 2	ADB 6	M 5 V
20	DB 3	ADB 7	M 5 V
22	DB 4	ADB 8	M 5 V
24	DB 5	ADB 9	M 5 V
26	DB 6	ADB 10	M 5 V
28	DB 7	ADB 11	M 5 V
30		BASP	M 5 V
32		M 5 V	BASPA
2	+ 5 V	M 5 V	
4	DB 12	DB 8	
6	DB 13	DB 9	
8	DB 14	DB 10	
10	DB 15	DB 11	
12	+ 5 V	+ 5 V	
14	+ 5 V	+ 5 V	
16	+ 5 V	+ 5 V	
18	RESET	PEU	
20			
22	M 5 V	M 5 V	
24	M 5 V	M 5 V	
26	M 5 V	M 5 V	
28	M 5 V	M 5 V	
30	M 5 V	M 5 V	
	M 5 V	M 5 V	
32	M 5 V	M 5 V	

Table 7 (3ff) Pin assignments of the backplane bus

5.6 Pin Designations of the Interrupt Signals on the Bus PCB

	Interrupt sink*)				Interrupt source*)
Module	CPU 1	CPU 2	CPU 3	CPU 4	I/O / CP
Slot no.	11	27	43	59	19, 35, 51, 67-131
Signal					
$\overline{IRA}$	1d 14				1d 14
$\overline{IRB}$		1d 14			1d 16
$\overline{IRC}$			1d 14		1d 18
$\overline{IRD}$				1d 14	1d 20
$\overline{IRE}$ Not evaluated by the CPU	1d 22	1d 22	1d 22	1d 22	1d 22
$\overline{IRF}$ Not evaluated by the CPU	1d 24	1d 24	1d 24	1d 24	1d 24
$\overline{IRG}$ Not evaluated by the CPU	1d 26	1d 26	1d 26	1d 26	1d 26

Table 8 Pin designation of the interrupt signals (on connector X1)

*) Designations according to ISO 2382/XVI - 1978 (DIN 44301)
Interrupt sink = module that receives the interrupt
Interrupt source = module that generates the interrupt

6 Technical Data of the S5-135U Central Controller

This power supply is UL and CSA listed.

Safety	
Device complies with	VDE 0160. Protection against overvoltage complying with VDE 0160 A1 (April 89), Section 6.3.4 Overvoltage proof achievable with additional measures.
Protection class	I
Degree of protection	IP 20 to IEC 529/DIN 40050 with covering of empty slots by dummy front panels
Climatic ambient conditions (tested according to IEC 68-2/-1/2/3)	
Ambient temperature in operation	0 to 55 °C (+32 to +131 °F) (Air intake measured at the lower air inlet of the power supply unit)
Transport and storage temperature	- 40 to 70 °C (-40 to +158 °F)
Temperature change	
operation	max. 10 K/h
transport and storage	max. 20 K/h (3 h adaptation time when delivered below 0 °C because of possible condensation)
Relative humidity	
operation	max. 95 % at 25 °C, no condensation
transport and storage	max. 95 % at 25 °C, no condensation
Operating altitude	
operation	- 1000 m to +1500 m, when used above 1500 m it is advisable to contact the manufacturer because of the required cooling conditions.
transport and storage	- 1000 m to + 3500 m
Toxic substances	
SO_2	Max. 0.5 ppm (relative humidity below 60 %)
H_2S	Max. 0.1 ppm (relative humidity below 60 %)
Mechanical ambient conditions (tested according to IEC 68-2-6)	
Vibrations during operation	10 to 58 Hz (constant amplitude 0.15 mm) 58 to 500 Hz (constant acceleration 2 g)

Table 9 (1ff) Technical data

Noise immunity (EMC)	
Radio interference suppression	to DIN VDE 0871(CISPR, Publication No. 11 and CENELEC, HD 344)
Limit class	A
Conducted interferences on:	
AC voltage supply lines (230 V AC)	2 kV to IEC 801-4 (Burst) 1 kV to IEC 801-5 Line against line (µs pulses) 2 kV nach IEC 801-5 Line against ground (µs pulses)
DC voltage supply lines (24 V-supply)	1 kV to IEC 801-4 (Burst)
Signal lines (24 V DC)	1 kV to IEC 801-4 (Burst)
Signal lines (230 V AC)	2 kV to IEC 801-4 (Burst)
Noise immunity to discharges of static electricity	A noise immunity of 8 kV must be guaranteed by an appropriate design
Noise immunity with respect to RF radiation	RF radiation according to VG 95373 LFO2G limit class 3 (up to 200 MHz), corresponding to 3 V/m
Back-up battery	
Type	Lithium-thionyl-chloride
Capacity	5 Ah at I_{max} = 35 mA
No-load voltage	3.6 V
Operating voltage	3.4 V
Storage life	approx. 10 years
Service life during operation	max. 3 years
Mechanical design	
Mechanical requirements	Installation in fixed devices not free from vibrations; installation on ships and vehicles with observance of special installation specifications, but not on the motor
Weight	approx. 14 kg
Dimensions (w x h x d)	482.6 x 432 x 310 mm

Table 9 (2ff) Technical data

6.1 Power Supply Unit 6ES5 955-3LC14

This power supply unit is UL and CSA listed.

Input	
Rated input voltage U_{EN}	230 / 120 V AC + 10 % /- 18.7 % [1]
Undervoltage signal U_E	< 187 V AC (or 93 V AC) [2]
Input frequency f_E	48 to 63 Hz
Input current I_{EN} with rated load and U_{EN} = 230 V (or 120 V)	1.25 A (or 2.5 A) [2]
Inrush current peak I_{Emax}	100 A (or 50 A) [2]
Efficiency with rated load, with fan	typically 61 %
Stored energy time during power failure	> 5 ms
Power factor cos φ	0.65
Input fuse	4 A fast-blow; 250 V; 6.3 mm x 32 mm; location F26 (printed on power supply board)
Output 1	
Rated output voltage U_{AN1}	5.1 V DC ± 0.5 %
Setting range of output voltage	(0.95 to 1.05) x U_{AN1}
Rated output current I_{AN1}	18 A DC
Ripple	≤ 1 % of U_{A1}
Dynamic voltage tolerance With load surge from 50 % to 100 % I_N	≤ 5 % of U_{A1}
Settling time	≤ 5 ms
Overvoltage shut-down U_{A1}	6 V ± 5 %
Undervoltage signal U_{A1}	4.75 + 5 %
Current limiting with overload	(1.05 to 1.15) x I_{AN1}
Output 2	
Rated output voltage U_{AN2}	24 V DC + 25 % / - 17 %
Rated output current I_{AN2}	0.8 A DC [3]
Ripple	≤ 5 % of U_{A2}
Fuse for overcurrent protection	1.5 A fast-blow; 250 V; 6.3 mm x 32 mm; location F90 (printed on power supply board)
Output 3 with supplementary module	
Rated output voltage U_{AN3}	15 V DC ± 5 %
Rated output current I_{AN1}	0.5 A DC [3]
Ripple	≤ 5 % of U_{AN3}
Overvoltage shut-down U_{A1}	U_{A3} ≥ 18.5 V
Undervoltage signal (LED on front panel)	U_{A3} ≤ 14 V ± 3 %
Overcurrent protection I_{A3} by current limiting	0.5 to 1.5 A

[1] Voltage selector
[2] Values in brackets for operation with 120 V
[3] Total of output currents (I_{A2} + I_{A3} + I_{A4}) ≤ 0.8 A DC

Table 9 (3ff) Technical data

Output 4: 24 V front	
Rated output voltage U_{AN4} Rated output current I_{AN4} Current limiting (reaction threshold) Undervoltage signal (LED on front panel) Capacitive load	24 V DC + 6 V / - 5 V 0.4 A [1] ≥ 0.44 A 16 V ± 20 % max. 100 nF
Fans	
Fan type Input voltage Delivery rate per fan Fan monitoring Service life of fan	2 axial fans 120 V AC, selectable (serial/parallel) 160 m^3/h (no load) Flow monitoring with thermistors as sensors; stoppage of 1 or both fans is recognized and signalled externally by LEDs and relay contacts. With jumper F-R it is possible to select whether or not the 5 V output voltage is also switched off. typically 30 000 h to 40 000 h at 55 °C typically 40 000 h to 50 000 h at 30 °C
Additional monitoring	
24 V load voltage (external voltage monitor)	≥ 14 to 20 V
Electrical isolation primary/secondary	yes

[1] Total of output currents ($I_{A2} + I_{A3} + I_{A4}$) ≤ 0.8 A DC

Table 9 (4ff) Technical data

6.2 Power Supply Unit 6ES5 955-3LF12

This power supply unit is UL and CSA listed.

Input	
Rated input voltage U_{EN}	230 / 120 V AC + 10 % /- 18.7 % [1]
Undervoltage signal U_E	< 187 V AC (or 93 V AC)
Input frequency f_E	48 to 63 Hz
Input current I_{EN} with rated load and U_{EN} = 230 V (or 120 V)	2.95 A (5.9 A)
Inrush current peak I_{Emax}	200 A (100 A)
Efficiency with rated load, with fan	typically ≥ 70%
Stored energy time during power failure	> 5 ms
Power factor cos φ	0.73
Input fuse	6 A fast-blow; 250 V; 6.3 mm x 32 mm; location F107 (printed on power supply board)
Output 1	
Rated output voltage U_{AN1}	5.1 V DC ± 0.5 %
Setting range of output voltage	(0.95 to 1.05) x U_{AN1}
Rated output current I_{AN1}	40 A DC
Ripple	≤ 1 % of U_{A1}
Dynamic voltage tolerance with load surge from 50 % to 100 % I_N	≤ 5 % of U_{A1}
Settling time	≤ 5 ms
Overvoltage shut-down U_{A1}	6 V ± 5 %
Undervoltage signal U_{A1}	4.75 + 5 %
Current limiting with overload	(1.05 to 1.15) x I_{AN1}
Output 2	
Rated output voltage U_{AN2}	24 V DC + 25 % / - 17 %
Rated output current I_{AN2}	2.8 A DC [2]
Ripple	≤ 5 % of U_{A2}
Fuse for overcurrent protection	4 A fast-blow; 250 V; 6.3 mm x 32 mm; location F255 (printed on power supply board)
Output 3 with supplementary module	
Rated output voltage U_{AN3}	15 V DC ± 5 %
Rated output current I_{AN1}	2 A DC [2]
Ripple	≤ 5 % of U_{AN3}
Overvoltage shut-down U_{A1}	U_{A3} ≥ 18.5 V
Undervoltage signal (LED on front panel)	U_{A3} ≤ 14 V ± 3 %
Overcurrent protection I_{A3} by current limiting	2 to 3 A

[1] Voltage selector
[2] Total of output currents (I_{A2} + I_{A3} + I_{A4}) ≤ 2.8 A DC

Table 9 (5ff) Technical data

Output 4: 24 V front	
Rated output voltage U_{AN4} Rated output current I_{AN4} Current limiting (reaction threshold) Undervoltage signal (LED on front panel) Capacitive load	24 V DC + 6 V / - 5 V 0.4 A [1] ≥ 0.44 A 16 V ± 20 % max. 100 nF
Fans	
Fan type Input voltage Delivery rate per fan Fan monitoring Service life of fan	2 axial fans 230/120 V AC, selectable 160 m^3/h (no load) Flow monitoring with thermistors as sensors; stoppage of 1 or both fans is recognized and signalled externally by LEDs and relay contacts. With jumper F-R it is possible to select whether or not the 5 V output voltage is also switched off. typically 30 000 h to 40 000 h at 55 °C typically 40 000 h to 50 000 h at 30 °C
Additional monitoring	
24 V load voltage (external voltage monitor)	≥ 14 to 20 V
Electrical isolation primary/secondary	yes

[1] Total of output currents ($I_{A2} + I_{A3} + I_{A4}$) ≤ 2.8 A DC

Table 9 (6ff) Technical data

6.3 Power Supply Unit 6ES5 955-3NA12

This power supply unit is UL and CSA listed.

Input	
Rated input voltage U_{EN}	24 V DC +25 %/-17 %
Undervoltage signal U_E	< 20 V DC
Input current I_{EN} with rated load and U_{EN} = 230 V	48.5 A
Inrush current peak I_{Emax}	100 A
Efficiency with rated load, with fan	typically 60 %
Stored energy time during power failure	> 5 ms
Input fuse	6 A fast-blow; 250 V; 6.3 mm x 32 mm; location F2 (printed on power supply board)
Output 1	
Rated output voltage U_{AN1}	5.1 V DC ± 0.5 %
Setting range of output voltage	(0.95 to 1.05) x U_{AN1}
Rated output current I_{AN1}	10 A DC
Ripple	≤ 1 % of U_{A1}
Dynamic voltage tolerance with load surge from 50 % to 100 % I_N	≤ 5 % of U_{A1}
Settling time	≤ 5 ms
Overvoltage shut-down U_{A1}	6 V ± 5 %
Undervoltage signal U_{A1}	4.75 + 5 %
Current limiting with overload	(1.05 to 1.15) x I_{AN1}
Output 2	
Rated output voltage U_{AN2}	24 V DC + 25 % / - 17 %
Rated output current I_{AN2}	2.8 A DC [1]
Ripple	≤ 5 % of U_{A2}
Fuse for overcurrent protection	1.5 A fast-blow; 250 V; 6.3 mm x 32 mm; location F490 (printed on power supply board)
Output 3 with supplementary module	
Rated output voltage U_{AN3}	15 V DC± 5 %
Rated output current I_{AN1}	2 A DC [1]
Ripple	≤ 5 % of U_{A3}
Overvoltage shut-down U_{A1}	U_{A3} ≥ 18.5 V
Undervoltage signal (LED on front panel)	U_{A3} ≤ 14 V ± 3 %
Overcurrent protection I_{A3} by current limiting	0.5 to 1.5 A

[1] Total of output currents ($I_{A2} + I_{A3} + I_{A4}$) ≤ 2.8 A DC

Table 9 (7ff) Technical data

Output 4: 24 V-Front	
Rated output voltage U_{AN4} Rated output current I_{AN4} Current limiting (reaction threshold) Undervoltage signal (LED on front panel) Capacitive load	24 V DC + 6 V / - 5 V 0.4 A [1] ≥ 0.44 A 16 V ± 20 % max. 100 nF
Fans	
Fan type Input voltage Delivery rate per fan Fan monitoring Service life of fan	2 axial fans 24 V DC 160 m^3/h (no load) Flow monitoring with thermistors as sensors; stoppage of 1 or both fans is recognized and signalled externally by LEDs and relay contacts. With jumper F-R it is possible to select whether or not the 5 V output voltage is also switched off. typically 30 000 h to 40 000 h at 55 °C typically 40 000 h to 50 000 h at 30 °C
Additional monitoring	
24 V load voltage (external voltage monitor)	no
Electrical isolation primary/secondary	no

[1] Total of output currents ($I_{A2} + I_{A3} + I_{A4}$) ≤ 2.8 A DC

Table 9 (8ff) Technical data

6.4 Power Supply Unit 6ES5 955-3NC13

This power supply unit is UL and CSA listed.

Input	
Rated input voltage U_{EN}	24 V DC +25 %/-17 %
Undervoltage signal U_E	< 20 V DC
Input current I_{EN}	
with rated load and	
U_{EN} = 230 V	6.9 A
Inrush current peak I_{Emax}	250 A
Efficiency with rated load, with fan	typically 67 %
Stored energy time during power failure	> 5 ms
Input fuse	15 A medium-blow; 250 V; 6.3 mm x 32 mm; location F1 (printed on power supply board)
Output 1	
Rated output voltage U_{AN1}	5.1 V DC ± 0.5 %
Setting range of output voltage	(0.95 to 1.05) x U_{AN1}
Rated output current I_{AN1}	18 A DC
Ripple	≤ 1 % of U_{A1}
Dynamic voltage tolerance	
With load surge from 50 % to 100 % I_N	≤ 5 % of U_{A1}
Settling time	≤ 5 ms
Overvoltage shut-down U_{A1}	6 V ± 5 %
Undervoltage signal U_{A1}	4.75 + 5 %
Current limiting with overload	(1.05 to 1.15) x I_{AN1}
Output 2	
Rated output voltage U_{AN2}	24 V DC + 25 % / - 17 %
Rated output current I_{AN2}	0.8 A DC [1]
Total current load of the 24V and 15V outputs	≤ 0.8 A
Ripple	≤ 5 % of U_{A2}
Fuse for overcurrent protection	1.5 A fast-blow; 250 V; 6.3 mm x 32 mm; location F90 (printed on power supply board)
Output 3 with supplementary module	
Rated output voltage U_{AN3}	15 V DC± 5 %
Rated output current I_{AN1}	0.5 A DC [1]
Ripple	≤ 5 % of U_{AN3}
Overvoltage shut-down U_{A1}	U_{A3} ≥ 18.5 V
Undervoltage signal	U_{A3} ≤ 14 V ± 3 %
(LED on front panel)	
Overcurrent protection I_{A3} by current limiting	0.5 to 1.5 A

[1] Total of output currents (I_{A2} + I_{A3} + I_{A4}) ≤ 0.8 A DC

Table 9 (9ff) Technical data

Output 4: 24 V-Front	
Rated output voltage U_{AN4} Rated output current I_{AN4} Current limiting (reaction threshold) Undervoltage signal (LED on front panel) Capacitive load	24 V DC + 6 V / - 5 V 0.4 A [1] ≥ 0.44 A 16 V ± 20 % max. 100 nF
Fans	
Fan type Input voltage Delivery rate per fan Fan monitoring Service life of fan	2 axial fans 24 V DC 160 m^3/h (no load) Flow monitoring with thermistors as sensors; stoppage of 1 or both fans is recognized and signalled externally by LEDs and relay contacts. With jumper F-R it is possible to select whether or not the 5 V output voltage is also switched off. typically 30 000 h to 40 000 h at 55 °C typically 40 000 h to 50 000 h at 30 °C
Additional monitoring	
24 V load voltage (external voltage monitor)	≥ 14 to 20 V
Electrical isolation primary/secondary	yes

[1] Total of output currents ($I_{A2} + I_{A3} + I_{A4}$) ≤ 0.8 A DC

Table 9 (10ff) Technical data

7 Index

A

Air filter 25
Ambient conditions 47

B

Back-up battery 39, 48
Battery failure 24
Battery voltage 27
Bus board 6

C

Cable duct 6
Coordinator 15
CSA 47, 49, 51, 53, 55

D

Degree of protection 47

E

EN input 21
Enable circuit 37
Enable power supply 21
Enable voltage 37
Equipotential bonding conductor 14

F

Fans 50
Fiber-optic interface module 14
Front plug 20

H

Heat exchanger 22

I

I/O address 8
I/O area 8
Individual locking mechanism 6
Installation specifications 48
Interface cable 10
Interrupt signals 45

L

Locking rail 6

M

Maintenance intervals 40
Mechanical design 48
Monitor output 21, 24
Multiplexer function 15
Multiprocessor operation 14

N

Noise immunity 48

O

O area 8
O page number 8

P

Power supply unit 6, 49
Protection class 47

R

Reset 25, 27

S

Safety 47
Screen connection 12
Signalling relay 24
Single processor operation 14
Standard peripheral devices 15
Supplementary module 19

T

Temperature monitoring device 22
Terminator 10
Thermostat 22

U

UL 47, 49, 51, 53, 55

V

Voltage monitor 21, 26

W

Winchester disk drive 23

SIEMENS

SIMATIC S5

CPU 921S Processor

6ES5 921-3UA11
6ES5 921-3UA12

Instructions

C79000-B8576-C262-06

Contents **Page**

1	**Description**	**3**
1.1	Application	3
1.2	Design	3
1.3	Mode of Operation	4
1.3.1	Notes on the Block Diagram	4
1.3.2	User Memory Submodule	5
1.3.3	Interrupt Processing	6
1.4	Memory Allocation	7
1.5	Technical Data	8
2	**Installation**	**9**
2.1	Removing and Inserting Modules	9
2.2	Slots in the Central Controller	9
3	**Operation**	**9**
3.1	Operating Mode Options	9
3.2	Operator Controls and Displays	10
3.3	Restart Modes	13
3.3.1	Non-Retentive Manual Cold Restart	14
3.3.2	Retentive Manual Cold Restart	14
3.3.3	Retentive Automatic Cold Restart	14
4	**Maintenance**	**15**
4.1	Pin Assignments of the Backplane Connectors	15
4.2	Pin Assignments of the Front Connectors	15

1 Description

1.1 Application

The S processor is designed to be installed in the S5-135U programmable controller. Modular CPU expansion makes multiprocessing with up to 4 CPUs possible without rewiring.

The S processor is particularly suitable for fast processing of control tasks (binary signal processing). Byte processing (computing, closed-loop control) is also possible.

By means of the coordinator 923 C the S processor -3UA12 can be operated on-line with the programmer (PG). It is no longer necessary to change the PG-PLC cable connectors. This function is not implemented in the PLC models -3KA12 and -3KB12 of the S5-135U (see Operating Instructions for the coordinator 923 C).

The programming language is STEP 5.

1.2 Design

The modules are designed as plug-in PCBs in double Euroformat. Two 48-pin blade connectors of the "row 2" type connect the modules to the S5 bus in the subrack.

The width of the front panel takes up 1 1/3 standard slots. In the front panel there is a rectangular recess available for a user memory submodule.

The controls consist of a selector with two possible settings and a second switch with three possible settings. Operating statuses are indicated by a red and green LED. Four small red LEDs indicate errors and reactions to errors.

The S processor can be connected to the PG 675 programmer or the DG-355U diagnostic unit via a 15-pin front connector.

1.3 Mode of Operation

1.3.1 Notes on the Block Diagram

The **byte processor** comprises the microprocessor 8031 and the necessary memories. It processes the operating system functions and executes the word commands of the user program.

The **bit processor** comprises the microprogram control and the logic unit. It fetches the user program commands, carries out the complete processing of the bit commands or transfers the word commands to the byte processor. I/Os are also accessed via the bit processor.

The **user memory submodule** contains the user commands in the MC5 code.

In the **command register** the user commands are divided into opcode and parameters.

The **system data memory** contains data for the operating system and the process image for digital inputs and outputs, flags, timers and counters.

The **logic unit** performs logic operations on binary information according to the processed bit commands. It also executes set and reset commands. Processing is carried out to a depth of seven bracket levels.

The **mapping PROM 1** converts the currently present opcode to a microprogram control entry address.

The **microprogram control** generates the control bits necessary for the execution of the particular command.

The **mapping PROM 2** supplies the entry addresses for the execution of commands by the microprocessor 8031.

The **system program memory** contains the whole operating system of the S processor, the interfacing software and the MC5 interpretation for word commands.

The **RAM for internal data** expands the memory area of the microprocessor 8031.

The **address counter** is clocked by the microprogram control and indicates the memory cell in which the user command resides.

With jump commands (word commands) the microprocessor 8031 reads the current address counter status in order to save the return address and sets the address counter to the new count.

The **interrupt register** stores all events which lead to the execution of commands being interrupted except for the clock stoppage.

The **clock** generates the basic clock frequency for the byte processor and the bit processor.

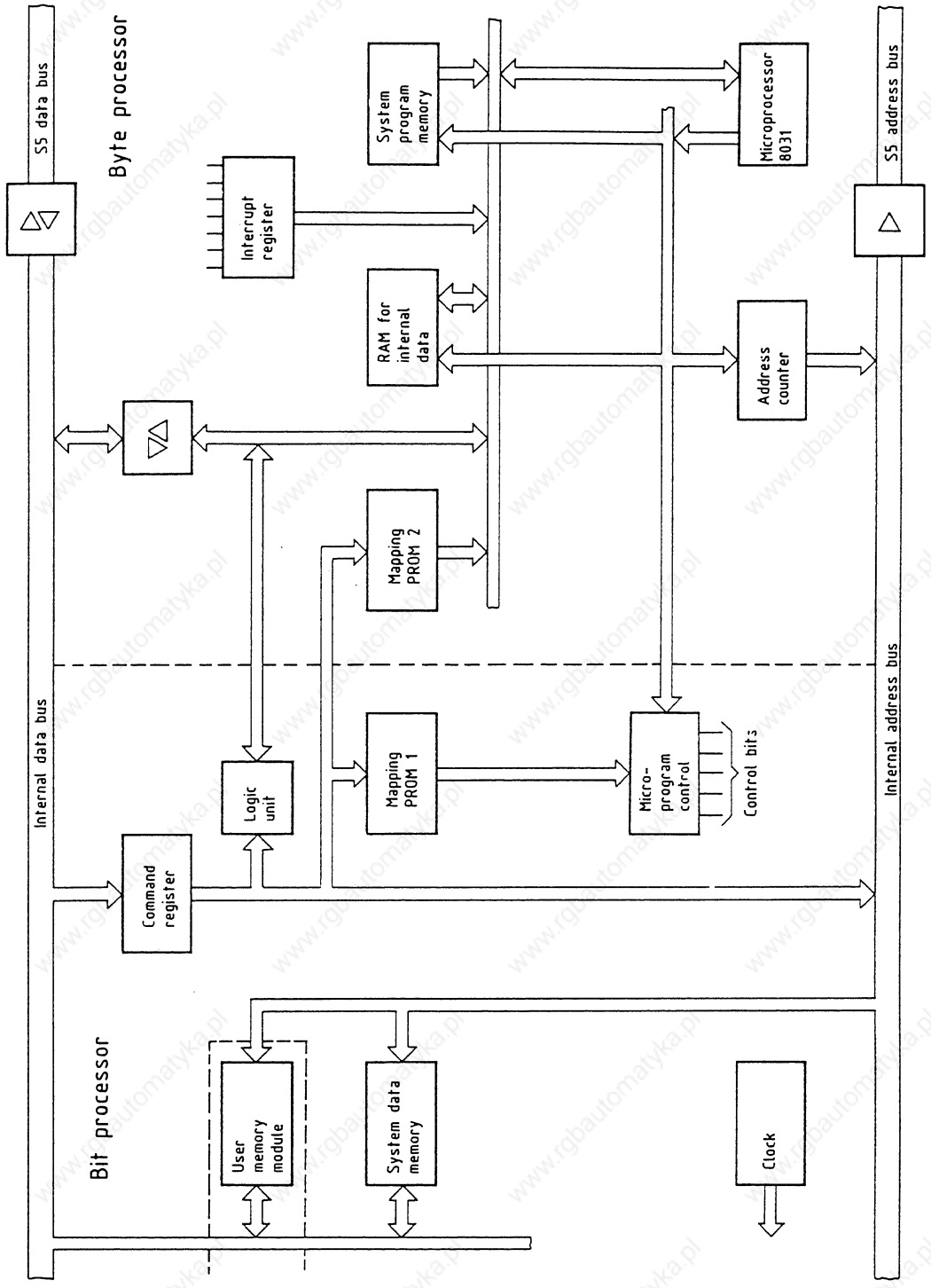


Fig. 1 Block diagram

1.3.2 User Memory Submodule

For the user memory submodule the user can plug in either an EPROM submodule (6ES5 376-...) or a RAM submodule (6ES5 377-...) in the S processor.

- EPROM submodule

The EPROM submodule is programmed off-line directly at the PG.

The following EPROM submodules are available:

Memory capacity	Order number
8 x 2^{10} words	6ES5 376-0AA11
16 x 2^{10} words	6ES5 376-0AA21
32 x 2^{10} words	6ES5 376-0AA31 1)

- RAM submodule

The user commands can be written into the RAM submodule on-line by the PG via the serial interface of the S processor. In order to retain the user data the RAM submodule is backed up by a battery in the housing of the S5 (S processor and RAM submodule must be plugged in).

The following RAM submodules are available:

Memory capacity	Order number
8 x 2^{10} words	6ES5 377-0AA11
16 x 2^{10} words	6ES5 377-0AB21
32 x 2^{10} words	6ES5 377-0AA32
32 x 2^{10} words	6ES5 377-0AA31 2)

1.3.3 Interrupt Processing

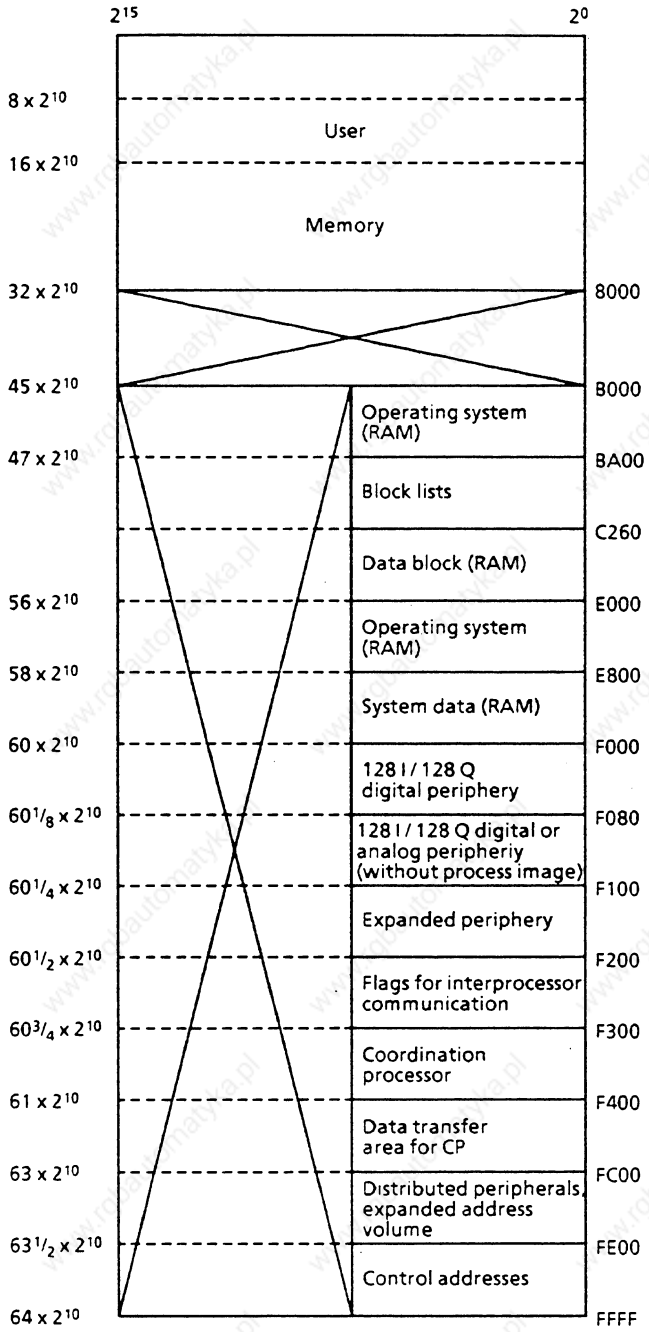
There is an interrupt line available for every CPU in the S5-135U. This can be used when a faster reaction to one or more events with higher priority is required than for other events. In order to process an interrupt the cyclic program execution is interrupted and the program which is stored in organization block OB 2 is inserted. For more details see the Programming Instructions.

The generation of an interrupt is only possible when a digital input module capable of interrupts (e.g. 6ES5 432-...) or an appropriate IP module is used.

1) Can only be programmed on the PG 675 or PG 685 if the MEP adaptor 6ES5 985-2AA11 is used. Besides, the S5-DOS operating system is required.

2) Long version, only for commissioning.

1.4 Memory Allocation



1.5 Technical Data

Degree of protection	IP 00
Permissible ambient temp.	0 to 55 °C (+32 to +131 °F)
Transport and storage temp.	-40 to 70 °C (-40 to +158 °F)
Relative humidity	max. 95 % at 25 °C (+77 °F), no condensation
Operating altitude	max. 3500 m above sea level
Power supply voltage	5 V ±5 %
Current consumption at 5 V	typ. 3 A
Back-up voltage	3.4 V
Back-up current without user memory RAM submodule	typ. 20 µA
Dimensions (W x H x D)	20.32 mm x 233.4 mm x 160 mm
Weight	approx. 0.5 kg

	P area	O area	Sum
Digital inputs with process image	max. 1024	-	max. 1024
Digital inputs without process image	max. 1024	max. 2048	max. 3072
or analog inputs	max. 64	max. 128	max. 192
Digital outputs with process image	max. 1024	-	max. 1024
Digital outputs without process image	max. 1024	max. 2048	max. 3072
or analog outputs	max. 64	max. 128	max. 192

Flags	2048
Timers	128
Counters	128
User memory capacity	max. 32×2^{10} words (16 bits wide), EPROM or RAM
Signalling rate of the serial PG interface	9600 bit/s
Program blocks (PB)	256
Sequence blocks (SB)	256
Function blocks (FB)	256
Data blocks (DB)	256
Organization blocks (OB)	1 to 39
Integrated special-function organization blocks (SF-OB)	40 to 255
Standard function blocks (SFB)	for digital functions (e.g. 32-bit binary divider, floating-point root extractor, shift register, buffer memory)

2 Installation

2.1 Removing and Inserting Modules

The modules are removed from the front of the central controller by gently rocking them up and down using the handles. Modules can only be removed or inserted when the central controller is switched off.

2.2 Slots in the Central Controller

Single processor operation:	the S processor must be plugged into slot 11 on the S5-135U.
Multiprocessor operation:	According to their number S, R and/or M processors must be plugged into slots 11, 27, 43 and 59 on the S5-135U. They must be in consecutive slots starting from slot 11.

3 Operation

3.1 Operating Mode Options

- Single processor operation

One S processor plugged in:

the S5 I/O bus and continuous cyclic operation permanently enabled.

- Multiprocessor operation

Two to four S, R and/or M processors are plugged in: the bus is enabled via the coordinator 923.

The IPC flag inputs and outputs and the I/O module inputs and outputs must be allocated for each CPU using the address list in DB 1. Without the address lists the whole PLC remains stopped. The address list in DB 1 is entered in the RAM submodule of the CPU on-line using the PG or directly in the EPROM submodule using the PG off-line (see Programming Instructions, Section 1.4.3).

Controlled by the coordinator 923, all the CPUs start the program synchronously.

If a CPU goes from cyclic program execution into the stop loop, all the other CPUs will also go into the stop loop and the digital outputs are disabled by the BASP signal. The test function is available for starting the CPUs asynchronously or independently using the coordinator 923 (see Programming Instructions, Section 5.2.3, and Operating Instructions for the coordinator 923 A or C).

A CPU intended for fast control (response time e.g. less than 1 ms) cannot at the same time exchange large amounts of data with the communications processors (CPs). It is then necessary to distribute different tasks among several CPUs.

3.2 Operator Controls and Displays

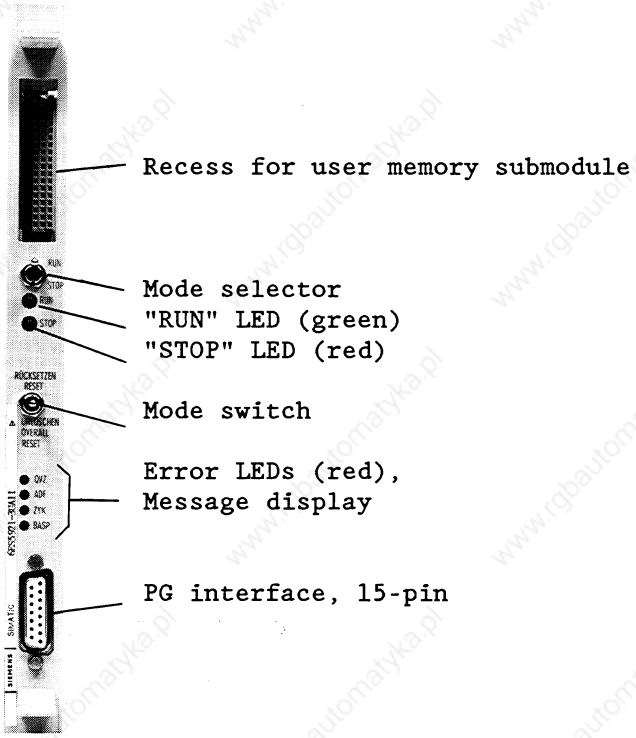


Fig. 2 Front panel of the S processor

Mode selector

- Switched to "RUN"

With the switch set to "RUN" the S processor is in cyclic operation, provided the green LED "RUN" is lit. Cyclic operation means:

The process image of the inputs, the execution of the user programs according to the call sequence in OB 1 (or FB 0 if no DB 1 has been loaded) the output of the process image of the outputs, the updating of IPC flags (corresponding to the programming in DB 1) and the triggering of the cycle monitoring are repeated in continuous cycles.

Cyclic processing is interrupted by the updating of the timer cells, by hardware interrupts and the processing of such interrupts and by the PG interface functions being called.

Cyclic processing is terminated if an error/fault is recognized or signalled in the system, in the device or in the program (e.g. power failure NAU, acknowledgement delay QVZ, substitution error SUF).

- Switched to "STOP"

After switching from "RUN" to "STOP" the CPU (or under certain circumstances the whole PLC) stops. The BASP signal is output and disables the digital output modules. The "force" function is possible from the PG since in this case the CPU suppresses the BASP signal and enables the digital outputs.

If the mode selector is set to "RUN" when the supply voltage is switched on (in multiprocessor operation this must apply to **all** S, R and /or M processors **and** the coordinator 923) then a retentive automatic cold restart will be carried out provided that the PLC was previously in cyclic operation and that neither the switch settings nor the device configuration were changed.

"RUN" LED

When the LED is lit continuously the cyclic execution of the program is running.

"STOP" LED

- LED lit continuously

After terminating the user program execution a stop will be caused:

a) in single processor operation by switching the mode selector to "STOP",
b) in multiprocessor operation as a) and also if the termination was caused by another CPU or the coordinator 923,
c) in multiprocessor operation by the PG function "PC STOP"[1)]
d) in single or multiprocessor operation by device faults, which cannot be attributed to a single CPU (NAU, BAU, PEU),
e) on completion of the PG function "PROG TEST" and after overall reset.

- LED flashing quickly

a) Prompts the user to perform an overall reset (see "Mode switch").
b) An error has been recognized during initialisation, e.g.: before switching on the back-up of the user memory RAM submodule or the CPU RAM was interrupted while the power supply was switched off; or the user memory RAM or the EPROM submodule is empty or has not been plugged in. An initial start without error handling is not possible. An overall reset must be carried out. Following this a non-retentive manual cold restart must be carried out.

1) PC = programmable controller (PLC)

- LED flashing slowly

a) The CPU caused an error during single or multiprocessor operation, which has led to a stoppage.
b) A CPU operator error has been made (selection of an illegal start-up mode, DB 1 error), this applies even when the mode selector is subsequently switched to "STOP". In this case the CPU was not yet in cyclic operation.
c) A stop command has been programmed into the start-up OB or in the cyclic user program.
d) The PG function "PC STOP" has been invoked in single processor operation.

Mode switch

- Overall reset

All the RAMs are erased and initialized. The "STOP" LED must first be lit continuously.

a) Hold the mode switch in the position "OVERALL RESET" and switch the mode selector from "STOP" to "RUN" and back to "STOP". Result: the "STOP" LED flashes quickly; an overall reset is required [1].
b) Hold the mode selector once again in the "OVERALL RESET" position and switch the mode selector from "STOP" to "RUN" and back to "STOP". Result: the overall reset is carried out, the CPU stops; the "STOP" LED is lit continuously. Following this, the only permissible start-up mode is a non-retentive manual cold restart.

- Reset

See Section 3.3

Error display and single LEDs

The errors and error reactions QVZ, ADF and ZYK are indicated by three LEDs. The fourth LED indicates the BASP signal.

"QVZ" LED

Lights up when the program addresses an I/O module which:

- (in single processor operation) has acknowledged during the cold restart of the S processor in the area of the process image (I 0 to 127, Q 0 to 127) and which has been entered in the so-called ninth rack as present, but no longer acknowledges, **or**
- (in single or multiprocessor operation) has been entered in DB 1 (address list) and was recognized as being present during the cold restart, but no longer acknowledges, **or**
- if an I/O module was addressed using direct access (commands L/T P..., L/T O...) but does not acknowledge or no longer acknowledges.

This can be caused by:

- Module failure.
- Removal of the module during operation or during the STOP state or while the PLC is switched off if no subsequent cold restart was carried out.

"ADF" LED

During the cold restart of the S processor the ninth track of the operating system is established. All the physically present I/O modules are marked in the area of the process image by a logical 1, all which are not present are marked by a logical 0 in the ninth track. In multiprocessor operation or when programming the address list in DB 1 the ninth track is established based on DB 1. If an I/O address is addressed by the user program and the corresponding module is not plugged in, the S processor interrupts the cyclic program processing.

"ZYK" LED

Lights up when the maximum cycle time has been exceeded. The cycle time comprises the sum of the run times of all parts of the user program (cyclic + timer-controlled + interrupt-driven). The error message "ZYK" interrupts the cyclic program processing.

"BASP" LED

Lights up when the command output is disabled. The digital outputs are switched directly into the safe status. With BASP the memories on the digital I/O modules are not reset. BASP is output when the power supply unit is switched on and off, when undervoltages occur and when the PLC is stopped.

3.3 Restart Modes

The data required from the operating system for cyclic operation are determined and set up. Following this the cyclic program processing begins. See also Programming Instructions.

1) Termination of the "overall reset" function: switch the mode selector from "STOP" to "RUN" and back to "STOP" without touching the mode switch. Result: overall reset is avoided: the CPU remains stopped.

3.3.1 Non-Retentive Manual Cold Restart

Flags, timers, counters and the process image are erased. The execution of the user program starts from the beginning.

The PLC must be stopped. In multiprocessor operation the switch on the coordinator 923 must be set to "STOP".

The CPU is reset and goes over to cyclic program execution, when

- the mode switch on the S processor is held in the "RESET" position,
- the mode selector is switched from "STOP" to "RUN" and
- in multiprocessor operation the switch on the coordinator 923 is switched from "STOP" to "RUN"
- or the PG function "PC START" is used.

3.3.2 Retentive Manual Cold Restart

Flags are retained; timers, counters and the process image are erased. The execution of the user program starts from the beginning.

Before it is stopped the PLC must have been in cyclic operation. In multiprocessor operation the switch on the coordinator 923 must be set to "STOP".

The CPU goes over to cyclic program processing, when

- the mode selector on the S processor is on the middle setting,
- the mode selector is switched from "STOP" to "RUN" or
- in multiprocessor operation the switch on the coordinator 923 is switched from "STOP" to "RUN"
- or the programmer function "PC START" is used.

3.3.3 Retentive Automatic Cold Restart

The same as retentive manual cold restart, however, after switching the power supply on or off no further action is required of the operator.

A retentive automatic cold restart is carried out after switching on the power supply, when

- the PLC was in cyclic operation before the power supply was switched on,
- the mode selector on the S processor (in multiprocessor operation on the S processor and on the coordinator 923) is set to "RUN" and has not been changed,
- the user memory submodule has not been unplugged and
- the back-up battery is functioning properly (the data in the RAM submodule must be retained).

4 Maintenance

4.1 Pin Assignments of the Backplane Connectors

Backplane connector 1

	d	b	z
2		M 5 V	+5 V
4	UBAT	PESP	
6	ADB 12	ADB 0	$\overline{\text{CPKL}}$
8	ADB 13	ADB 1	$\overline{\text{MEMR}}$
10	ADB 14	ADB 2	$\overline{\text{MEMW}}$
12	ADB 15	ADB 3	$\overline{\text{RDY}}$
14	$\overline{\text{IR}}$	ADB 4	DB 0
16		ADB 5	DB 1
18		ADB 6	DB 2
20		ADB 7	DB 3
22		ADB 8	DB 4
24		ADB 9	DB 5
26		ADB 10	DB 6
28	$\overline{\text{DSI}}$	ADB 11	DB 7
30	BUSEN	BASP	$\overline{\text{QUITT}}$
32	$\overline{\text{BASPA}}$	M 5 V	$\overline{\text{HALT}}$

Backplane connector 2

	d	b	z
2	T1	M 5 V	+5 V
4	T2	DB 8	DB 12
6	M 5 V	DB 9	DB 13
8	MA 0	DB 10	DB 14
10	MA 1	DB 11	DB 15
12	MA 2	MA 5	M 5 V
14	MA 3	MA 6	$\overline{\text{NAU}}$
16	MA 4	MA 7	$\overline{\text{BAU}}$
18		M 5 V	STAT
20		$\overline{\text{STEU}}$	
22	TXD [1]		$\overline{\text{PEU}}$
24	VKE	M 5 V	GEP
26	ZYK	RXD [1]	$\overline{\text{BE}}$
28		$\overline{\text{PERO}}$	
30	$\overline{\text{TE}}$	M 24 V	M 24 V
32		M 5 V	+24 V

4.2 Pin Assignments of the Front Connectors

1		9	RxD
2	RxD	10	24 V frame
3		11	20 mA/transmitter
4	+24 V from bus	12	
5		13	20 mA/receiver
6	TxD	14	
7	TxD	15	
8			

1) Applies only to S processor -3UA12

SIEMENS

SIMATIC S5

CPU 922R Processor 6ES5 922-3UA11

Instructions C79000-B8576-348-05

Contents **Page**

1	**Description**	**3**
1.1	Application	3
1.2	Design	3
1.3	Principle of Operation	3
1.3.1	Notes on the Block Diagram	3
1.3.2	User Memory Submodule	5
1.3.3	Interrupt Processing	5
1.4	Memory Map	6
1.5	Technical Data	7
2	**Installation**	**8**
2.1	Inserting and Removing Modules	8
2.2	Slots in the Central Controller	8
3	**Operation**	**8**
3.1	Possible Operating Modes	8
3.2	Operator Controls and Displays	9
3.3	Restart Modes	13
3.3.1	Cold Restart	13
3.3.2	Manual Warm Restart	13
3.3.3	Automatic Warm Restart	13
4	**Maintenance**	**14**
4.1	Pin Assignments of the Backplane Connectors	14
4.2	Pin Assignments of the Front Connectors	14

1 Description

1.1 Application

The R processor is used in the SIMATIC S5-135U programmable controller. The expansion capability of the CPU permits multiprocessor operation with up to four CPUs without rewiring.

The R processor is particularly suitable for fast byte processing (arithmetic, closed-loop control). Binary signal processing is also possible.

STEP 5 is the programming language used.

1.2 Design

The plug-in module is of double-height Eurocard format. Two 48-pin Range 2 plug connectors connect the module to the S5 bus in the rack.

The width of the front plate is 1 1/3 SPS (standard plug-in station). There is a rectangular receptacle on the front plate for a user memory submodule.

There is also a two-position switch and a button with 3 positions on the front plate. Operating states are indicated by means of a red and a green LED. Four small red LEDs display errors and reactions to them.

A 15-pin front connector is used to connect the R processor with programmers, diagnostic units or operator panels.

1.3 Principle of Operation

1.3.1 Notes on the Block Diagram

The R processor hardware structure is divided into three main function groups:

- Main CPU
- Quasi-dual-port RAM and bus link
- Interface CPU

o Main CPU

The heart of the main CPU is the **80186 16-bit microprocessor** with direct access to the following memories:

The **system program memory** contains the entire operating system and MC5 interpreter.

The **system data memory** contains data for the operating system, process image for digital inputs and outputs, flags (internal relays), timers and counters.

The **user memory submodule** contains the user program in MC5 code.

The **watchdog** monitors hardware for "Time-out" (QVZ), "Addressing error" (ADF), "Scan time exceeded" (ZYK) and clock failure in the 80186 microprocessor.

- Quasi-dual-port RAM

The quasi-dual-port RAM makes it possible to interface the 8031 and 80186 microprocessors.

- Interface CPU

The heart of the interface CPU is the **8031 microprocessor**. Its integral I/O components and serial interface make for optimum solutions in terms of space.

The **interface software memory** contains the operating system for interface functions.

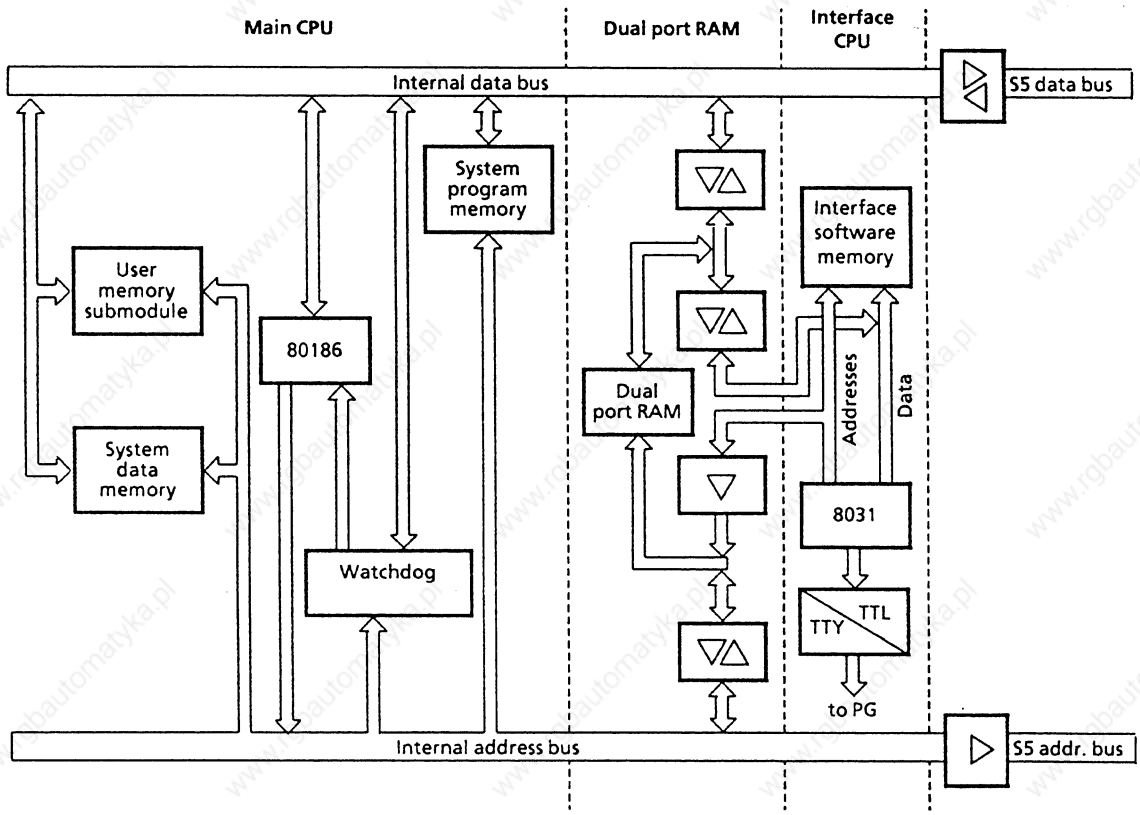


Fig. 1 Block diagram

1.3.2 User Memory Submodule

Either an EPROM submodule (6ES5 376-...) or RAM submodule (6ES5 377-...) can be plugged into the R processor as user memory submodule.

- EPROM submodule

The EPROM submodule is programmed off-line directly in the programmer. The following EPROMs are available:

Memory capacity	Order no.
$8x2^{10}$ words	6ES5 376-0AA11
$16x2^{10}$ words	6ES5 376-0AA21
$32x2^{10}$ words	6ES5 376-0AA31 [1]

- RAM submodule

The user operations can be written into the RAM submodule on-line from the programmer via the serial interface of the R processor. To retain user data, the RAM is backed up by a battery in the S5 (the R processor and RAM must be plugged in). The following RAMs are available:

Memory capacity	Order no.
$8x2^{10}$ words	6ES5 377-0AA11
$16x2^{10}$ words	6ES5 377-0AA21
$32x2^{10}$ words	6ES5 377-0AA32
$32x2^{10}$ words	6ES5 377-0BA31 [2]

1.3.3 Interrupt Processing

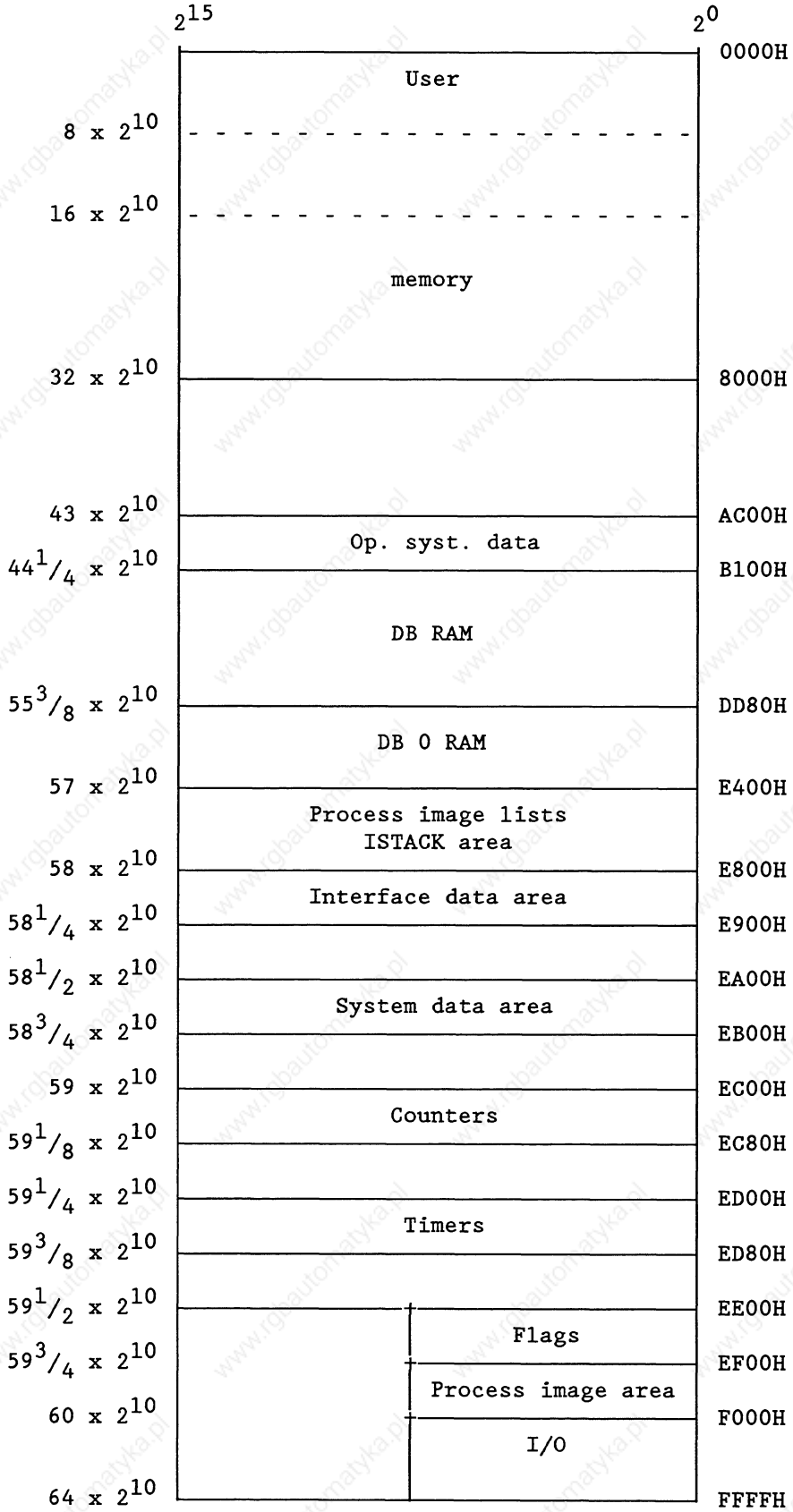
In the S5-135U programmable controller there is an interrupt line (IR) for each CPU. It can be used if the S5 must respond to one or more events faster and with a higher priority than to other events. Cyclic program processing is interrupted for processing an interrupt and the program stored in OB 2 is started.

Interrupt generation is only possible if a digital input module with interrupt capabilities (e.g. 6ES5 432-...) or an intelligent I/O module is used.

[1] Can only be programmed on the PG 675 or PG 685 if the MEP adapter 6ES5 985-2AA11 is used. Besides, the S5-DOS operating system is required.

[2] Long version, only for commissioning.

1.4 Memory Map



1.5 Technical Data

Degree of protection	IP 00 (no protection against water)		
Operating temperature	0 to 55°C (+32 to 131 °F)		
Transport and storage temperature	-40 to 70°C (-40 to +158 °F)		
Relative humidity	Max. 95% at 25°C, (+77 °F) no condensation		
Operating altitude	Max. 3500 m above sea level		
Supply voltage	5 V ± 5%		
Current consumption at 5 V	2.2 A (typical)		
Backup voltage	3.4 V		
Backup current without user RAM	Type 20 µA (typical)		
Dimensions (W x H x D)	20.32 mm x 233.4 mm x 160 mm (0.8 in. x 9.2 in. x 6.3 in.)		
Weight	Approx. 0.5 kg (1.1 lbs)		
	P area (I/O)	O area (extended I/O)	Total
Digital inputs			
with process image	max. 1024	-	max. 1024
without process image	max. 1024	max. 2048	max. 3072
or analog inputs	max. 64	max. 128	max. 192
Digital outputs			
with process image	max. 1024	-	max. 1024
without process image	max. 1024	max. 2048	max. 3072
or analog outputs	max. 64	max. 128	max. 192
Flags (internal relays)	2048		
Timers	128		
Counters	128		
User memory capacity	Max. 32K words (16 bits wide), EPROM or RAM		
Transmission speed of serial programmer interface	9600 bit/s		
Program blocks	256 PBs		
Sequence blocks	256 SBs		
Function blocks	256 FBs		
Data blocks	256 DBs		
Expanded function blocks	256 FXs		
Expanded data blocks	256 DXs		
Organization blocks	OB 1 to OB 39		
Special function organization blocks	SF-OB 40 to 255		
Standard function blocks (SFB)	8 compact loop controllers in 20 ms or a maximum of 64 compact loop controllers; shift registers		

2 Installation

2.1 Inserting and Removing Modules

The modules are removed from the central controller by pulling at the handles with a light rocking movement. The modules may only be inserted or removed from the central controller when it is switched off (does not apply to U range I/Os).

2.2 Slots in the Central Controller

Single processor operation:	The R processor must be inserted in the S5-135U in slot 11.
Multiprocessor operation:	According to their number the R, M and/or S processors must be inserted in the S5-135U in slots 11, 27, 43 and 59 without gaps, starting from location 11.

3 Operation

3.1 Possible Operating Modes

- Single processor mode

An R processor is inserted:
Permanent enabling of the S5 I/O bus and continuous cyclic operation.

- Multiprocessor mode

Two to four R, M and/or S processors are plugged in: The bus is enabled by the 923 coordinator. The assignment of interprocessor communication flag inputs and outputs and I/O module inputs and outputs must be defined in DB 1 as an address list for each CPU. Without address lists, the entire PLC remains in the stop state. The address list in DB 1 is entered in the CPU RAM on-line via the PG or direclty off-line with the PG in the EPROM (see Programming Instructions, Section 1.4.3).

All CPU programs are started synchronously when controlled by the 923 coordinator.

If a CPU goes into the stop state during cyclic program processing, all other CPUs also stop and the digital outputs are disabled with the BASP signal. For a synchronous or independent CPU start, the "Test" function can be selected via the 923 coordinator.

A CPU for rapid closed-loop control (sampling time 20 ms, for example) is unable to exchange quantities of data with communication processors simultaneously. It is then advisable to dedicate this CPU to its particular task and use further CPUs for communications.

3.2 Operator Controls and Displays

Recess for
user memory submodule

Mode selector

"RUN" LED (green)

"STOP" LED (red)

Operating mode switch

Error LEDs (red)
"QVZ"
"ADF"
"ZYK"

"BASP" LED (red)

PG interface, 15-pin

Fig. 2 Front panel of the R processor

Mode selector

"RUN" position

When the mode selector is set to "RUN" and the green "RUN" LED is simultaneously lit, the R processor is in cyclic mode. In cyclic mode, the process input image is read in repeatedly, the user program is processed in accordance with the call sequence in OB 1 of FB 0, the process output image is output, the interprocessor communication flags are updated (if programmed in DB 1) and scan time monitoring is triggered.

Cyclic processing is interrupted by the updating of timers, by hardware interrupts and their processing, by activated control loops and by any programmer interface functions called.

Cyclic processing is aborted if errors in the system, device and program, e.g. power supply failure (NAU), timeout (QVZ), substitution errors (SUF) are detected or reported.

"STOP" position

If the mode selector is switched from "RUN" to "STOP", the CPU (or the entire PLC) goes into the stop state. The BASP signal is output and causes the S5 digital output modules to be disabled. The "Force" function is possible via the programmer, as the CPU in this case suppresses the BASP signal and enables the digital outputs.

If the mode selector is set to "RUN" when the power supply is switched on (in multiprocessor operation for **all** R, M and/or S processors **and** the 923 coordinator), an automatic warm restart is executed if no errors (with the exception of "Power supply failure") have been recorded and the PLC was in cyclic mode before.

"RUN" LED

LED lit continuously:

- The program is being executed cyclically.

"STOP" LED

Stop status is indicated either by continuous illumination of the LED or by rapid or slow flashing.

LED lit continuously:

- After the power supply has been switched on if the mode selector is at "STOP" and no errors have occurred during initialization. A restart is possible.
- After cyclic user program processing has been aborted in multiprocessor mode, if the abort was caused by another CPU, the 923 coordinator or by switching the mode selector from "RUN" to "STOP".
- In the case of device faults which cannot be directly assigned to an individual CPU (e.g.: BAU, PEU).
- After an overall reset.

LED flashing rapidly:

- An overall reset has been requested (see "Operating mode button"). A restart is only possible if an overall reset has been executed or errors have been corrected with subsequent overall reset.

LED flashing slowly:

- An error has occurred in cyclic program processing by this CPU. The module is in the stop state if no relevant error handling has been programmed. By switching the mode selector from "RUN" to "STOP", the LED is lit again continuously.
- Operator error (e.g. selecting an illegal restart mode or DB 1 error).
- Stop command in user program.

Operating mode button

- "OVERALL RESET"

All RAMs are erased and initialized. Initial situation:
The "STOP" LED is lit continuously.

- Keep the operating mode button in, the "OVERALL RESET" position and switch the mode selector from "STOP" to "RUN" and back to "STOP".

  Result: The "STOP" LED flashed rapidly, overall reset is requested [1].

- Keep the operating mode button in the "OVERALL RESET" position once more and switch the mode selector from "STOP" to "RUN" and back to "STOP".

  Result: Overall reset is executed; the CPU remains in the stop state; the "STOP" LED is lit continuously. A warm restart can subsequently be executed.

- "RESET": See Section 3.3

[1] The overall reset can be aborted as follows at this point: Switch the mode selector from "STOP" to "RUN" and back to "STOP" **without** operating the operating mode button.

Result: An overall reset is avoided; the CPU remains in the stop state. The "STOP" LED is lit continuously.

LEDs for error display

"QVZ" LED is lit

- if in single processor mode on a cold restart of the R processor an I/O module addressed by the program has acknowledged in the process image area (I 0 to 127, Q 0 to 127) and has been entered as existing in track 9, if this I/O no longer acknowledges **or**
- if in multiprocessor or single processor mode an I/O addressed by the program has been entered in DB 1 (address list) and has been recognized as existing on a warm restart no longer acknowledges **or**
- if an I/O module addressed by the program with direct access (operations L/T P..., L/T Q...) does not acknowledge or no longer acknowledges.

Possible causes:

- Module failure.
- Removal of the module during operation or in the stop state or when the PLC is switched off without a subsequent cold restart. Reactions to QVZ can be programmed by the user in interface OBs.

"ADF" LED

On a cold restart of the R processor, the 9th track is created by the operating system. All I/O modules physically existing are marked in the 9th track with logical 1, all non-existant I/Os by logical 0. In multiprocessor mode or while programming the address list in DB 1, the 9th track is created on the basis of DB 1. If an I/O address is referenced by the user program and no module has been inserted at this address, the R processor interrupts cyclic program processing. The reaction to ADF can be programmed in the interface OBs.

"ZYK" LED

This LED lights up if the maximum scan time has been exceeded. The scan time is the sum of the run times of all user program sections (cyclic + timer-controlled + interrupt-driven). The "ZYK" error message interrupts cyclic program processing. The reaction to ZYK can be programmed by the user in interface OBs.

"BASP" LED

"BASP" lights up if command output has been disabled. The digital outputs are switched to a safe state. The BASP signal does not cause the memories on the digital I/Os to be reset. BASP is output when the power supply is switched on and off, if the voltage is too low and if the PLC is in the stop state.

3.3 Restart Modes

The data necessary (statuses etc.) from the operating system for cyclic operation are scanned in a start-up routine. Cyclic program processing then begins. The system program differentiates between three restart modes:

3.3.1 Cold Restart

Flags, timers, counters and the proces image are all reset. The user program is processed again from the beginning.

The PLC must be in the stop state. In multiprocessor mode, the switch on the 923 coordinator must be at "STOP".

The CPU is reset and returns to cyclic program processing, if

- the operating mode button on the R processor is held in the "RESET" position
- the mode selector is switched from "STOP" to "RUN" and, subsequently,
- the switch on the 923 coordinator is switched from "STOP" to "RUN" in multiprocessor mode.

3.3.2 Manual Warm Restart

The statuses of flags, timers, counters and the process image are retained during the down-time. The user program is resumed from the point of interruption.

The PLC must have been in cyclic mode before going into the stop state. In multiprocessor mode, the switch on the 924 coordinator must be at "STOP". The CPU returns to cyclic program processing if

- the operating mode button on the R processor is in the middle position,
- the mode selector is switched from "STOP" to "RUN" and/or, subsequently,
- the switch on the 923 coordinator is switched from "STOP" to "RUN" in multiprocessor mode.

3.3.3 Automatic Warm Restart

The statuses of flags, timers, counters and the process image are retained during the down-time. The user program is resumed from the point of interruption.

An automatic warm restart is executed after the power supply has been switched on if

- the PLC was in cyclic mode before "Power supply failure",
- the mode selector on the R processor (on the R, M and/or S processors and the 923 coordinators in multiprocessor mode) is in the "RUN" position,
- the user memory submodule has not been removed, and
- the backup battery is functioning correctly (data in RAM submodule must be retained).

Test operation: See Programming Instructions, C79000-B8576-C364.

4 Maintenance

4.1 Pin Assignments of the Backplane Connectors

Backplane connector 1

	d	b	z
2		M	+5 V
4	UBAT	PESP	
6	ADB 12	ADB 0	$\overline{\text{CPKL}}$
8	ADB 13	ADB 1	$\overline{\text{MEMR}}$
10	ADB 14	ADB 2	$\overline{\text{MEMW}}$
12	ADB 15	ADB 3	$\overline{\text{RDY}}$
14	$\overline{\text{IR}}$	ADB 4	DB0
16		ADB 5	DB1
18		ADB 6	DB2
20		ADB 7	DB3
22		ADB 8	DB4
24		ADB 9	DB5
26		ADB 10	DB6
28	$\overline{\text{DSI}}$	ADB 11	DB7
30	BUSEN	BASP	$\overline{\text{QUITT}}$
32	$\overline{\text{BASPA}}$	M	$\overline{\text{HALT}}$

Backplane connector 2

	d	b	z
2		M	+5 V
4			
6	M		
8			
10			
12			M
14			$\overline{\text{NAU}}$
16			$\overline{\text{BAU}}$
18		M	
20		$\overline{\text{STEU}}$	
22	TxD	STOPPA	$\overline{\text{PEU}}$
24		M	GEP
26	TEST	RxD	
28		$\overline{\text{PERO}}$	
30		M 24 V	M 24 V
32		M	+24 V

4.2 Pin Assignment of the Front Connector

1	Housing/GND/M_{ext}
2	RxD
3	VPG +5 V
4	+24 V from bus
5	GND/M_{int}
6	TxD
7	TxD
8	Housing/GND/M_{ext}
9	RxD
10	24 V GND
11	20 mA/transmitter
12	GND/M_{int}
13	20 mA/receiver
14	VPG +5 V
15	GND/M_{int}

SIEMENS

SIMATIC S5

376 EPROM Submodule
377 RAM Submodule

6ES5 376-0AA..
6ES5 377-0AA..
6ES5 377-0BA..

Instructions

C79000-B8576-C615-03

Page

1 EPROM Submodule 376 ... 3

2 RAM Submodule 377 (without battery back-up) ... 4

3 Buffered RAM Submodule 377 ... 5

3.1 Operating States ... 6

3.1.1 Normal Operation ... 6
3.1.2 Standby Mode ... 6
3.1.3 Autonomous Mode ... 7

3.2 Battery Monitoring and Battery Faults ... 7

3.3 Fitting or Removing the Back-up Battery ... 8

3.4 Using the RAM Submodule with Battery Back-up in a CPU ... 9

3.4.1 Starting Up ... 9
3.4.2 Important Notes ... 10

4 Technical Data of Memory Submodules ... 11

5 Pin Assignments of EPROM Submodule ... 13

6 Pin Assignments of RAM Submodule (with and without battery back-up) ... 14

Only the SIMATIC S5 memory submodules of short design with a storage capacity up to 64 x 2^{10} bytes can be used in the CPU 921, CPU 922, CPU 928 and CPU 928B.

Three different types of memory submodule are available:

- EPROM submodules
- RAM submodules (without battery back-up)
- RAM submodule with battery back-up.

Caution:

Switch the programmable controller off before inserting a memory submodule into the CPU or removing it from the CPU.

1 EPROM Submodule 376

EPROM submodules are programmed offline on the SIMATIC S5 programmer. The front of the PG has a special plug onto which the submodule is connected. Programming of the EPROM submodule is described in the manual of the respective programmer. Following programming, insert the submodule into the CPU; the programmable controller must be switched off (power supply OFF). The mode selector of the CPU should be switched to "STOP" at the same time.

EPROM submodules are available with the following capacities for the CPU mentioned above:

- 16 x 2^{10} bytes
- 32 x 2^{10} bytes
- 64 x 2^{10} bytes

The cover must be removed in order to erase the EPROM submodules.

You can find the order nos. of the EPROM submodules in the Appendix.

2 RAM Submodule 377 (without battery back-up)

RAM submodules are programmed online in the CPU. Loading of the individual blocks or of the complete program is described in the manual of the respective programmer.

You can load or modify the CPU both in the STOP and RUN modes. There are no limitations in the STOP mode. In the RUN mode, the mentioned conditions for loading online apply.

Note:

If the RAM submodule is full, DB/DX blocks are loaded into the DB RAM. Code blocks (OB, PB, SB, FB, FX) are rejected, however.

RAM submodules are available with the following capacities for the CPU 921, CPU 922, CPU 928 and CPU 928B:

- 16×2^{10} bytes
- 32×2^{10} bytes
- 64×2^{10} bytes

You can find the order nos. of the RAM submodules in the Appendix.

3 Buffered RAM Submodule 377

If you use a RAM submodule **with** battery back-up to store your user program, you can remove this from the CPU without losing the data. A battery protects the module from data loss and ensures that the data are retained until required again.

The RAM submodules with battery back-up can be used in the CPU 928B, CPU 928, CPU 922, CPU 920 and CPU 921.

A RAM submodule 377 with battery back-up and 64 x 2^{10} bytes is available.

Note:

The RAM submodule with battery back-up is not a replacement for an EPROM submodule!

To protect the battery, the RAM submodules have a cover on both sides.
The button cell battery with its terminal lugs is screwed to two holders on the RAM submodule.
The battery fault LED (see Chapter 4) can be seen when the handle is opened.

You can find the order nos. of the RAM submodule 377 in the Appendix.

3.1 Operating States

Three different operating states can be defined for the RAM submodule with battery back-up.

3.1.1 Normal Operation

In this operating state:

- The RAM submodule with battery back-up is plugged in the CPU
- The programmable controller (PLC) is switched on (power supply ON)
- Neither the back-up battery of the PLC nor the submodule battery of the RAM is supplying power.

Note:

Inserting or removing the RAM submodule in this operating state leads to corruption or loss of data and is therefore not permitted!

3.1.2 Standby Mode

In this operating state (unit standby mode):

- The RAM is plugged into the CPU
- The programmable controller (PLC) is switched on (power supply OFF)
- The back-up battery of the PLC is backing up the RAM submodule
- The submodule battery is not supplying power.

Note:

The RAM submodule can only be inserted or removed without corruption of data in this operating state!

If the central battery in the PLC fails in this state, the submodule battery takes over the back-up of the RAM submodule. This prevents any loss of data.

3.1.3 Autonomous Mode

In this operating state (submodule standby mode):

- The RAM submodule is removed from the CPU
- The submodule battery takes over the back-up of the RAM submodule
- The contents of the buffered RAM are retained.

3.2 Battery Monitoring and Battery Faults

Battery monitoring

The battery of the RAM submodule is monitered. If the RAM submodule changes over to the normal mode (buffered RAM inserted in CPU, power supply for PLC is switched on), the battery monitoring recognizes the following faults:

- The submodule battery is missing
- The submodule battery is faulty (voltage less than 2.6 V).

The red battery fault LED on the front of the submodule lights up permanently

Note:

A brief loss of voltage from the submodule battery is not detected by the battery monitoring in autonomous mode (e.g. caused by storage at temperatures below 0 °C or by replacing the battery) if the voltage is present again when the RAM submodule is inserted and the PLC is switched on. However, the loss of voltage may lead to loss or corruption of the data in the RAM submodule!

CPU responds to data errors

If the system program of the CPU detects incorrect data in the RAM submodule during the restart, the CPU stops and the LED flashes rapidly (= overall reset request). "MOD-FE" is marked in the ISTACK.

3.3 Fitting or Removing the Back-up Battery

Before using the RAM submodule for the first time, you must fit the accompanying battery. This is supplied separately to prevent it discharging. Proceed as follows:

- Open the upper part of the cover by releasing the catch: grip between the cover and the board and pull upwards.
- Insert the submodule battery and secure in place using screws on the left and right. Make sure the polarity is correct (+/-).
- Close the cover again.

Proceed in the same manner when replacing the battery at a later date:

- Open the top part of the cover by releasing the catch.
- Loosen the screws on the left and right of the battery.
- Replace the submodule battery and secure the new battery (make sure the polarity is correct).
- Close the cover again.

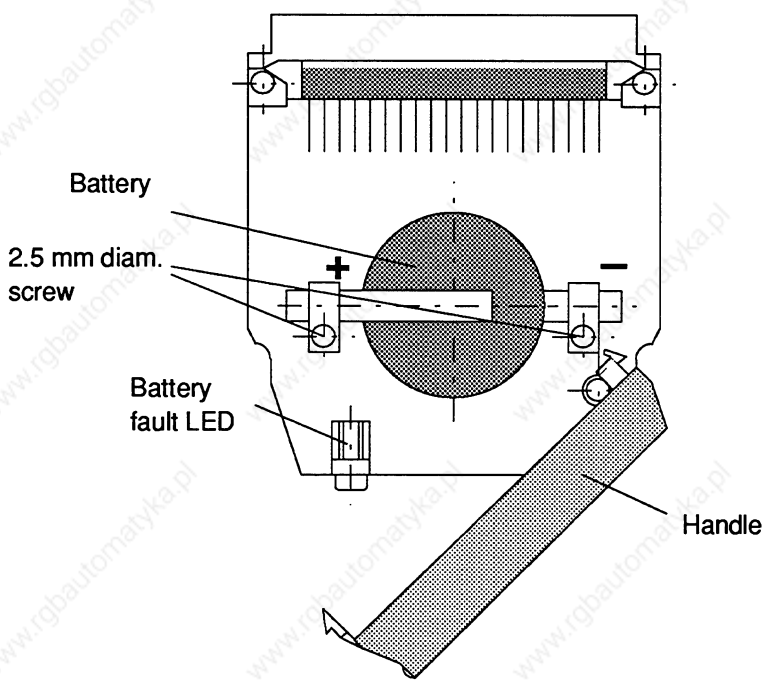


Fig. 1 Position of back-up battery

CAUTION: LITHIUM THIONYL CHLORIDE BATTERY!

Do not dispose of batteries in fire and do not solder on cell body - danger of explosion (max. temperature 100 °C) and do not attempt to recharge them. Do not open batteries. Only replace by batteries of the same type. Order replacement batteries only from Siemens using the order numbers listed in the Appendix. You can then be sure that you are using a short-circuit-proof battery.
Old batteries with some charge remaining should be discharged with a 10 Ω resistor or torch bulb until no further no-load voltage can be measured.
Completely discharged batteries no longer contain thionyl chloride and are therefore non-toxic and can be disposed of with normal garbage.
Charged lithium thionyl chloride batteries must otherwise be treated as toxic waste.

3.4 Using the RAM Submodule with Battery Back-up in a CPU

3.4.1 Starting Up

Before starting:
The CPU is plugged into the programmable controller.
The power supply for the PLC is switched off.
The mode selector on the CPU is set to "STOP".

- First fit the battery into the RAM submodule.
- Insert the RAM submodule into the CPU.
- Switch on the power supply to the programmable controller.
- Carry out an overall reset.
- Connect your programmer (PG) to the CPU.
- Once the user program has been loaded into the RAM submodule, carry out a cold restart on the CPU.

Note:

RAM submodules with battery back-up must not be programmed via the EPROM interface of the PLC as they could then be destroyed.

3.4.2 Important Notes

If you insert **programmed RAM submodules** into a CPU and the contents are to be retained (or if a CPU with an inserted submodule has been removed, or with a battery failure on the PLC), observe the following rules:

- You must carry out an overall reset of the CPU before inserting a programmed RAM submodule with battery back-up into the CPU (e.g. CPU 928, CPU 928B). (Carry out the overall reset with a different RAM submodule; each overall reset erases the contents of the inserted RAM submodule!)

- Before removing the RAM submodule, check that the module battery is still OK: if the battery fault LED lights up on the RAM submodule when the power supply to the PLC is switched on, the contents of the RAM submodule will be lost when it is removed.

- Switch off the power supply to the PLC before inserting or removing the RAM submodule into/from the CPU; only then can you be sure that the data in the RAM submodule are not corrupted.

4 Technical Data of Memory Submodules

a) For all memory submodules

Power supply	+5 V ± 5 %
Operating temperature	0 to 55 °C (+32 to +131 °F)
Storage temperature	0 to 60 °C (+32 to +140 °F)
Relative humidity	Up to 95 % at 25 °C (+77 °F), no condensation
Operating altitude	Max. 3500 m above sea level
Dimensions (h x d x w) Memory submodule 377 (64 x 2^{10} byte)	55 mm x 58 mm x 14 mm
Weight	Approx. 40 g

b) Additional data/only EPROM submodules

Current consumption (at 5 V)	Max. 200 mA
Access time t_{ACC}	250 ns

c) Additional data/only RAM submodules without battery back-up

Current consumption (at 5 V)	Max. 100 mA (16 x 2^{10} bytes/64 x 2^{10} bytes) Max. 200 mA (32 x 2^{10} bytes)
Back-up current/standby	Typically approx. 20 µA (16 x 2^{10} bytes/64 x 2^{10} bytes) Typically approx. 40 µA (32 x 2^{10} bytes)
Back-up voltage/UCMOS	2.7 to 3.6 V
Access time t_{ACC}	150 ns (16 x 2^{10} bytes/64 x 2^{10} bytes) 200 ns (32 x 2^{10} bytes)

d) Additional data/only RAM submodule with battery back-up

Current consumption (at 5 V)	Max. 140 mA
Back-up current	Typically 13 µA with 64 x 2^{10} byte
Back-up voltage/UCMOS	2.7 V to 3.6 V
Submodule battery	Lithium button cell 3 V/200 mAh
Back-up time	Min. 1 year at 25 °C (+77 °F)
Access time t_{ACC}	150 ns (16 x 2^{10} bytes/64 x 2^{10} bytes)

You can find the order nos. of the submodule battery in the Appendix.

5 Pin Assignments of EPROM Submodule

	c	b	a
1	SADB 12	M	+5 V
2	SADB 0	SADB 1	SADB 2
3	SADB 3	SADB 4	SADB 5
4	SADB 6	SADB 7	SADB 8
5	SADB 9	SADB 10	SADB 11
6	SADB 13	SADB 14	$\overline{R}$
7	$\overline{PGM}$	SDBH 0	SDBH 1
8	SDBH 2	SDBH 3	SDBH 4
9	SDBH 5	SDBH 6	SDBH 7
10	SDBL 0	SDBL 1	SDBL 2
11	SDBL 3	SDBL 4	SDBL 5
12	SDBL 6	SDBL 7	K 1
13	$\overline{CS\ 1}$	$\overline{CS\ 3}$	K 2
14	$\overline{CS\ 2}$	$\overline{CS\ 4}$	K 3
15	-	PSW	K 4
16	VPP	M	K 5

SADB 0 - 14 : Memory submodule address bus 0 - 14
SDBL 0 - 7 : Memory submodule data bus Low byte
SDBH 0 - 7 : Memory submodule data bus High byte
$\overline{R}$, $\overline{W}$: Read/write signals
$\overline{CS1}$, $\overline{CS2}$, $\overline{CS3}$: Chip select line for memory selection
K 1 - K 5, PSW : Module identifier
+5V : Fixed power supply
PGM : Programming pulse
VPP : Programming voltage
M : Ground

6 Pin Assignments of RAM Submodule (with and without battery back-up)

	c	b	a
1	SADB 12	M	+5 V
2	SADB 0	SADB 1	SADB 2
3	SADB 3	SADB 4	SADB 5
4	SADB 6	SADB 7	SADB 8
5	SADB 9	SADB 10	SADB 11
6	SADB 13	SADB 14	$\overline{R}$
7	$\overline{W}$	SDBH 0	SDBH 1
8	SDBH 2	SDBH 3	SDBH 4
9	SDBH 5	SDBH 6	SDBH 7
10	SDBL 0	SDBL 1	SDBL 2
11	SDBL 3	SDBL 4	SDBL 5
12	SDBL 6	SDBL 7	K 1
13	$\overline{CS\ 1}$	$\overline{CS\ 3}$	K 2
14	$\overline{CS\ 2}$	$\overline{STBY}$	K 3
15	UCMOS	PSW	K 4
16	-	M	K 5

SADB 0-14	:	Memory submodule address bus 0-14
SDBL 0-7	:	Memory submodule data bus Low byte
SDBH 0-7	:	Memory submodule data bus High byte
$\overline{R}$, $\overline{W}$	:	Read/write signal
$\overline{CS1}$, $\overline{CS2}$, $\overline{CS3}$	:	Chip select line for memory selection
K1-K5, PSW	:	Module identifier
$\overline{STBY}$	:	Switchover for standby operation
+ 5 V	:	Fixed power supply
UCMOS	:	Power supply for memory (+5 V or central battery voltage)
M	:	Ground

SIEMENS

SIMATIC S5

923A Coordinator 6ES5 923-3UA11

Instructions C79000-B8576-C263-08

Contents **Page**

1	**Technical Description**	**3**
1.1	Application	3
1.2	Design	3
1.3	Mode of Operation	3
1.3.1	Bus Arbitration	3
1.3.2	Mailbox	4
1.4	Technical Data	5
2	**Installation**	**5**
2.1	Removing and Inserting Modules	5
2.2	Slot in the S5-135U	5
3	**Operation**	**5**
3.1	Control Element	5
3.2	Operating Modes	5
3.3	Coding the Number of CPUs	7
3.4	Addressing the Mailbox	7
3.5	Jumper Assignment	8
4	**Connector Pin Assignment of the Backplane Connectors**	**9**
5	**Spare Parts List**	**9**

1 Technical Description

1.1 Application

The 923 A coordinator is installed in the S5-135U programmable controller. It is primarily intended to perform two independent tasks:

- Bus arbitration

The coordination of multiprocessing, i.e. the simultaneous use of two to four CPUs (S, R, M processors and/or CPU 928).

- Mailbox

For the exchange of data between CPUs.

1.2 Design

The 923 A coordinator (COR) is a plug-in PCB in double Euroformat.

Two 48-pin blade connectors of the "row 2" type connect the PC module to the S5 bus in the subrack.

The width of the front panel takes up 1 1/3 standard slots.

There is a toggle switch with three switch positions on the front panel for operator control functions.

1.3 Mode of Operation

1.3.1 Bus Arbitration

- Bus enable signals

The COR 923 A enables each of the two to four CPUs in the S5-135U to use the bus cyclically. A CPU can only use the common S5 bus during the time allocated to it.

The bus enables are allocated in a time-division multiplex operation. The number of CPUs can be adjusted with jumpers on the COR 923 A. The enable time for accessing the S5 bus is fixed at 2 µs for all the CPUs. The bus enable time can be extended with a bus lock.

The order of the bus allocations begins with CPU 1 after the reset signal has been cleared by the power supply and, depending on the set number, the CPUs are enabled in the following order:

CPUs 1, 2, 3, 4, 1, 2 etc.

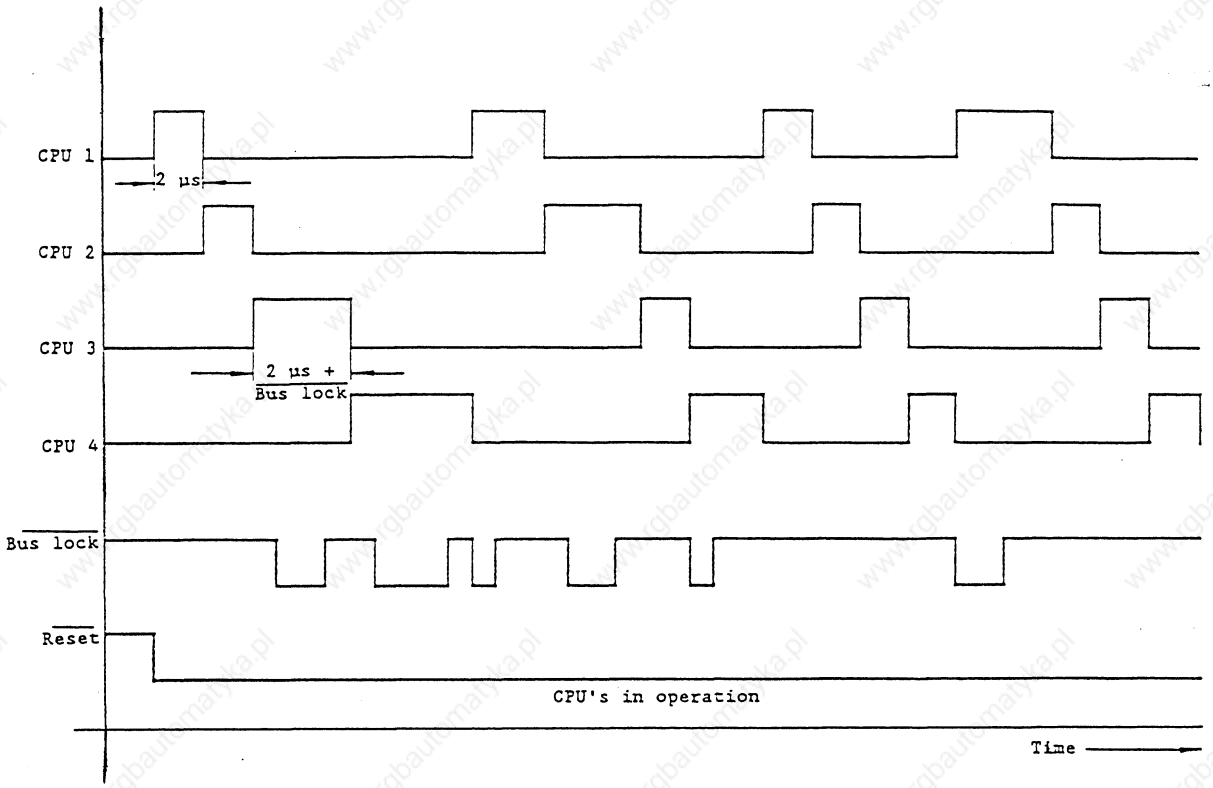


Fig. 1 Operational sequences of the bus control signals

1.3.2 Mailbox

A mailbox on the COR 923 A assumes the function of the interprocessor communication (IPC) flags. The IPC flags make possible the cyclic exchange of data between the CPUs and/or between the CPUs and the communications processors (CPs) in the S5-135U.

The mailbox consists of a RAM, buffered centrally via the S5 system.

The programming of this function is explained in the programming instructions of the processors.

1.4 Technical Data

Degree of protection	IP 00
Operational temperature	0 to 55 °C
Transport and storage temperature	-40 to 70 °C
Relative humidity	95 % at 25 °C no condensation
Operating altitude	max. 3500 m above sea level
Power supply voltage	5 V ±5 %
Current consumption at 5 V	typ. 0.5 A
Back-up battery voltage	2.7 V via accu. in power supply unit
Back-up current	typ. 100 nA
Dimensions (W x H x D)	20.32 mm x 233.4 mm x 160 mm
Weight	approx. 0.3 kg

2 Installation

2.1 Removing and Inserting Modules

The modules are removed from the front of the central controller by gently rocking them up and down using the handles. Modules can only be removed or inserted when the central controller is switched off.

2.2 Slot in the S5-135U

The COR 923 A is inserted in slot 3 in the S5-135U.

3 Operation

3.1 Control Element

The control element is a three level mode selector on the front panel with the switch positions "RUN", "STOP" and "TEST".

3.2 Operating Modes

- Stop status

If the mode selector is at "STOP" after the supply voltage has been switched on or if another stop request is present, the CPUs remain in the stop status.

- Cold restart

A cold restart is carried out when the position of the mode selector is changed from "STOP" to "RUN" providing the supply voltage has been switched on, and the CPUs have been reset.

- Cold restart with memory, warm restart

After the mode selector has been switched from "STOP" to "RUN", the CPUs can go into cyclic operation. This is only possible when the CPUs were in cyclic operation prior to this, the mode selectors on the CPUs are also switched to "RUN" and no other stop request is present.

- Automatic cold restart, automatic warm restart

When the mode selector is on "RUN" after the supply voltage has been switched on, an automatic cold restart with memory (S processor 3UA11 and 3UA12) or an automatic warm restart (R processor) then follows, provided that the mode selectors of the CPUs are also on "RUN" and the S5 system was previously in cyclic operation.

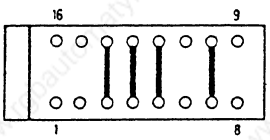
- Test operation

The test operation is used for commissioning the system for multi-processing. The test operation can be activated by inserting jumpers 3-14 in slot 45. When the position of the switch is changed from "STOP" to "TEST", this also leads to the cyclic status. The output of the BASP signal is suppressed for those CPUs which are held in the stop status owing to the front switch position. Some of the CPUs can therefore be put into operation, without the digital output modules being disabled by the BASP signal.

If an error occurs in a CPU which has been switched to "RUN" only this CPU will pass into the step status. Further, the BASP signal will be suppressed. In text operation, the error which has occured on this CPU does thus not affect other CPUs switched to "RUN".

| When the system start-up has been completed, the test operation must be made inactive in order to avoid operator errors occur ring.

Slot 45



Jumper 3-14 inserted = test function is set

Fig. 2 Setting the test function

3.3 Coding the Number of CPUs

The number of processors used for multiprocessing must be set by means of jumpers on the COR 923 A.

Number of CPUs	Jumper(s) on slot 62
2	7–10, 8–9
3	7–10
4	8–9

3.4 Addressing the Mailbox

The IPC flag area extends from F200H to F2FFH; this corresponds to 256 bytes. It can be set in 32-byte steps. Without CPs all 256 bytes are enabled for operation in the S5-135U and are available for IPC flag functions.

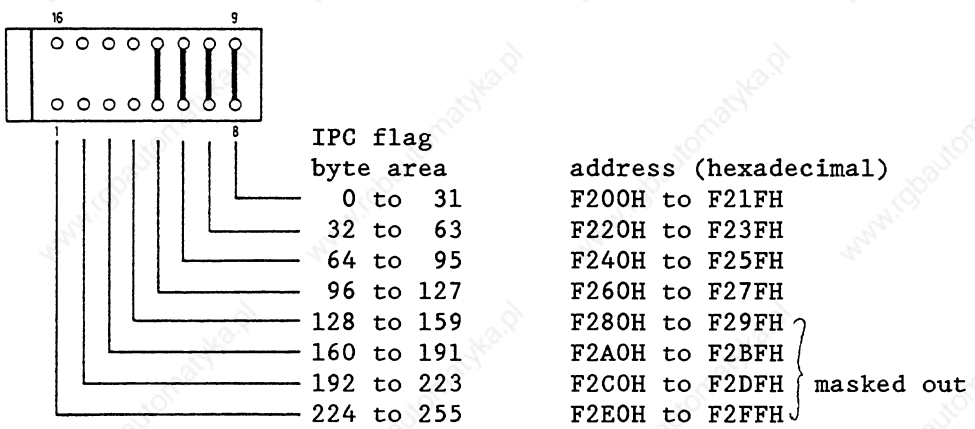
By removing jumpers on slot 7, one or several of the 32-byte areas can be masked out.

If IPC flag bytes are used on a CP, the corresponding areas must be masked out on the COR.

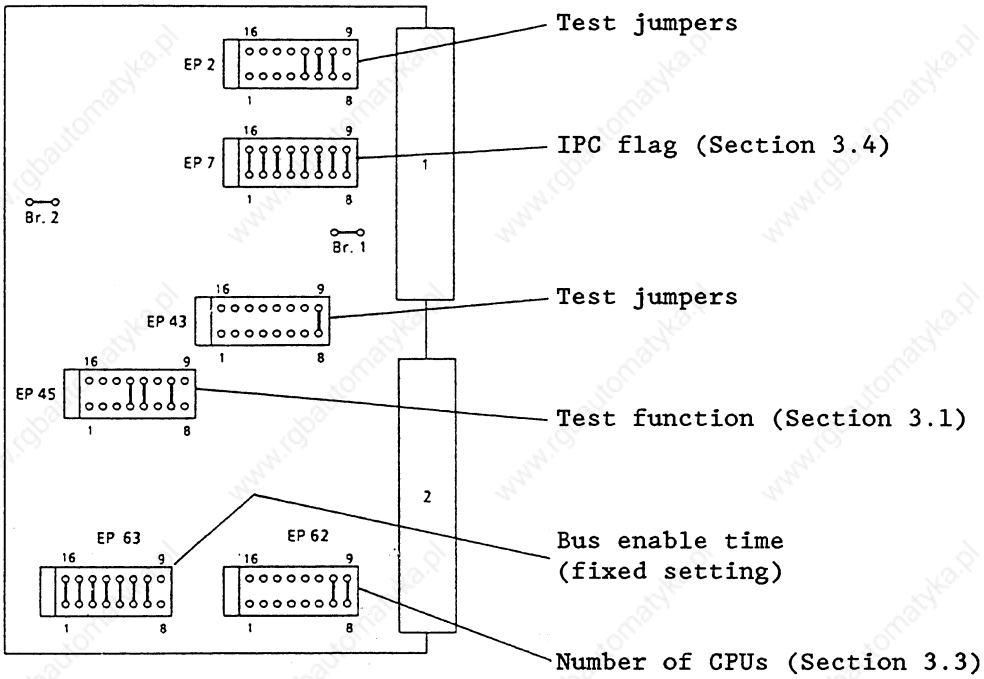
Example:

The four IPC flag areas with the highest addresses are to be masked out:

Slot 7



3.5 Jumper Assignment



Br = Jr
EP = slot

Fig. 3 Jumper locations and coding block
(illustrated as delivered)

4 Connector Pin Assignment of the Backplane Connectors

Backplane connector 1

	d	b	z
2		M 5 V	+5 V
4	UBAT		
6		ADB 0	CPKL
8		ADB 1	MEMR
10		ADB 2	MEMW
12		ADB 3	RDY
14	BUSEN 1	ADB 4	DB 0
16	BUSEN 2	ADB 5	DB 1
18	BUSEN 3	ADB 6	DB 2
20	BUSEN 4	ADB 7	DB 3
22		ADB 8	DB 4
24		ADB 9	DB 5
26		ADB 10	DB 6
28	DSI	ADB 11	DB 7
30			
32		M 5 V	HALT

Backplane connector 2

	d	b	z
2		M 5 V	+5 V
4			
6			
8			
10			
12			
14			NAU
16			
18			
20		STEU	
22		STOPPA	
24			
26	TEST		
28		PERO	
30			
32		M 5 V	

5 Spare Parts List

Coding plug C79334-A3011-B12

SIEMENS

SIMATIC S5

923C Coordinator 6ES5 923-4UC11

Instructions C79000-B8576-C049-01

Page

1 Technical Description of the 923C Coordinator ... 3

1.1 Application ... 3

1.2 Design ... 4

1.3 Mode of Operation ... 4

1.3.1 Bus Arbitration ... 4
1.3.2 Monitoring of Bus Occupation Time ... 5
1.3.3 Mailbox ... 6
1.3.4 PG Multiplexer ... 7

1.4 Technical Data ... 8

2 Installation of the 923C Coordinator ... 9

2.1 Removing and Inserting Modules ... 9

2.2 Slots in the Programmable Controllers ... 9

3 Operation of the 923C Coordinator ... 10

3.1 Display and Control Elements ... 10

3.2 Operating Modes ... 11

3.3 Setting the Coordination Unit ... 12

3.4 Setting the PG Multiplexer ... 12

3.5 Addressing the Mailbox ... 14

3.6 Jumpers for Disabling the Coordination Signals ... 16

3.7 Error Register ... 16

3.8 Locations of Switches and Jumpers ... 17

4 Connector Pin Assignments of the 923C Coordinator ... 18

1 Technical Description of the 923C Coordinator

1.1 Application

The 923C coordinator can be used in the S5-135U and S5-155U programmable controllers and in the EG-185U and EG-186U expansion units. It is primarily intended to perform three independent tasks:

- **Bus arbitration**

  To coordinate multiprocessor operation, i.e. the simultaneous use of two to four CPUs (CPU 946/947, CPU 928B, CPU 928, CPU 922 (R processor), CPU 920 (M processor), CPU 921 (S processor)).

- **Mailbox**

  For data transfer between CPUs via communication flags and data blocks.

- **Central programmer connection (PG-MUX)**

  To enable programming and commissioning of up to 8 modules via a PG connection.

  To enable programming of a programmable controller via the SINEC H1 or SINEC L1 bus, connect the programmer connections of the 923C coordinator using cable connector 725.

  The S5-DOS operating system must be present in the programmer to enable operation of the central PG interface.

1.2 Design

The 923C coordinator is a plug-in module in double Eurocard format.

Two 48-pin male connectors of "Series 2" design connect the module to the S5 bus in the subrack.

The front panel width is 1 1/3 standard slots.

There is a recess with a cover in the upper third of the front panel. DIL switches for the assignment of module parameters are accessible when this cover is removed.

There is a toggle switch with three positions on the front panel for other operator control functions.

Errors are indicated by five small red LEDs.

The 923C coordinator can be connected to a programmer, OP, the operator panel or the CP 530 or CP 143 by means of a 15-pin front connector.

1.3 Mode of Operation

1.3.1 Bus Arbitration

Bus enable signals:

The 923C coordinator enables each of the two to four CPUs in the S5-135U or S5-155U programmable controller to use the bus cyclically. A CPU can only use the common S5 bus during the time allocated to it.

The bus enables are assigned in time-division multiplex operation. The number of CPUs can be set on the 923C coordinator using DIL switches. The enable time for accessing the S5 bus is fixed at 2 µs for all CPUs. The bus enable time can be extended up to the end of the current read or write operation with a bus lock ($\overline{\text{PERO}}$) by means of the CPU currently using the bus.

The sequence of bus assignments commences with CPU 1 after the reset signal has been cleared by the power supply, and the CPUs are enabled in the following order depending on the set number:

CPU 1, CPU 2, CPU 3, CPU 4, CPU 1, CPU 2 etc.

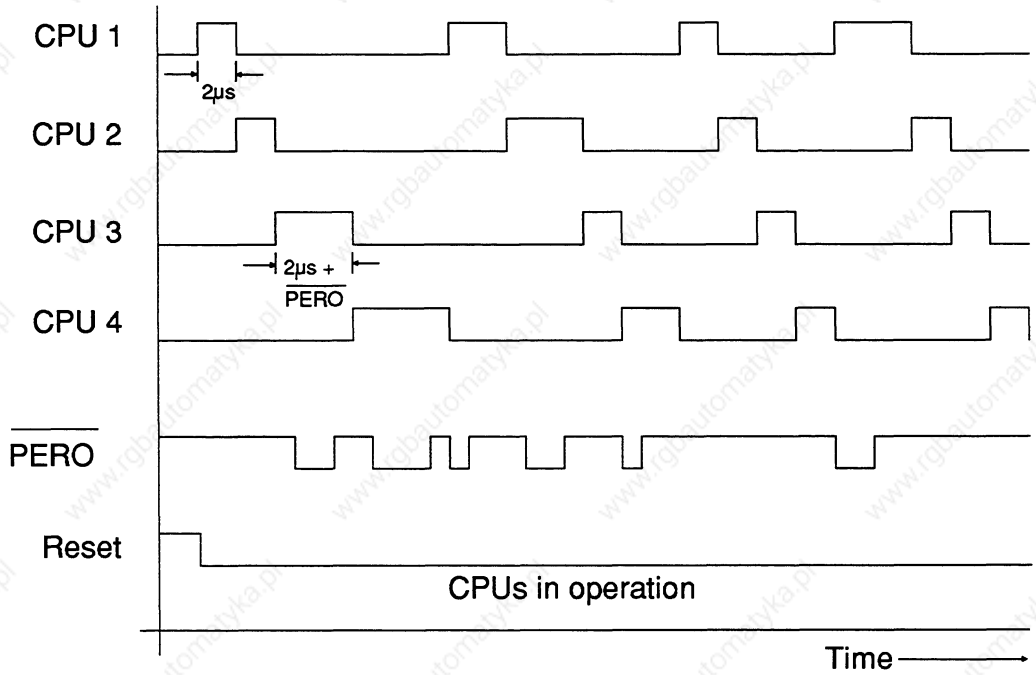


Fig. 1 Operational sequences of the bus control signals

1.3.2 Monitoring of Bus Occupation Time

The signal for bus lock can only be output by the CPU which has already received a bus enable from the 923C coordinator. The bus enable time for the CPU is extended by the duration of the bus lock signal (see Fig. 1). The factory setting of the bus lock signal is 2 ms. If the signal remains active for longer than this period, the 923C coordinator outputs a signal which stops all CPUs.

The CPU whose bus lock signal exceeded the maximum time is identified in a register which can be read by the programmable controller under the address FEFFH. The corresponding "BUS FAULT" LED on the front panel of the 923C coordinator lights up. The register and the LED are cleared again when the signal which caused the stop becomes inactive.

1.3.3 Mailbox

The mailbox consists of a RAM which is buffered centrally via the programmable controller. It has three areas, namely the interprocessor communication flags, the semaphores and a total of four page frames.

The interprocessor communication flags are in a memory area extending from F200H to F2FFH. The IPC flags enable cyclic data transfer between the CPUs in the S5-135U and S5-155U.

The four page frames are used to transfer data blocks between the CPUs.

The programming of these two functions is explained in the programming instructions of the CPUs.

The semaphores are used to coordinate the CPUs in the case of access operations to the same I/O address (see SED and SEE commands in the programming instructions)

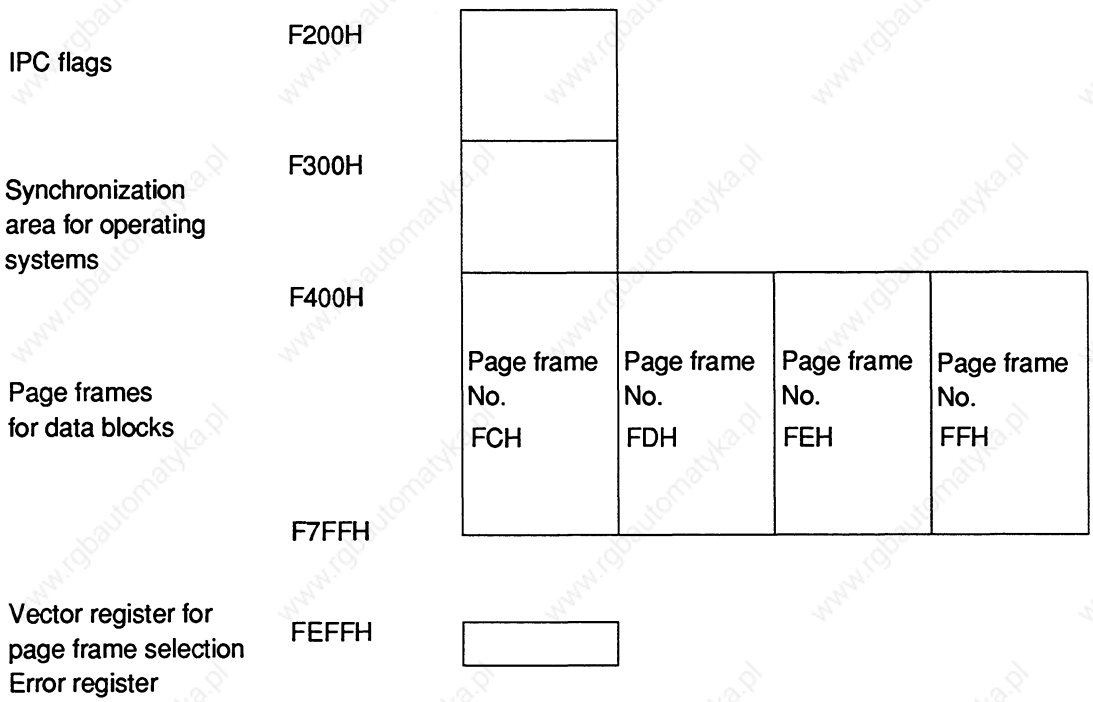


Fig. 2 Memory areas of the mailbox on the S5 bus

Addressing procedure for the page frame (vector register)

The vector register is used to create subaddresses of several memories in a common address area. The register is an 8-bit register which can be written into at address FEFFH. It cannot be read back.

There are four page frames with 1 x 2^{10} bytes each. Each page frame has an ID number allocated to it. These numbers are 0FFH, 0FEH, 0FDH, 0FCH. These numbers have a fixed setting on the 923C coordinator and cannot be changed.

These numbers must not be used on other modules (CP, IP) in the same programmable controller, otherwise double addressing will result.

The vector register is erased when the power supply is switched on. The vector register then contains the number 0H.

Data are transferred to and from this memory by special CPU functions. These functions are described in the relevant programming instructions. The CPU 921 does not possess these functions.

1.3.4 PG Multiplexer

The TTY interface of the 923C coordinator can be switched to eight different serial interfaces by selecting the path with the PG software.
These multiplex interfaces have TTL level and are wired to the other modules via backplane connector 2 and the bus PCB.

Procedure for selection of serial interfaces

Subscriber numbers are allocated to all modules in the programmable controller served by the multiplexer. These numbers must be between 1 and 31 (decimal). The lowest of these numbers, the base address, is set in binary code using the DIL switch S2. The maximum of eight numbers are allocated to the S5-135U slots 11, 27, 43, 59, 75, 83, 91 and 99 (the lowest number to slot 11) and to the S5-155U slots 11 or 27, 51 or 67, 91, 99, 107, 115, 123 and 131.

All eight numbers (or slots) are assigned to switch S3: the lowest number to switch S3.1, the highest number to switch S3.8. Selection of the subscriber numbers is described in Section 3.4 "Setting the PG Multiplexer".

If slots are not occupied, or if you wish to operate modules using their own front connectors, you must mask out the numbers allocated to the particular slots using switch S3.

The front connector of the PG interface of the CPU must remain unused in the case of a module serviced by the multiplexer. This only applies in the CPU 928B to the integrated PG interface SI1.

1.4 Technical Data

Degree of protection	IP 00
Operating temperature	0 to 55 °C (+32 to +131 °F)
Transport and storage temperature	-40 to +70 °C (-40 to +158 °F)
Relative humidity	Max. 95 % at 25 °C (+77 °F) no condensation
Operating altitude	Max. 3500 m above sea level
Power supply	5 V ± 5 % 24 V +25 %/-15 %
Current consumption at 5 V	Typically 1.1 A
Current consumption at 24 V	60 mA
Minimum back-up voltage	2.7 V
Back-up current	Typically 2 µA
Acknowledgement time for access to mailbox via S5 bus	Typically 320 ns
Transmission rate of serial interface	9600 baud
Transmission cable	Screened 4-wire line, PG connection cable
Transmission distance	Max. 1 km at 9600 baud
Weight	Approx. 0.3 kg
Dimensions (w x h x d)	20.32 mm x 233.4 mm x 160 mm

2 Installation of the 923C Coordinator

2.1 Removing and Inserting Modules

Modules must only be removed or inserted when the power supply is switched off. Remove the modules from the front of the central controller by gently rocking them up and down using the handles.

2.2 Slots in the Programmable Controllers

- Multiprocessor operation and PG-MUX:

  In the S5-135U, slot 3

  In the S5-155U, slot 3

- Only as PG-MUX:

  In the EG-185U expansion unit, slot 11

  In the EG-186U expansion unit, slot 19

3 Operation of the 923C Coordinator

3.1 Display and Control Elements

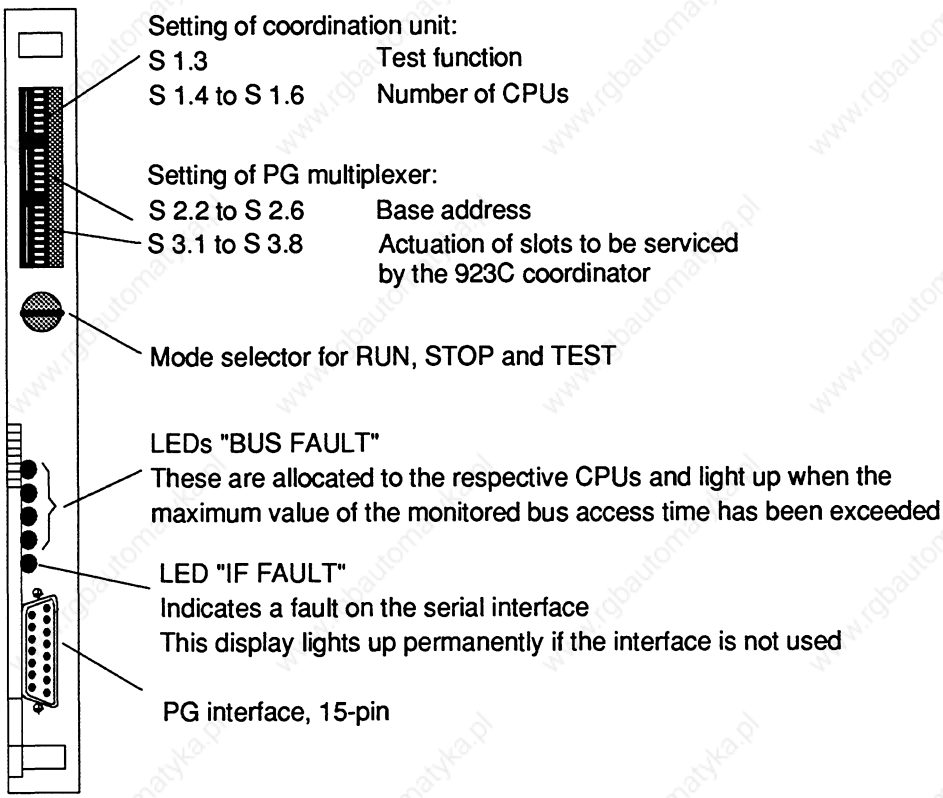


Fig. 3 Front panel of the 923C coordinator

3.2 Operating Modes

- **Stop mode**

  The CPUs remain in the stop mode if the mode selector is at "STOP" after the power supply has been switched on or if another stop request is present.

- **Automatic cold restart, automatic warm restart and RUN**

  If the mode selector is at "RUN" after the power supply has been switched on, an automatic retentive cold restart (CPU 921) or an automatic warm restart (CPU 946/947, CPU 928B, CPU 928 and CPU 922) is carried out provided the mode selectors of the CPUs are also at "RUN" and the programmable controller was previously in cyclic operation.

- **Test operation**

  Test operation is used to commission the system with multiprocessor operation and can be activated by means of the DIL switch S1.3. When the position of the switch is changed from "STOP" to "TEST", this also leads to the cyclic status. **Output of the BASP signal is suppressed** for those CPUs which are held in the stop mode as a result of the front switch position. Some of the CPUs can therefore be put into operation without the digital output modules being disabled by the BASP signal.
  If an error occurs in a CPU which has been switched to "RUN", only this CPU is set to the stop status; the BASP signal is suppressed in addition. In test operation, the error on this CPU does not affect other CPUs switched to "RUN".

Caution:

It is essential to deactivate test operation once commissioning is completed to prevent critical system conditions, i.e. the DIL switch S1.3 must be set to "OFF".

3.3 Setting the Coordination Unit

The three DIL switches are used to set the number of CPUs installed in the programmable controller. Only one switch must be set to "On". Three slots must be enabled when installing two CPU 928Bs and/or CPU 928s in the S5-135U; i.e. the number of CPUs must be set to 3.

"Number of CPUs = 2" is the factory setting (see below).

Factory setting:

	On	Off	
S1.1		x	—
2		x	—
3		x	Test operation (see also Section 3.2)
4	x		Number of CPUs = 2
5		x	Number of CPUs = 3
6		x	Number of CPUs = 4

3.4 Setting the PG Multiplexer

Base address

Set a base address between 1 and 31 using the DIL switch S2. The modules selected by the multiplexer can be addressed at this address and the following seven addresses. The base address is the sum of the binary values activated by setting the switch position "On".

Factory setting:

	Off "0"	On "1"	
S2.1	x		—
2	x		Value 16
3	x		Value 8
4	x		Value 4
5	x		Value 2
6		x	Value 1 (base address = 1)

Activation of addresses

The numbers or slots which are to be serviced from the 923C coordinator must be activated using switch S3.

Factory setting:

	Off	On		Slot No. in S5-135U	Slot No. in S5-155U
S3.1	x		Base address + 0	11	11 or 27
2	x		Base address + 1	27	51 or 67
3	x		Base address + 2	43	91
4	x		Base address + 3	59	99
5	x		Base address + 4	75	107
6	x		Base address + 5	83	115
7	x		Base address + 6	91	123
8	x		Base address + 7	99	131

Example:

You wish to address the modules in slots 11, 59, 75 and 99 in the S5-135U or in slots 11, 99, 107 and 131 in the S5-155U by means of the 923C coordinator using base address 10.

Setting the base address:

	Off	On			
S2.1	x		—		
2	x		Value 16		
3		x	Value 8	→	8
4	x		Value 4		
5		x	Value 2	→	+ 2
6	x		Value 1		10 (base address)

Activation of the required slots for the S5-135U:

	Off	On		Slot No. (in the S5-135U)	Slots to be serviced	Address
S3.1		x	Base address + 0	11	11	10
2	x		Base address + 1	27		
3	x		Base address + 2	43		
4		x	Base address + 3	59	59	13
5		x	Base address + 4	75	75	14
6	x		Base address + 5	83		
7	x		Base address + 6	91		
8		x	Base address + 7	99	99	17

Activation of the required slots for the S5-155U:

	Off	On		Slot No. (in the S5-155U)	Slots to be serviced	Address
S3.1		x	Base address + 0	11	11	10
2	x		Base address + 1	51		
3	x		Base address + 2	91		
4		x	Base address + 3	99	99	13
5		x	Base address + 4	107	107	14
6	x		Base address + 5	115		
7	x		Base address + 6	123		
8		x	Base address + 7	131	131	17

3.5 Addressing the Mailbox

The interprocessor communication flag area extends from F200H to F2FFH; this corresponds to 256 bytes. It can be set in 32-byte steps. Without CPs, all 256 bytes are enabled for operation in the S5-135U or S5-155U programmable controllers and are available for IPC flag functions.

One or several of the 32-byte areas can be masked out by removing jumpers at location 60.

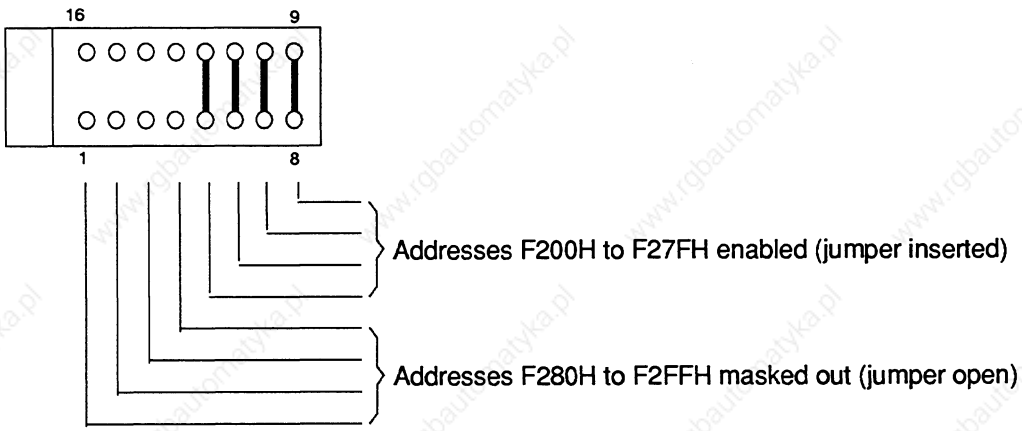
If interprocessor communication flag bytes are used on a CP, the corresponding areas must be masked out on the 923C coordinator.

Jumper at location 60	IPC flag byte area	Address (hexadecimal)
1 - 16	224 to 335	F2E0H to F2FFH
2 - 15	192 to 223	F2C0H to F2DFH
3 - 14	160 to 191	F2A0H to F2BFH
4 - 13	128 to 159	F280H to F29FH
5 - 12	96 to 127	F260H to F27FH
6 - 11	64 to 95	F240H to F25FH
7 - 10	32 to 63	F220H to F23FH
8 - 9	0 to 31	F200H to F21FH

Example:

The four IPC flag areas with the highest addresses are to be masked out:

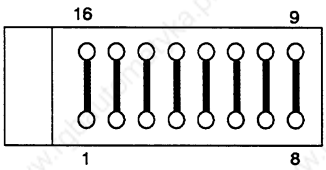
Location 60



3.6 Jumpers for Disabling the Coordination Signals

All output signals required for coordination can be interrupted using a jumper plug. This is necessary if the 923C coordinator is used in units other than the S5-135U and S5-155U.
You can then use the 923C coordinator as a PG-MUX in the EG-185U and EG-186U expansion units. The coordination signals must be disabled in this mode of operation.

Location 61

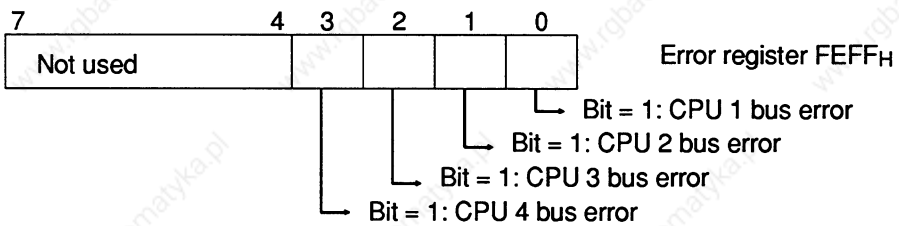


Coordination	
In operation	Disabled
All jumpers inserted	All jumpers removed

All jumpers are inserted in the factory setting.

3.7 Error Register

The error register is an 8-bit register and can be read on the CPU side at address FEFFH. An entry is made in the register by the bus monitoring function if a bus error occurs. One bit in the error register is allocated to each CPU, and this is set to "1" if an error occurs. The register is erased whenever the halt signal becomes inactive.



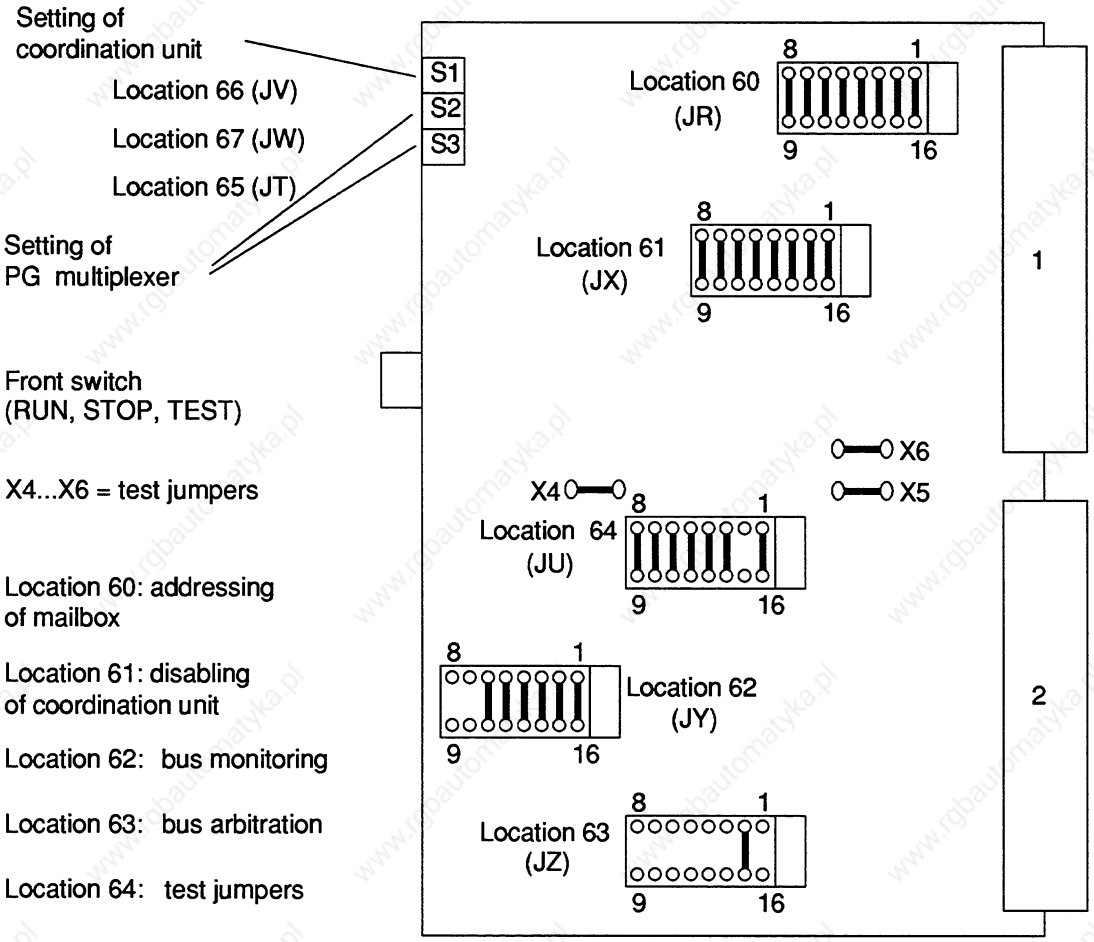
The error register can be read by all CPUs so that central functions can be initiated.

Note:

The error register and the page register are present at address 0FEFFH. When writing at 0FEFFH, the page register is written, the error register is read when reading this address.

3.8 Locations of Switches and Jumpers

The jumper settings shown correspond to the factory settings.



The jumpers X4 to X6 and locations 62 to 64 must not be changed.

4 Connector Pin Assignments of the 923C Coordinator

Backplane connector 1

	d	b	z
2		M 5 V	+ 5 V
4	UBAT		
6	ADB 12	ADB 0	RESET
8	ADB 13	ADB 1	MEMR
10	ADB 14	ADB 2	MEMW
12	ADB 15	ADB 3	RDY
14	BUSEN 1	ADB 4	DB 0
16	BUSEN 2	ADB 5	DB 1
18	BUSEN 3	ADB 6	DB 2
20	BUSEN 4	ADB 7	DB 3
22		ADB 8	DB 4
24		ADB 9	DB 5
26		ADB 10	DB 6
28	DSI	ADB 11	DB 7
30			
32		M 5 V	HALT

Backplane connector 2

	d	b	z
2		M 5 V	+ 5 V
4			
6	RXD 8		
8	TXD 8		
10	RXD 7		
12	TXD 7	RXD 6	
14	RXD 5	TXD 6	NAU
16	TXD 5	RXD 4	
18	RXD 3	TXD 4	
20	TXD 3	STEU	
22	RXD 1	STOPPA	
24	TXD 1	RXD 2	
26	TEST	TXD 2	
28		PERO	
30			M 24 V
32		M 5 V	+ 24 V

Front connector

Pin	Designation
1	Housing/ground/M_{ext}
2	Receiver TTY (-)
3	Private line
4	+24 V
5	Private line
6	Transmitter TTY (+)
7	Transmitter TTY (-)
8	Housing/ground/M_{ext}
9	Receiver TTY (+)
10	24-V ground (current sources (-) 20 mA)
11	Current source (+) 20 mA
12	Private line
13	Current source (+) 20 mA
14	Private line
15	Private line

SIEMENS

SIMATIC S5

Multiprocessor Operation

Instructions C79000-B8576-C048-01

Page

1 Introduction to Multiprocessor Operation 3

2 Starting Up in Multiprocessor Operation 5

2.1 Requirements 5

2.2 Procedure 5

3 Operating Modes of the Coordinator 14

3.1 Normal Operation and Stop on Errors 14

3.2 Test Operation 14

4 The PG Multiplexer on the 923C Coordinator 15

1 Introduction to Multiprocessor Operation

The S5-135U and S5-155U belong to the family of SIMATIC S5 programmable controllers. The controllers can be used in single processor and multiprocessor operation with up to four CPUs. The following CPUs are available:

For the S5-135U:

- **CPU 928B:**

  For very fast word and binary signal processing and for communication. Programming in STEP 5.

- **CPU 928:**

  For fast word and binary signal processing. Programming in STEP 5.

- **CPU 922 (R processor):**

  Ideal for word processing (computing, closed-loop control, monitoring, signalling). Programming in STEP 5.

- **CPU 920 (M processor):**

  For programming with higher programming languages (BASIC, C) and assembler; for arithmetic, sorting and statistics related to S5 system functions (cold restart, warm restart, communication, process image).

- **CPU 921 (S processor):**

  Suitable for binary signal processing (open-loop control tasks). Programming in STEP 5.

For the S5-155U:

- **CPU 946/947:**

  For very fast word and binary signal processing, especially fast double-word and floating-point processing, as well as for extensive programs that have large memory requirements. Programming in STEP 5.

- You can also use the CPU 928B, CPU 928, CPU 922 (R processor) and CPU 920 (M processor) in the S5-155U.

You can plug the CPUs into the respective central controller in any combination as long as the CPU slots are wide enough and the S5 bus assignment is appropriate for the CPU. The slot requirements are as follows:

- CPU 946/947 requires up to 5 slots (3 for CPU 946/947 and 1 or 2 for the type 355 memory modules). Only two locations in the S5-155U central controller housing can accommodate this CPU.
- CPU 928B and CPU 928 each require two slots.
- CPU 922, CPU 920 and CPU 921 each require one slot.

In multiprocessor operation, every CPU processes its individual user program independent of the other CPUs (multi-computing).
Data transfer with I/O modules, CPs, IPs and other CPUs is via the common S5 bus. In multiprocessor operation, a coordinator controls CPU access to the S5 bus. The instructions for the 923C and 923A coordinators explain how bus allocation works.

Data transfer between the CPUs in multiprocessor operation can take place by means of:

- **"Interprocessor communication (IPC) flags":**

  For cyclic transfer of binary data (see Programming Guide)

and/or

- **"Special functions for multiprocessor communication":**

  For program-controlled transfer of entire data blocks; only possible with 923C coordinator and not with CPU 921 (S processor); see the instructions in "Multiprocessor Communication" in this manual

The following describes the basic procedure for commissioning the programmable controller in multiprocessor operation.

2 Starting Up in Multiprocessor Operation

This section describes starting up the S5-135U and S5-155U programmable controllers in multiprocessor operation. The reactions of the CPUs to the individual operating steps are also listed.

2.1 Requirements

The following explanation assumes an understanding of operating and programming the individual modules in single processor operation. For further information, refer to the instructions in the manuals of the S5-135U and S5-155U.
Debugged programs and fully functional CPUs are further requirements.

2.2 Procedure

You can use up to four CPUs in the S5-135U and S5-155U. The instructions for the two central controllers list the permissible slots.

A coordinator module is always required in multiprocessor operation. Either a 923A or 923C coordinator can be used for the S5-135U, only the 923C coordinator can be used for the S5-155U.

The coordinator allocates time slices to the CPUs, during which they can access the S5 bus (bus enable time). The coordinator contains the global memory for data transfer between the CPUs via IPC flags. The 923C coordinator contains an additional memory with four page frames for the multiprocessor communication function as well as the multiplexer for the serial PG interface (PG-MUX). **The fixed bus enable time must not be changed in the 923C or 923A coordinator!**

Note:

All CPUs are automatically in multiprocessor operation as soon as a coordinator is plugged into the S5-135U or S5-155U central controller. Even if you operate the coordinator with one CPU, multiprocessing conditions apply to **this** CPU (e.g. DB 1 is necessary, DX 0 may be required, CPU 946/947 can only operate in 155U mode etc.).

Figs. 1 and 2 show the positions of the coding bases on the modules on which the settings required for installation must be made.

Location 2: test jumpers

Location 7: masking of IPC flag area

Location 43: test jumpers

Location 45: test function

Location 62: number of occupied CPU slots

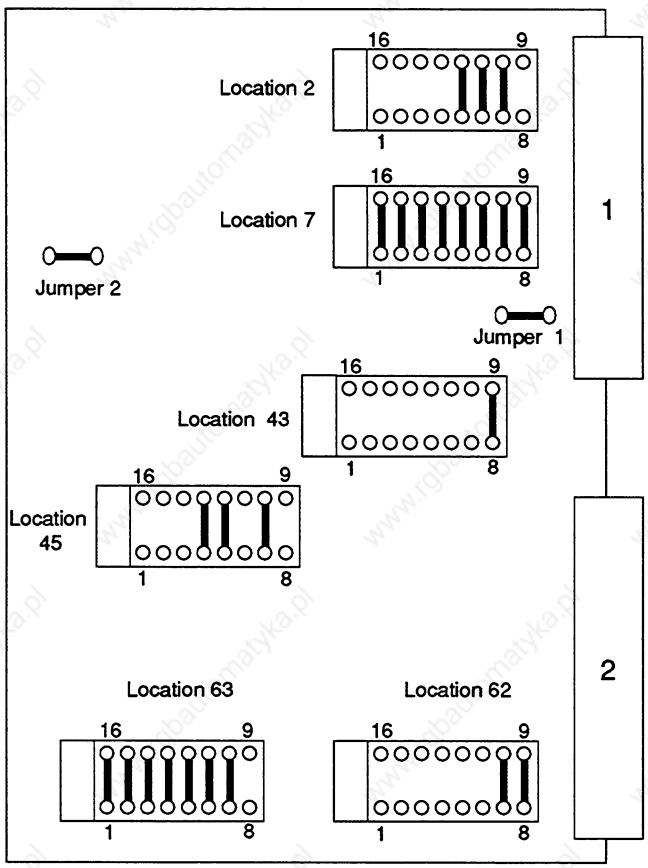


Fig. 1 Position of coding bases on 923A coordinator (factory settings)

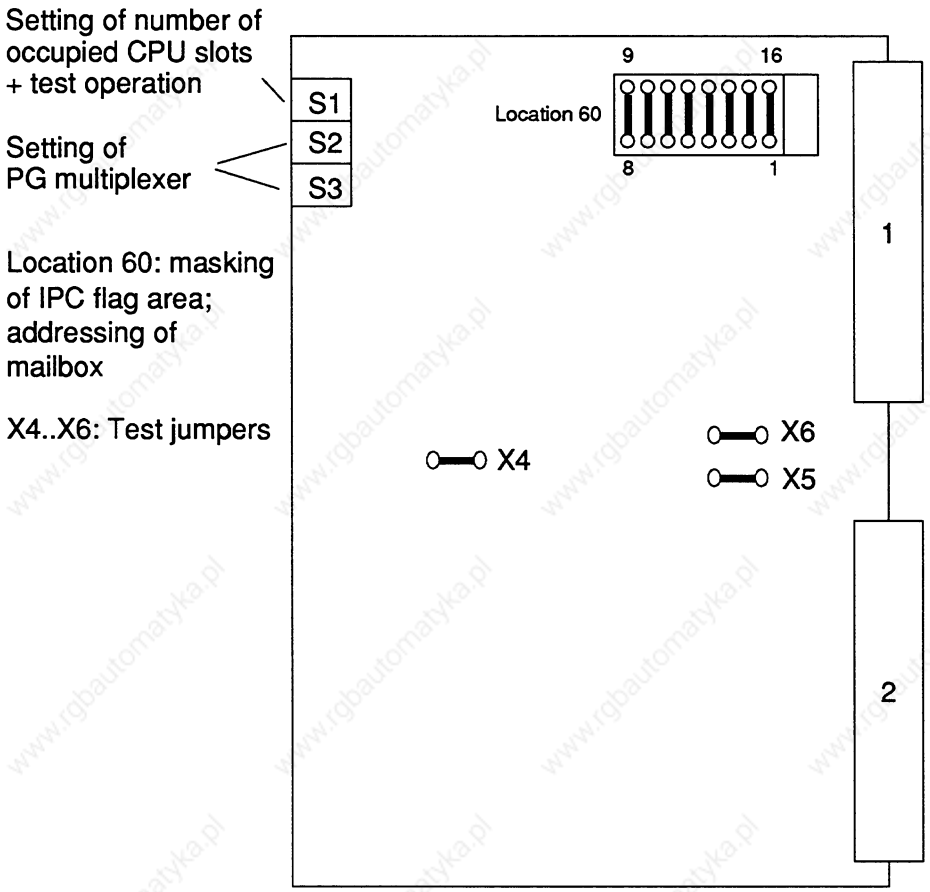


Fig. 2 Position of coding bases on 923C coordinator (factory settings)

Starting up can be divided into six steps:

- **Step 1:**

  Set the number of occupied CPU slots on the coordinator, and enable the IPC flags.

  Coding of occupied CPU slots with:

  a) 923A coordinator

  Coding by means of jumpers on the coding base in location 62:

Number of occupied CPU slots	Jumper(s) at location 62
2	7 - 10; 8 - 9
3	7 - 10
4	8 - 9

  b) 923C coordinator

  Coding by switching on only one of the DIL switches S1.4, S1.5 or S1.6 in the recess on the front panel:

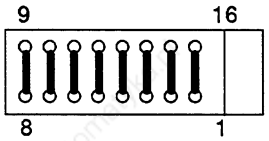
	On	Off	
S 1.1		x	—
1.2		x	—
1.3		x	Test operation (see Chapter 3.2)
1.4	x		Number of occupied CPU slots = 2
1.5		x	Number of occupied CPU slots = 3
1.6		x	Number of occupied CPU slots = 4

  "2" is the factory setting.

  Coding the IPC flags

  It may be necessary to address the mailbox on the coordinator. The 256 IPC flag bytes can be masked out in groups of 32 by removing jumpers on the coding base at location 7 on coordinator A (see Fig. 1 for position) or location 60 on coordinator C (see Fig. 2).

  IPC flag blocks must be masked out if they are used on CPs (see associated manuals). In this case the corresponding areas must be masked out on the coordinator to prevent double addressing.



In the factory setting, all IPC flag areas are unmasked (see above).

Location 60 on coordinator C

Location 7 on coordinator A

Jumper	IPC flag byte	Address
8 - 9	0 to 31	F200H to F21FH
7 - 10	32 to 63	F220H to F23FH
6 - 11	64 to 95	F240H to F25FH
5 - 12	96 to 127	F260H to F27FH
4 - 13	128 to 159	F280H to F29FH
3 - 14	160 to 191	F2A0H to F2BFH
2 - 15	192 to 223	F2C0H to F2DFH
1 - 16	224 to 255	F2E0H to F2FFH

Jumper inserted: area unmasked (coordinator acknowledges in this area)
Jumper removed: area masked (coordinator does not acknowledge in this area)

- **Step 2:**

  With the central controller switched off, plug the CPUs and the coordinator into the appropriate slots in the central controller housing.

  Insert all memory submodules (EPROM or RAM) into the CPUs or the type 355 memory module in accordance with the selected configuration. You must first program the EPROM modules on a PG programmer.

  Although it is not necessary, it is recommended that you set all mode selectors on the CPUs and the coordinator to STOP.

  The power supply can now be switched on.

Reaction:

After the power supply has been switched on, all CPUs require an "overall reset", i.e. the red STOP LED flashes quickly. Every CPU outputs the BASP signal (provided the test function is not set on the coordinator and the mode selector of the coordinator is not set to "TEST", see Section 3.2). This disables the digital outputs. The red BASP LED on the front panel of the CPU indicates this.

Possible errors:

Error response:

The RUN and STOP LEDs remain off on some CPUs; the other CPUs request an overall reset. All CPUs output the BASP signal.

Remedy:

Check the setting on the coordinator for the number of occupied CPU slots. Are the CPUs inserted without gaps?

- **Step 3:**

  Set the mode selector of the coordinator to "STOP" (if you did not already do so in Step 2), **and carry out an overall reset of all CPUs.**

  Overall reset of each CPU: switch the mode selector from "STOP" to "RUN" and back to "STOP" again while holding the operating mode key in the "OVERALL RESET" position.

  Reaction:

  The CPUs which have carried out an overall reset display a steady red STOP LED. Each CPU continues to output the BASP signal (BASP LED is on).

- **Step 4:**

  Load the user programs into an inserted RAM submodule.

  If you have not already plugged EPROM submodules containing your user program into the CPU(s) in Step 2, you must now load it into the RAMs or RAM submodules.
  You can use any combination of CPUs that run with an EPROM or RAM, or with both memory types simultaneously (CPU 946/947 only).

  Before the multiprocessor can go into cyclic operation, the peripheral allocation must be programmed in each CPU: DB 1 must be loaded. With the CPU 946/947, you must also assign parameters in DX 0 for multiprocessor operation (155U mode) and for switching off the process alarms.

An additional user program can also be loaded into the CPUs following Step 6 (in cyclic operation).

With the CPU 920 (M processor), the user program can only be loaded if the mode selector is set to "RUN". When switching from "STOP" to "RUN" in order to load the program, the steady light of the STOP LED will stay on in multiprocessor operation (in single processor operation, the steady light changes to a slower flashing light and only changes back to a steady light again when the program is being loaded).

Reaction:

No changes from the reactions following Step 3.

- **Step 5:**

  Carry out a cold restart for all CPUs 946/947, 928B, 928, 922 and 920, carry out a non-retentive cold restart for all CPUs 921 (S processor).

  This means that the mode selector of all CPUs must be switched from "STOP" to "RUN" in succession whilst holding the operating mode key in the "RESET" position.

  Reaction:

  The red STOP LED on each CPU continues to display a steady light, each CPU outputs the BASP signal.

  Possible errors:

  Error response 1:

  A CPU displays a slowly flashing STOP LED. "DB 1 error" is marked in the control bits (can be read using the programmer) of this CPU, in addition to the usual data. An ISTACK is not output.

  Remedy:

  Check whether the data block DB 1 has been programmed for this CPU.

  Error response 2:

  Following execution of cold restart:

  Undefined modes or errors occur following a cold restart of the CPUs (e.g. an R processor goes into the RUN mode following a cold restart although the other CPUs are still at stop).

Remedy:

Check the following points:

Is the coordinator plugged in?

Are all modules correctly inserted (locked in place)?

Are all modules in the correct slots?

- **Step 6:**

  a) Switch the mode selector of the coordinator from "STOP" to "RUN".

  b) If the coordinator is set to test operation (see Chapter 3.2), the mode selector can also be set to "TEST".

  Reaction:

  The green RUN LEDs of all CPUs light up permanently in both cases. All CPUs change over to cyclic operation simultaneously. The BASP signal is not output (the BASP LED is off).

  If the coordinator is **not** set to test operation, and if you switch the mode selector on the coordinator from "STOP" to "TEST", there is no reaction.

  Possible errors:

  Error response 1:

  All CPUs remain in the stop mode.

  Remedy:

  Check that the mode switches on all CPUs are in the "RUN" position.

  Subsequent starting of individual CPUs is not possible. Switch the coordinator to "STOP" again. Carry out a cold restart (non-retentive) on all CPUs and subsequently switch the coordinator to "RUN" again.

  Only those CPUs run in test mode when the coordinator is switched from "STOP" to "TEST" whose switches are in the "RUN" position.

  Error response 2:

  CPU 946/947, CPU 928B, CPU 928, CPU 922 (R processor): the STOP LED immediately displays a slow flashing light which becomes steady when you switch back from "RUN" to "STOP".

  CPU 920 (M processor): the STOP LED immediately displays a slow flashing light.

  CPU 921 (S processor): the STOP LED immediately displays a slow flashing light. This is retained when switching back from "RUN" to "STOP".

Remedy:

Check whether all CPUs 921 (S processors) have been started with a non-retentive cold restart, and all other CPUs with a cold restart.

Starting of the CPUs initiated with a cold restart is possible with the coordinator in test mode if this is switched from "STOP" to "TEST".

Notes on multiprocessor start-up

The STOP and RUN LEDs remain off with the CPU 946/947, 928B, 928, 922, 920 during the restart phase (processing of OB 20, OB 21 or OB 22). The RUN LED only displays a steady light when the CPUs change to cyclic program processing. The RUN LED of the CPU 921 (S processor), however, already displays a steady light during the restart phase.

Note:

If the 923C coordinator is used without the PG interface on the front panel being connected to the PG, and if it is switched online, the "IF FAULT" (LED) lights up on the 923C coordinator. In this case, the message can be ignored.

3 Operating Modes of the Coordinator

3.1 Normal Operation and Stop on Errors

The switchover of the individual CPUs to cyclic program execution is synchronized (unless data block DX 0 of the CPU 922, 928, 928B, 946/947 has been programmed differently, see programming instructions for the appropriate CPU), i.e. after every CPU has completed its restart, all CPUs change to cyclic program execution simultaneously.

If the mode selector of the coordinator is set to "RUN", and if one CPU goes into the stop mode, all other CPUs stop as well. The red STOP LED(s) of the CPU(s) responsible for the stoppage flash(es) slowly, the STOP LEDs of the other CPUs display a steady light.

In addition to the possible display of error LEDs on the CPU responsible for the stoppage, all CPUs output the BASP signal.

3.2 Test Operation

Test operation can be enabled by inserting the jumper 3-14 on the coding base at location 45 on coordinator A or by switching on the DIL switch S1.3 on coordinator C.

The CPUs can be operated individually if the mode selector of the coordinator is switched from "STOP" to "TEST". The switchover to cyclic program execution is therefore **not synchronized**. The output of the BASP signal is suppressed on all CPUs (even if an error occurs!).

If an error occurs on a CPU which is set to "RUN", only this CPU will stop during test operation. The error is indicated by the slow flashing of the STOP LED on the CPU. The error on this CPU does not therefore affect the other CPUs.

Caution:

Since no CPU can output the BASP signal in the event of an error during test operation, it is essential to deactivate the test mode before putting into operation to prevent a critical system condition!

If the test function is deactivated, a switchover from "STOP" to "TEST" does not produce any reaction in the CPUs.

4 The PG Multiplexer on the 923C Coordinator

The PG multiplexer on the 923C coordinator allows central access to serial PG interfaces of up to eight modules in the programmable controller via the serial PG interface on the front panel of the coordinator.
The PG software S5-DOS is required to operate the multiplexer. It is not possible to access the CPU 920 (M processor) using this method.

The instructions for the S5-135U and S5-155U describe the slots that can be reached using the PG multiplexer and coordinator.

A subscriber number between 1 and 31 (decimal) must be allocated to each module. The lowest number, the base address, is always allocated to slot 11. The subscriber numbers for the other modules are allocated as in the following tables:

S5-135U:	
Slot	**Subscriber No.**
11	Base address
27	Base address + 1
43	Base address + 2
59	Base address + 3
75	Base address + 4
83	Base address + 5
91	Base address + 6
99	Base address + 7

S5-155U:	
Slot	**Subscriber No.**
11/27	Base address
51/67	Base address + 1
91	Base address + 2
99	Base address + 3
107	Base address + 4
115	Base address + 5
123	Base address + 6
131	Base address + 7

The base address is set on the DIL switch S2 located in the recess of the front panel of the 923C coordinator. The base address is the sum of the binary values set by the switches in the "On" position:

	Off	On	
S 2.1	x		—
2.2	x		Value 16
2.3	x		Value 8
2.4	x		Value 4
2.5	x		Value 2
2.6		x	Value 1

(Factory setting) (In this case, base address = 1)

Note:

When setting the base address, make sure that no subscriber number is larger than 31.

Switch S3, which is also located in the recess of the front panel of the C coordinator, is used to enable the slots which are to be operated from the PG multiplexer:

Example with S5-135U:

		Off	On		
S	3.1	x		Slot 11	(1st CPU slot)
	3.2	x		Slot 27	(2nd CPU slot)
	3.3	x		Slot 43	(3rd CPU slot)
	3.4	x		Slot 59	(4th CPU slot)
	3.5	x		Slot 75	
	3.6	x		Slot 83	
	3.7	x		Slot 91	
	3.8	x		Slot 99	

All slots are disabled in the factory setting.

If slots are not occupied, or if modules are to be operated via their own front connectors, these slots must be disabled (= masked). It is especially important to mask the corresponding slot when using a CPU 920 (M processor).

The front connector of the interface must not be inserted at the same time when the module is operated via the multiplexer.

Note:

The "On" position of the DIL switch S1 is to the left, but it is to the right with switches S2 and S3.

S5-DOS must be used with the programmer. The "Bus selection" package establishes the desired link between the PG and subscriber (refer to instructions of the respective programmer for further details).

Example of a setting on the 923C coordinator (S5-135U):

	Off	On	
S 2.1	x		—
2.2	x		Value 16
2.3		x	Value 8 --> 8
2.4		x	Value 4 --> 4
2.5	x		Value 2
2.6		x	Value 1 --> 1

Base address 13

	Off	On		
S 3.1		x	Slot 11	Address 13
3.2	x			
3.3	x			
3.4	x			
3.5		x	Slot 75	Address 17
3.6		x	Slot 83	Address 18
3.7	x			
3.8		x	Slot 99	Address 20

SIEMENS

SIMATIC S5

Multiprocessor Communication
S5-135U, CPU 922 (R Processor),
CPU 928 and CPU 928B
S5-155U, CPU 946/947

User's Guide

C79000-B8576-C468-05

www.rgbautomatyka.pl

Page

1	**Introduction**	**3**
1.1	**Configuration**	**4**
1.2	**Principle**	**4**
1.3	**Sender/Receiver Identification**	**5**
1.4	**Buffering the Data**	**6**
1.5	**System Restart**	**9**
1.6	**Calling and Nesting the Special Function Organization Blocks OB 200 and OB 202 to OB 205**	**10**
1.7	**Parallel Processing in a Multiprocessor Programmable Controller**	**11**
1.8	**Required Memory Areas**	**11**
1.9	**Runtime**	**12**
2	**Parameter Assignment**	**14**
2.1	**Evaluating the Output Parameters**	**14**
2.1.1	Condition Codes	15
2.1.2	Condition Code Byte: Initialization Conflict/Error/Warning	16
3	**INITIALIZE Function (OB 200)**	**21**
3.1	**Input Parameters**	**23**
3.1.1	Mode (Automatic / Manual)	23
3.1.2	Number of CPUs	24
3.1.3	Block ID and Number / Start Address of the Assignment List	24
3.2	**Output Parameters**	**26**
3.2.1	Condition Code Byte	26
3.2.2	Total Capacity	28
4	**SEND Function (OB 202)**	**29**
4.1	**Input Parameters**	**29**
4.1.1	Receiving CPU	29
4.1.2	Block ID and Number / Field Number	29
4.2	**Output Parameters**	**31**
4.2.1	Condition Code Byte	31
4.2.2	Transmitting Capacity	33

5	**SEND TEST Function (OB 203)**	**34**
5.1	**Input Parameters**	**34**
5.1.1	Receiving CPU	34
5.2	**Output Parameters**	**34**
5.2.1	Condition Code Byte	34
5.2.2	Transmitting Capacity	35
6	**RECEIVE Function (OB 204)**	**36**
6.1	**Input Parameters**	**36**
6.1.1	Transmitting CPU	36
6.2	**Output Parameters**	**37**
6.2.1	Condition Code Byte	37
6.2.2	Receiving Capacity	38
6.2.3	Block ID and Number	38
6.2.4	Address of the First/Last Received Data Word	39
7	**RECEIVE TEST Function (OB 205)**	**40**
7.1	**Input Parameters**	**40**
7.1.1	Transmitting CPU	40
7.2	**Output Parameters**	**40**
7.2.1	Condition Code Byte	40
7.2.2	Receiving Capacity	41
8	**Applications**	**42**
8.1	**Calling the Special Function OB Using Function Blocks**	**42**
8.1.1	Setting Up a Buffer (FB 200)	43
8.1.2	Sending a Block of Data (FB 202)	45
8.1.3	Testing the Transmitting Capacity (FB 203)	47
8.1.4	Receiving a Block of Data (FB 204)	48
8.1.5	Testing the Receiving Capacity (FB 205)	50
8.2	**Transferring Data Blocks**	**51**
8.2.1	Functional Description	51
8.2.2	Transferring a Data Block (FB 110)	51
8.2.3	Application Example (for the S5-135U)	54
8.3	**Extending the IPC Flag Area**	**56**
8.3.1	The Problem	56
8.3.2	The Solution	57
8.3.3	Data Structure	57
8.3.4	Program Structure	60
8.3.5	Sending Data Word Areas (FB 100)	62
8.3.6	Receive Data Word Areas (FB 101)	65
8.3.7	Application Example (for S5-135U)	68

1 Introduction

You can operate the multiprocessor programmable controllers S5-135U and S5-155U with up to four CPUs. You can use the following "tools" individually or in combination to exchange data between the CPUs:

- F flags are transferred, if you define them as interprocessor communication (IPC) output flags in **one** CPU and as IPC input flags in one or more CPUs.

- To transfer data blocks, or to be more precise, blocks of data with a maximum length of 64 bytes (= 32 data words), you can use the following special functions that are integrated in the CPU:

INITIALIZE (OB 200) :	preassign
SEND (OB 202):	send a block of data
SEND TEST (OB 203):	test sending capacity
RECEIVE (OB 204):	receive a block of data
RECEIVE TEST (OB 205):	test receiving capacity

  To use these functions, you only require basic knowledge of the STEP 5 programming language and the way in which SIMATIC S5 programmable controllers operate. You can obtain this basic information from the publications listed in the table of documentation.

Whereas the IPC flags are updated "automatically" by the system program, you must call the **INITIALIZE, SEND, SEND TEST, RECEIVE and RECEIVE TEST** functions as special function organization blocks using the **JU OB** or **JC OB** operations.

1.1 Configuration

S5-135U or S5-155U

These PLCs contain the S5 bus and in the multiprocessor mode they also have the following components:

- **1 coordinator 923C**

  This module contains four pages. These are memory areas of 1024 bytes. They all occupy the address area F400H to F7FFH. You select (address) the "current" page using the select register (also known as the identification or page address register, similar to chip select). The select numbers 252, 253, 254 and 255 are **fixed** as the four pages of the 923C coordinator and are used for multiprocessor communication.

- **2 to 4 CPUs**

  For the S5-135U: CPU 922 (R processor), CPU 928 , CPU 928B or CPU 920 (M processor)
  For the S5-155U: CPU 946/947, CPU 922 (R processor), CPU 928 , CPU 928B or CPU 920 (M processor).

  These CPUs can exchange data with each other in any combination, you can also use "handling blocks" which also work with page addressing without any restrictions.
  If you have one or more additional CPU 921s (S processors) in the same rack, they **cannot** take part in the multiprocessor communication. You must not call the S processor handling blocks as long as R and M processors, CPU 928s, CPU 928Bs and CPU 946/947s are processing their handling blocks or are involved in multiprocessor communication. CPUs can, however, always communicate via IPC flags.

1.2 Principle

To transfer data, you must activate the SEND function on the transmitting CPU and the RECEIVE function on the receiving CPU.
The data words of a DB or DX data block located in the transmitting CPU are transported via the coordinator 923C to the receiving CPU one after the other and written to the DB or DX data block with the same number and under the same data word address; i.e. this represents a "1:1" copying.

Example

	Data to be sent in the transmitting CPU	Data received in the receiving CPU
Data block	DB 17	DB 17
Data word address	DW 32 to DW 63	DW 32 to DW 63

The amount of data that can be transferred with the SEND and RECEIVE functions is normally 32 words.
If the block length (without header) is not a multiple of 32 words, the last block of data to be transferred is an exception and is less than 32 words.

The data block in the receiving CPU can be longer or shorter than the data block to be sent. It is, however, important that the data words transferred by the SEND function exist in the receiving block; otherwise the RECEIVE function signals an error.

1.3 Sender/Receiver Identification

The CPUs are numbered so that the leftmost CPU has the number 1 and each subsequent CPU to the right has a number increased by 1.

Example

S5-135U/155U:

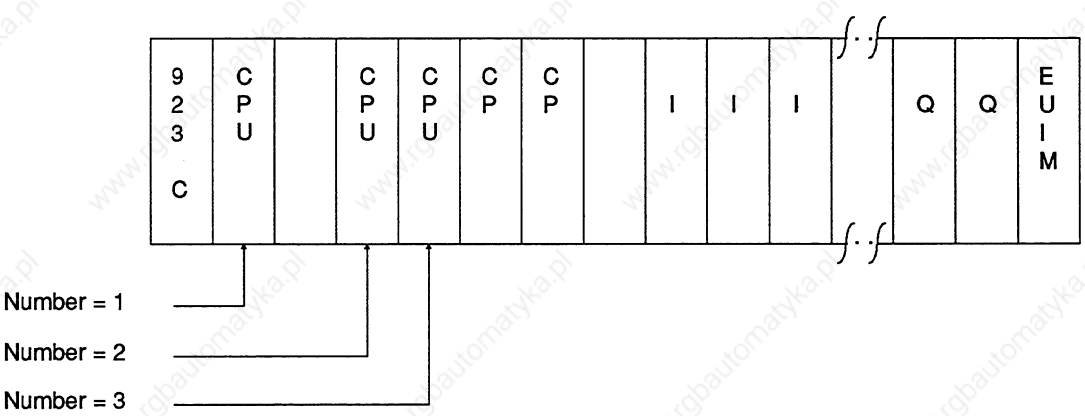


Fig. 1 Sender/receiver identification

1.4 Buffering the Data

The cycle time of a CPU depends on the number of tasks the CPU executes and the performance of the CPU itself. Among other things, the cycle time is determined by the following:

- the size of the individual program sections,
- how often the program sections are required (multiple calls, loops),
- the number of closed loop controllers (with CPU 922, CPU 928 and CPU 928B).

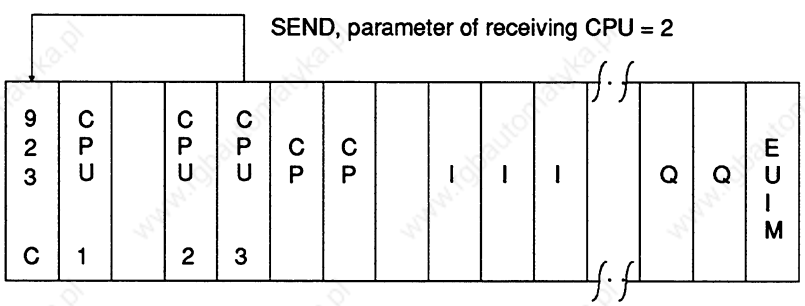
The cycle time of a CPU **varies** depending on the number of conditional block calls (e.g. JC PB xy), the occurrence of interrupts (e.g. interrupt-driven processing via OB 2) and similar. This means that the cyclic program execution in each individual CPU is **asynchronous** to the cyclic program execution of the other CPUs.

In contrast to cyclic program execution, time-controlled program execution is processed periodically depending on a clock signal, for example every 100 ms (OB 13). In this example, the clock signal of a CPU can be delayed by up to 100 ms compared with another CPU.
Because of this asynchronous processing, the data to be transferred are buffered on the coordinator 923C.

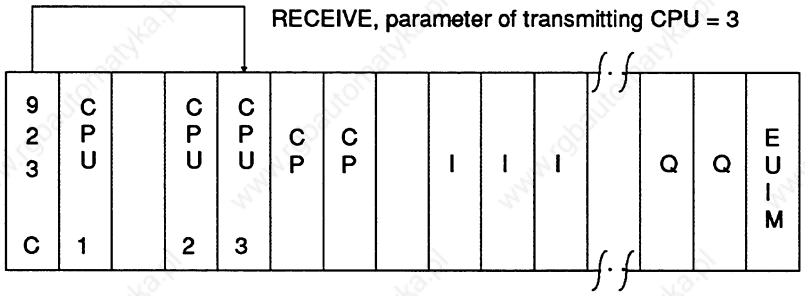
The CPU's "own" number and the number of a receiver (for the SEND function) or the number of a transmitter (for the RECEIVE function) specify the source and destination.

Example: data transfer from CPU 3 to CPU 2

1st step



2nd step



- 1st step — the buffer is based on the FIFO principle (first in, first out queue principle). The data is received in the order in which it is sent. This applies to each individual transmission (identified by the transmitting and receiving CPU) and is independent of other connections.

- 2nd step — the buffer is battery-backed; this means that the "automatic warm restart following power down" is possible without any restrictions. A loss of power during a data transfer does not cause any loss of data in the programmable controller.

The coordinator 923C has a memory capacity of 48 data fields each capable of containing 32 words. The INITIALIZE function assigns these fields to individual connections.
Each **memory field** (always with a length of 32 words) can hold exactly one **block of data** (with a length between 1 data word and 32 data words). The SEND block enters one **block of data** in a **memory field** from where it is read out by the RECEIVE block.
The number of memory fields assigned to a connection is directly related to the parameters for the transmitting capacity (SEND, SEND TEST function) and receiving capacity (RECEIVE, RECEIVE TEST function).

The transmitting capacity indicates how many of the memory fields reserved for a connection are free at any particular time.

The receiving capacity indicates how many of the memory fields reserved for a connection are occupied at any particular time.

The sum of the transmitting and receiving capacity parameters is always equal to the number of memory fields reserved for a connection.

Example

The following table indicates a possible data transfer sequence assuming that the connection "from CPU 3 to CPU 2" has seven memory fields assigned by the INITIALIZE function.

Transmitter: CPU 3			Receiver: CPU 2	
Sequence	Transmitting capacity (free memory fields)	Time	Receiving capacity (occupied memory fields)	Sequence
				Initialize
Sends a blocks of data (A)	7		0	
Sends four blocks of data (B,C,D,E)	6		1	
	2		5	Receives two blocks of data (A, B)
Sends four blocks of data (F,G,H,I)	4		3	
	0		7	Receives five blocks of data (C,D,E,F,G)
Sends two blocks of data (K,L)	5		2	Receives two blocks of data (H,I)
	5		2	

REMEMBER

Sending/receiving n data blocks means that the corresponding function is called n times.

To simplify the representation, at any one time, data can either be sent or received in this example.
It is, however, possible and useful to transmit (CPU 3) and receive (CPU 2) simultaneously (see "Parallel processing in a multiprocessor programmable controller"). In the example, blocks H and I are received while blocks K and L are sent.

The example illustrates the queue organization of the buffer; the blocks of data sent first (A,B,C...) are received first (A,B,C...).

Summary

Buffering data on the coordinator 923C allows the asynchronous operation of transmitting and receiving CPUs and compensates for their different processing speeds.

Since the capacity of the buffer is limited, the receiver should check "often" and "regularly" whether there are data in the buffer (RECEIVE TEST function, receiving capacity > 0) and should attempt to fetch stored data (RECEIVE function). Ideally, the RECEIVE function should be repeated until the receiving capacity is zero. This means that the transmitted data are not buffered for a longer period of time and that the receiver always has the current data. This also means that memory fields remain free (the transmitting capacity is increased) and prevents the sender from being blocked (i.e. when the transmitting capacity is zero).

A receiving capacity of zero represents the ideal state (i.e. all transmitted data have been fetched by the receiver), on the other hand a transmitting capacity of zero indicates incorrect planning, as follows:

- the SEND function is called too often.
- the RECEIVE function is not called often enough,
- there are not enough memory fields assigned to the connection. The capacity of the buffer is insufficient to compensate temporary imbalances in the frequency with which the CPUs transmit and receive data.

1.5 System Restart

If you require multiprocessor communication, then all the CPUs involved must go through the **same** STOP-RUN transition (= RESTART), i.e. all the CPUs go through a COLD RESTART or all CPUs go through a WARM RESTART.

You must make sure that the restart of at least all the CPUs involved in the communication is **uniform** (see Chapter 10), in the following ways:

- **direct operation** (front switch, programmer),
- **parameter assignment** (DX 0) and/or
- **programming** (using the special function organization block OB 223 "stop if non-uniform restarts occur in the multiprocessor mode").

COLD RESTART

In organization block OB 20 (COLD RESTART) **one** CPU must set up the buffer (in the 923C) using the INITIALIZE function. Any existing data is lost.
Following this, i.e. during the RESTART, you can call the SEND, SEND TEST, RECEIVE, RECEIVE TEST functions in the individual CPUs. With appropriate programming, you must make sure that this only occurs after the buffer in the coordinator has been correctly initialized.
On completion of the RESTART, i.e. in the RUN mode, the user program is processed from the beginning, i.e. from the first operation in OB 1 or FB 0.

WARM RESTART

You must **not** use the INITIALIZE function in the organization blocks OB 21 (MANUAL WARM RESTART) and OB 22 (AUTOMATIC WARM RESTART). Calling the SEND, SEND TEST, RECEIVE, RECEIVE TEST functions can cause problems (refer to the following section).
On completion of the WARM RESTART, i.e. in the RUN mode, the user program is not processed from the start, **but** from the point at which it was interrupted. The point of interruption can, for example, be within the SEND function.

1.6 Calling and Nesting the Special Function Organization Blocks OB 200 and OB 202 to OB 205

The simplest procedure is as follows:

- program the call for the INITIALIZE function only in the cold restart organization block OB 20;
- program the call for the SEND, SEND TEST, RECEIVE, RECEIVE TEST functions either **only** within the cyclic program **or only** within the time-driven program.

REMEMBER

Depending on the assignment of parameters in DX 0 ("interrupts at command boundaries" for the CPU 928B, CPU 928 and CPU 920, or "155U mode" for the CPU 946/947), and the type of program execution (WARM RESTART, interrupt handling, e.g. OB 26 for cycle time error) it is possible that one of the functions INITIALIZE, SEND, SEND TEST, RECEIVE and RECEIVE TEST can be interrupted. If a user interface inserted at the point of interruption (e.g. OB 13 when interrupts are possible at operation boundaries or OB 22 following power down) also contains one of the functions SEND, SEND TEST, RECEIVE and RECEIVE TEST an illegal call (double call) is recognized and an error is signalled (error number 67, Section 2.1.2).

1.7 Parallel Processing in a Multiprocessor Programmable Controller

Once you have completed the assignment of the buffer (INITIALIZE function), you can execute the functions SEND, SEND TEST, RECEIVE and RECEIVE TEST in any combination and with any parameter assignment in all the CPUs simultaneously and parallel to each other.

Taking a single connection (from CPU 'SE' to CPU 'RE') it is possible to execute the SEND function (CPU 'SE') and the RECEIVE function (CPU 'RE') simultaneously. While CPU 'SE' is sending blocks of data to the coordinator, CPU 'RE' can receive (fetch) buffered blocks of data from the coordinator.

1.8 Required Memory Areas

The special function organization blocks OB 200 and OB 202 to OB 205 do not require a working area (e.g. for buffering variables) and do not call data blocks. They do, of course, access areas containing parameters, although only the parameters marked as output parameters are modified. These OBs also affect the condition codes (CC1, RLO etc., see Section 2.1).

- CPU 922, CPU 928, CPU 928B: The contents of ACCU 1 to ACCU 4 and the contents of the registers are not affected by the special function OBs for multiprocessor communication.
- CPU 946/947: The contents of all registers and ACCU 1, 2 and 3 remain the same, only the contents of ACCU 4 are affected.

1.9 Runtime

Special function OB		Runtime			
Block / name	Block function	R processor	CPU 928	CPU 946/947	CPU 928B
OB 200 / initialize	Assign buffer	230 ms	130 ms	128 ms	130 ms
OB 202 / send	Send a block of data (32 data words)	806 µs (294 µs basic time + 16 µs / word); 118 µs if a warning occurs	666 µs (250 µs basic time + 13 µs / word); 115 µs if a warning occurs	762 µs (426 µs basic time + 21 µs / double word); 243 µs if a warning occurs	696 µs (280 µs) basic time +31µs/word); 145 µs if a warning occurs
OB 203 / send test	Test transmitting capacity	72 µs	50 µs	207 µs	80 µs
OB 204/ receive	Receive a block of data (32 data words)	825 µs (281 µs basic time + 17 µs / word); 115 µs if a warning occurs	660 µs (244 µs basic time + 13 µs / word); 98 µs if a warning occurs	772 µs (421 µs basic time + 22 µs / double word); 243 µs if a warning occurs	690 µs (274 µs basic time + 13 µs/ word); 128 µs if a warning occurs
OB 205 / receive test	Test receiving capacity	70 µs	48 µs	223 µs	78 µs

The "runtime" is the processing time of the special function organization blocks; the time from calling a block to its termination can be much greater if it is interrupted by higher priority activities (e.g. updating timers, processing closed loop controllers etc.).

The runtimes listed above assume that of four CPUs inserted in a rack, only the CPU whose runtimes are being measured accesses the SIMATIC S5 bus. If other CPUs use the bus intensively, the runtime increases particularly for the send/receive functions.

An important factor of a connection (from CPU 'SE' to CPU 'RE') is the total data transfer time. This is made up of the following components:

- time required to send (see runtime)
- length of time the data are buffered (on the 923C coordinator)
- the time required to receive data (see runtime)

The length of time that the data are "in transit" is largely dependent on the length of time that the data is buffered and therefore on the structure of the user program (see "Buffering Data").

2 Parameter Assignment

The "actual" parameters are located in a maximum 10 byte long data field in the F flag area. The number of the first flag byte in the data field (= pointer to the data field) must be loaded in ACCU-1-L. Permitted values are 0 to 246.
The data field is divided into an area for **input parameters** and an area for **output parameters**.

- **Input parameters**

  All or part of the input parameters are read and evaluated by the functions, the functions do not write to this area.

- **Output parameters**

  Some or all of the output parameters are written to by the functions, the functions do not read this area.

REMEMBER

You can assign **a flag area with 10 flag bytes** for **all** communications functions. The functions themselves require different numbers of bytes. Refer to the description of the single functions (Chapters 3 to 7).

Example: data field with parameters for the RECEIVE function (OB 204)

FY x + 0:	transmitting CPU	input parameter
FY x + 1:	—	not used
FY x + 2:	condition code byte	output parameter
FY x + 3:	receiving capacity	output parameter
FY x + 4:	block ID	output parameter
FY x + 5:	block number	output parameter
FY x + 6:	address of the first	output parameter
FY x + 7:	received data word	output parameter
FY x + 8:	address of the last	output parameter
FY x + 9:	received data word	output parameter

This example illustrates that the number of the first F flag byte in the data field must not be higher than FY 246, since otherwise the parameter field of up to 10 bytes would exceed the limits of the flag area (FY 255).

2.1 Evaluating the Output Parameters

Output parameters are data made available to the user program for evaluation. Among other things, they indicate whether or not a function could be executed and if not they indicate the reason for the termination of the function.

2.1.1 Condition Codes

The INITIALIZE, SEND, SEND TEST, RECEIVE and RECEIVE TEST functions affect the condition codes (see programming instructions for your CPUs, general notes on the STEP 5 operations):

- the OV and OS bits (word condition codes) are always cleared,
- the OR, STA, ERAB bits (bit condition codes) are always cleared,
- RLO, CC 0 and CC 1 indicate whether a function has been executed correctly and completely.

  RLO = 0: Function executed correctly and completely

  RLO = 1: Function aborted: the pointer to the data field in the flag area may have an illegal value, i.e. the low word of the ACCU contains a value greater than 246.
  In the following sections, it is assumed that the pointer to the data field contains a correct value. The first byte of the output parameter provides detailed information about the cause of termination.

  CC 1 = 1: Additional warning information (warning number 1 or 2)

  CC 0 = 1: Additional error indication (error number 1-9)

Situation	Condition codes			Evaluation with operation
	RLO	CC 1	CC 0	
Function executed completely and correctly	0	0	0	JC =
Function aborted, pointer to data field illegal	1	0	0	JC =
Function aborted owing to an initialization conflict	1	0	0	JC =
Function aborted owing to an error	1	0	1	JC = and JM =
Function aborted owing to a warning	1	1	0	JC = and JP =

2.1.2 Condition Code Byte: Initialization Conflict/Error/Warning

The first byte in the field of the output parameters (condition code byte) also indicates whether or not a function has been correctly and completely executed. This byte contains detailed information about the cause of termination of a function.
Assuming that at least the pointer to the data field contains a correct value, this byte is **always** relevant.

If the function has been executed correctly and completely, all the bits are cleared (= 0), and all other output parameters are relevant.

If the function is aborted with a warning (bit $2^7 = 1$), only the condition code for the transmitting/ receiving capacity is relevant, other output parameters (if they exist) are unchanged.

If the function is aborted owing to an error (bit $2^6 = 1$) or an initialization conflict (bit $2^5 = 1$), all other output parameters remain unchanged.

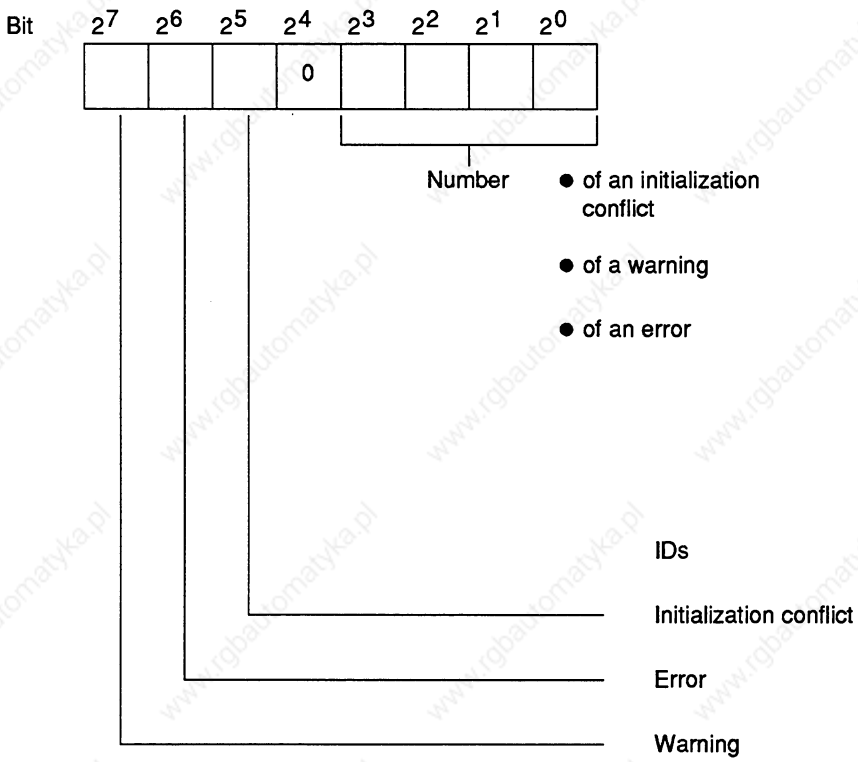


Fig. 2 Coding of the first byte

Evaluation

The identifiers in bit positions 2^5 to 2^7 indicate the significance of the numbers in bit positions 2^0 to 2^3.
Apart from this bit-by-bit evaluation, it is also possible to interpret the whole condition code byte as a fixed point number without sign. If you interpret the condition code byte as a **byte**, the groups of numbers have the following significance:

Number group	Significance
0	Function executed correctly and completely
33 to 42	Function aborted owing to an initialization conflict
65 to 73	Function aborted owing to an error
129 to 130	Function aborted owing to a warning

The values of the following numbers **also** indicate **the order** in which errors or initialization conflicts were recognized and indicated by the functions.

Example
The SEND function indicates an error and is not executed. If you then make program and/or parameter modifications and the SEND function once again indicates an error with a higher number than previously, you can assume that you have corrected one of several errors.

Initialization conflict

An initialization conflict can only occur with the INITIALIZATION function. If a conflict occurs, you must modify the program or parameters.

Initialization conflict numbers (evaluation of the condition code byte as a byte)

- **(33)** The pages required for multiprocessor communication (numbers 252 to 255) are not or not all available.
- **(34)** The pages required for multiprocessor communication (numbers 252 to 255) are defective.
- **(35)** The parameter "automatic/manual" is illegal. The following errors are possible:

  The "automatic/manual" ID is less than 1.

  The "automatic/manual" ID is greater than 2.

- **(36)** The parameter "number of CPUs" is illegal. The following errors are possible:

  The number of CPUs is less than 2.

  The number of CPUs is greater than 4.

- **(37)** The parameter "block ID" is illegal. The following errors are possible:

  The block ID is less than 1.

  The block ID is greater than 2.

- **(38)** The parameter block number" is illegal, since it is a data block with a special significance. The following errors are possible:

  If block ID = 1 : DB 0, DB 1, DB 2

  If block ID = 2 : DX 0, DX 1, DX 2

- **(39)** The parameter "block number" is incorrect, since the data block does not exist.
- **(40)** The parameter "start address of the assignment list" is too high or the data block is too short.
- **(41)** The assignment list in the data block is not correctly structured.
- **(42)** The sum of the assigned memory fields is greater than 48.

Errors

If an error occurs, you must change the program/parameters.

Error numbers (evaluation of the condition code byte as a byte)

- **(65)** The parameter "receiving CPU" (SEND, SEND TEST) is illegal, since it is a data block with a special significance. The following errors are possible:

  The number of the receiving CPU is greater than 4.

  The number of the receiving CPU is less than 1.

  The number of the receiving CPU is the same as the CPU's own number.

- **(66)** The parameter "transmitting CPU" (RECEIVE, RECEIVE TEST) is illegal, since it is a data block with a special significance. The following errors are possible:

  The number of the transmitting CPU is greater than 4.

  The number of the transmitting CPU is less than 1.

  The number of the transmitting CPU is the same as the CPU's own number.

- **(67)** The special function organization block call is wrong (SEND, RECEIVE, SEND TEST,RECEIVE TEST). The following errors are possible:

  a) Secondary error, since the INITIALIZE function could not be called or was terminated by an initialization conflict.

  b) Double call: the call for this function, SEND, SEND TEST, RECEIVE or RECEIVE TEST is illegal, since one of the functions INITIALIZE, SEND, SEND TEST, RECEIVE or RECEIVE TEST has already been called in this CPU in a lower processing level (e.g. cyclic program execution). (See "Calling and nesting the special function organization blocks".)

  c) The CPU's own number is incorrect (system data corrupted); following power down/power up the CPU number is generated again by the system program.

- **(68)** The management data (queue management) of the selected connections are incorrect; set up the buffer in the coordinator 923C again using the INITIALIZE function (SEND, RECEIVE, SEND TEST, RECEIVE TEST).

- **(69)** The parameter "block ID" (SEND) or the block ID provided by the sender (RECEIVE) is illegal. The following errors are possible:

  The block ID is less than 1.

  The block ID is greater than 2.

- **(70)** The parameter "block number" (SEND) or the block number supplied by the sender (RECEIVE) is illegal, since it is a data block with a special significance. The following errors are possible:

  If the block ID = 1 : DB 0, DB 1, DB 2

  If the block ID = 2 : DX 0, DX 1, DX 2

- **(71)** The parameter "block number" (SEND) or the block number provided by the sender (RECEIVE) is incorrect. The specified data block does not exist.

- **(72)** The parameter "field number" (SEND) is incorrect. The data block is too short or the field number too high.

- **(73)** The data block is not large enough to receive the block of data transmitted by the sender (RECEIVE).

Warning

The function could not be executed; the function call must be repeated, e.g. in the next cycle.

Warning numbers (evaluation of the condition code byte as a byte)

- **(129)** The SEND function cannot transfer data, since the transmitting capacity was already zero when the function was called.

- **(130)** The RECEIVE function cannot accept data, since the receiving capacity was already zero when the function was called.

3 INITIALIZE Function (OB 200)

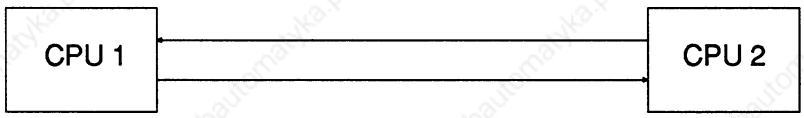
Call parameters

1st data field	Before calling OB 200, you must supply the input parameters in the data field.OB 200 requires eight F flag bytes in the data field for input and output parameters:		
	FY x + 0:	Mode (automatic/ manual)	input parameter
	FY x + 1:	Number of CPUs	input parameter
	FY x + 2:	Block ID	input parameter
	FY x + 3:	Block number	input parameter
	FY x + 4:	Start address of the	input parameter
	FY x + 5:	assignment list	input parameter
	FY x + 6:	Condition code byte	output parameter
	FY x + 7:	Total capacity	output parameter
2 ACCU-1-L:	No. of the 1st flag byte "x" in the data field, permitted values:		ACCU-1-LH:0 ACCU-1-LL: 0 to 246

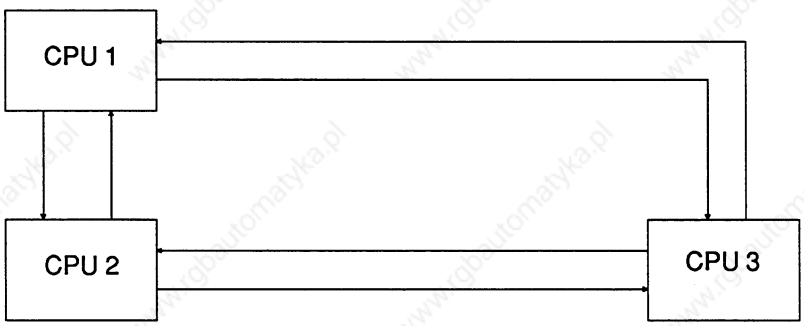
To transfer data from one CPU to another CPU, the data must be temporarily buffered. The INITIALIZE function sets up a buffer on the KOR 923C coordinator.
The memory capacity is stipulated in fields (with a length of 32 words).

Each memory field (always with a length of 32 words) accepts one block of data (with a length between one data word and 32 data words). A block of data is entered in a memory field by a SEND block and read out by a RECEIVE block.

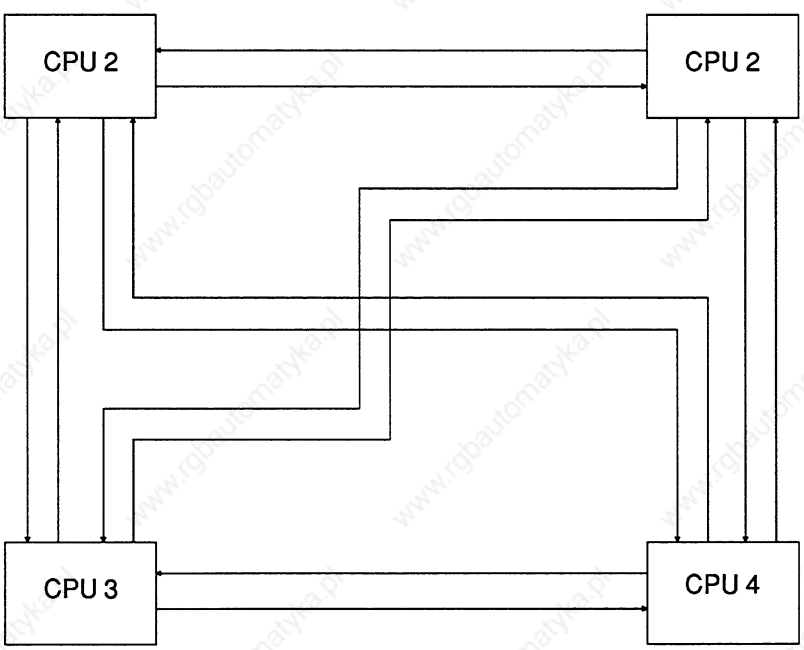
If you are using two CPUs, there are two connections (transfer directions, "channels"):



If you are using three CPUs, there are six connections:



If you are using four CPUs, there are twelve connections:



The INITIALIZE function specifies how the total of **48** available memory fields are assigned to the maximum twelve connections.
This means that each possible connection, specified by the parameters "transmitting CPU" and "receiving CPU" has a certain memory capacity available.

REMEMBER

Before you can call the SEND / RECEIVE / SEND TEST / RECEIVE TEST functions, one CPU must have already called the INITIALIZE function and executed it completely and without errors.

If the INITIALIZE function is called several times, one after the other, the last assignment made is valid. While a CPU is processing the INITIALIZATION function, no other functions including the INITIALIZE function can be called on other CPUs.

3.1 Input Parameters

3.1.1 Mode (Automatic / Manual)

Mode = 1 : automatic
Mode = 2 : manual
Mode = 0 or 3 to 255 : illegal, causes an initialization conflict

"Automatic" mode

If you select the "automatic" mode, the memory fields available are divided **equally** according to the number of CPUs:

Number of CPUs	Number of connections	Memory fields per connection
2	**2**	**24**
3	**6**	**8**
4	**12**	**4**
0; 1; 5 to 255	Illegal: causes an initialization conflict	

"Manual" mode

If you select the "manual" mode, you must create an assignment list in a data block in which the 48 (or less) available memory fields are assigned to the maximum 12 connections according to a fixed scheme. This function is particularly useful when some connections have far more data traffic than others. For example, CPUs 921 (S processors) cannot take part in the multiprocessor communication described here; the potential connections between this CPU and other CPUs do not therefore need memory fields and should not have memory fields assigned to them. The parameters

- block ID,
- block number and the
- start address of the assignment list

specify where the assignment list is stored. These three parameters are therefore only relevant for the "manual" mode.

3.1.2 Number of CPUs

This parameter is only relevant if you select the "automatic" mode; (see 3.1.1)

3.1.3 Block ID and Number / Start Address of the Assignment List

These parameters are only relevant if you select the "manual" mode.

Block ID and number

ID = 1:	DB data block
ID = 2:	DX data block
ID = 0 or 3 to 255 :	illegal, causes an initialization conflict

For the block number, you specify the number of the DB or DX data block in which the assignment list is stored.

Start address of the assignment list

Along with the block ID and number, this specifies the area (or more precisely, the start address of the area) in which the assignment list is stored.
The assignment list contains further input parameters for the INITIALIZE function, i.e. this area is only read (the contents are not changed). The assignment list has the structure shown on the following page:

Assignment list

Data word	:	Format	Value	:	Significance
DW n + 0	:	KS	S1	:	Transmitter = CPU 1
DW n + 1	:	KY	2 , **a**	:	Receiver = CPU 2
DW n + 2	:	KY	3 , **b**	:	Receiver = CPU 3
DW n + 3	:	KY	4 , **c**	:	Receiver = CPU 4
DW n + 4	:	KS	S2	:	Transmitter = CPU 2
DW n + 5	:	KY	1 , **d**	:	Receiver = CPU 1
DW n + 6	:	KY	3 , **e**	:	Receiver = CPU 3
DW n + 7	:	KY	4 , **f**	:	Receiver = CPU 4
DW n + 8	:	KS	S3	:	Transmitter = CPU 3
DW n + 9	:	KY	1 , **g**	:	Receiver = CPU 1
DW n + 10	:	KY	2 , **h**	:	Receiver = CPU 2
DW n + 11	:	KY	4 , **i**	:	Receiver = CPU 4
DW n + 12	:	KS	S4	:	Transmitter = CPU 4
DW n + 13	:	KY	1 , **k**	:	Receiver = CPU 1
DW n + 14	:	KY	2 , **l**	:	Receiver = CPU 2
DW n + 15	:	KY	3 , **m**	:	Receiver = CPU 3

REMEMBER

You must keep to this structure even if you have less than four CPUs.

The lower case letters a to m in bold face represent numbers between 0 and 48; **the sum of these numbers must not exceed 48.**
The next page shows an example of a completed assignment list.

Example

You have three CPUs in your rack, CPU 2 sends a lot of data to the other two CPUs. The other two CPUs, however, only send a small amount of data back to CPU 2 as acknowledgements in a logical handshake. There is no data exchange between CPU 1 and CPU 3.

Assignment list

Data word	:	Format	Value	:	Significance
DW n + 0	:	KS	S1	:	Transmitter = CPU 1
DW n + 1	:	KY	2 , **2**	:	Receiver = CPU 2
DW n + 2	:	KY	3 , **0**	:	Receiver = CPU 3
DW n + 3	:	KY	4 , **0**	:	Receiver = CPU 4
DW n + 4	:	KS	S2	:	Transmitter = CPU 2
DW n + 5	:	KY	**1 , 22**	:	Receiver = CPU 1
DW n + 6	:	KY	3 , **22**	:	Receiver = CPU 3
DW n + 7	:	KY	**4 , 0**	:	Receiver = CPU 4
DW n + 8	:	KS	S3	:	Transmitter = CPU 3
DW n + 9	:	KY	1 , **0**	:	Receiver = CPU 1
DW n + 10	:	KY	2 , **2**	:	Receiver = CPU 2
DW n + 11	:	KY	4 , **0**	:	Receiver = CPU 4
DW n + 12	:	KS	S4	:	Transmitter = CPU 4
DW n + 13	:	KY	1 , **0**	:	Receiver = CPU 1
DW n + 14	:	KY	2 , **0**	:	Receiver = CPU 2
DW n + 15	:	KY	**3 , 0**	:	Receiver = CPU 3

3.2 Output Parameters

3.2.1 Condition Code Byte

This byte informs you whether the INITIALIZE function was executed correctly and completely.

Initialization conflict

The initialization conflicts listed are recognized and indicated by the function in the ascending order of their numbers.

If an initialization conflict occurs, you must change the program / parameters.

Initialization conflict numbers (evaluation of the condition code byte as a byte)

- **(33)** The pages required for multiprocessor communication (numbers 252 to 255) are not or not all available.
- **(34)** The pages required for multiprocessor communication (numbers 252 to 255) are defective.
- **(35)** The parameter "automatic/manual" is illegal. The following errors are possible:

  The "automatic/manual" ID is less than 1.

  The "automatic/manual" ID is greater than 2.

- **(36)** The parameter "number of CPUs" is illegal. The following errors are possible:

  The number of CPUs is less than 2.

  The number of CPUs is greater than 4.

- **(37)** The parameter "block ID" is illegal. The following errors are possible:

  The block ID is less than 1.

  The block ID is greater than 2.

- **(38)** The parameter "block number" is illegal, since it is a data block with a special significance. The following errors are possible:

  If block ID = 1 : DB 0, DB 1, DB 2

  If block ID = 2 : DX 0, DX 1, DX 2

- **(39)** The parameter "block number" is incorrect, since the data block does not exist.
- **(40)** The parameter "start address of the assignment list" is too high or the data block is too short.
- **(41)** The assignment list in the data block is not correctly structured.
- **(42)** The sum of the assigned memory fields is greater than 48.

Errors

The "error" number group cannot occur with the INITIALIZE function.

Warning

The "warning" number group cannot occur with the INITIALIZE function.

3.2.2 Total Capacity

This parameter specifies how many of the 48 available memory fields are assigned to connections.
In the "automatic" mode, this parameter always has the value 48. In the "manual" mode, it can have a value less than 48. This means that existing memory capacity is not used.

4 SEND Function (OB 202)

Call parameters

1st data field: Before calling OB 202 you must specify the input parameters in the data field.OB 202 requires six F flag bytes in the data field for input and output parameters:

FY x + 0:	receiving CPU	input parameter
FY x + 1:	block ID	input parameter
FY x + 2:	block number	input parameter
FY x + 3:	field number	input parameter
FY x + 4:	condition code byte	output parameter
FY x + 5:	transmitting capacity	output parameter

2. ACCU-1-L: No. of the first flag byte "x" in the data field:
permitted values: ACCU-1-LH: 0
ACCU-1-LL: 0 to 246

The SEND function transfers a data block to the buffer of the 923C coordinator. It also indicates how many blocks of data can still be sent and buffered.

4.1 Input Parameters

4.1.1 Receiving CPU

The data to be sent are intended for the receiving CPU; the permitted value is between 1 and 4 but must be different from the CPU's own number.

4.1.2 Block ID and Number / Field Number

Block ID

ID = 1:	DB data block
ID = 2:	DX data block
ID = 0 or 3 to 255:	illegal, causes an error message

Block number

The block number, along with the block ID (see above) and the field number (see below) specifies the area from which the data to be sent is taken (and where it is to be stored in the receiving CPU).

Remember that certain data blocks have a special significance, for example, DB 0, DB 1 or DX 0 (see programming instructions for your CPUs). These data blocks must therefore not be used for the data transfer described here.
If you attempt to use these block numbers, the function is aborted with an error message.

Field number

The field number indicates the area in which the data to be sent is located.

Field number	Data area	
	First data word	Last data word
0	DW 0	DW 31
1	DW 32	DW 63
2	DW 64	DW 95
3	DW 96	DW 127
4	DW 128	DW 159
5	DW 160	DW 191
6	DW 192	DW 223
7	DW 224	DW 255
8	DW 256	DW 287
9	DW 288	DW 319
:	:	:
:	:	:

The following situations are possible:

- If the data block is sufficiently long, you obtain a 32-word long area as shown in the table above.
- If the end of the data block is within the selected field, an area with a length between 1 and 32 words will be transferred.
- If the first data word address is not within the length of the data block, the SEND function detects and indicates an error.

Example

Data block with a length of 80 words: DW 0 to DW 74, 5 words are required for the block header.

Field number	Data area		Length
	First data word	Last data word	
0	DW 0	DW 31	32 words
1	DW 32	DW 63	32 words
2	DW 64	DW 74	11 words
3 and higher	Incorrect parameter assignment		

4.2 Output Parameters

4.2.1 Condition Code Byte

This byte informs you whether the SEND function was executed correctly and completely.

Errors

If an error occurs, you must change the program/parameters.

Error numbers (evaluation of the condition code byte as a byte)

- **(65)** The parameter "receiving CPU" is illegal. The following errors are possible:

  The number of the receiving CPU is greater than 4.

  The number of the receiving CPU is less than 4

  The number of the receiving CPU is the same as the CPU's own number.

- **(67)** The special function organization block call is wrong. The following errors are possible

  a) Secondary error, since the INITIALIZE function could not be called or was terminated by an initialization conflict.

  b) Double call: the call for this function, SEND, SEND TEST, RECEIVE or RECEIVE TEST is illegal, since one of the functions INITIALIZE, SEND, SEND TEST, RECEIVE or RECEIVE TEST has already been called in this CPU in a lower processing level (e.g. cyclic program execution). (See "Calling and nesting the special function organization blocks".)

  c) The CPU's own number is incorrect (system data corrupted); following power down/power up the CPU number is generated again by the system program.

- **(68)** The management data (queue management) of the selected connections are incorrect; set up the buffer in the coordinator 923C again using the INITIALIZE function.

- **(69)** The parameter "block ID" is illegal. The following errors are possible:

  The block ID is less than 1.

  The block ID is greater than 2.

- **(70)** The parameter "block number" is illegal, since it is a data block with a special significance. The following errors are possible:

  If the block ID = 1 : DB 0, DB 1, DB 2

  If the block IF = 2 : DX 0, DX 1, DX 2

- **(71)** The parameter "block number" provided by the sender (RECEIVE) is incorrect. The specified data block does not exist.

- **(72)** The parameter "field number" is incorrect. The data block is too short or the field number too high.

Warning

The function could be executed; the function call must be repeated, e.g. in the next cycle.

Warning numbers (evaluation of the condition code byte as a byte)

- **(129)** The SEND function cannot transfer data, since the transmitting capacity was already zero when the function was called.

Initialization conflict

The "initialization conflict" number group cannot occur with the SEND function.

4.2.2 Transmitting Capacity

The "transmitting capacity" indicates how many blocks of data can still be sent and buffered.

5 SEND TEST Function (OB 203)

Call parameters

1. Data field:	Before calling OB 203, you must specify the input parameters in the data field. OB 203 requires 4 F flag bytes in the data field for input and output parameters:		
	FY x + 0:	receiving CPU	input parameter
	FY x + 1:	—	not used
	FY x + 2:	condition code byte	output parameter
	FY x + 3:	transmitting capacity	output parameter
2. ACCU-1-L:	No. of the first flag byte "x" in the data field, permitted values:		ACCU-1-LH: 0 ACCU-1-LL: 0 to 246

The SEND TEST function determines the number of free memory fields in the buffer of the 923C coordinator.
Depending on this number m, the SEND function can be called m times to transfer m blocks of data.

5.1 Input Parameters

5.1.1 Receiving CPU

The CPU's own number and the number of the receiving CPU identify the connection for which the transmitting capacity (see above) is determined.

5.2 Output Parameters

5.2.1 Condition Code Byte

This byte indicates whether the SEND TEST function was executed correctly and completely.

Errors

If an error occurs, you must change the program parameters.

Error numbers (evaluation of the condition code byte as a byte)

- **(65)** The parameter "receiving CPU" is illegal. The following errors are possible:

  The number of the receiving CPU is greater than 4.

  The number of the receiving CPU is less than 1.

  The number of the receiving CPU is the same as the CPU's own number.

- **(67)** The special function organization block call is wrong. The following errors are possible:

  a) Secondary error, since the INITIALIZE function could not be called or was terminated by an initialization conflict.

  b) Double call: the call for this function, SEND, SEND TEST, RECEIVE or RECEIVE TEST is illegal, since one of the functions INITIALIZE, SEND, SEND TEST, RECEIVE or RECEIVE TEST has already been called in this CPU in a lower processing level (e.g. cyclic program execution). (See "Calling and nesting the special function organization blocks".)

  c) The CPU's own number is incorrect (system data corrupted); following power down/power up the CPU number is generated again by the system program

- **(68)** The management data (queue management) of the selected connections are incorrect; set up the buffer in the coordinator 923C again using the INITIALIZE function.

Warning

The "warning" number group cannot occur with the SEND TEST function.

Initialization conflict

The "initialization conflict" number group cannot occur with the SEND TEST function.

5.2.2 Transmitting Capacity

The "transmitting capacity" parameter indicates how many blocks of data can be sent and buffered.

6 RECEIVE Function (OB 204)

Call parameters

1. Data field

Before calling OB 204, you must specify the input parameters in the data field. OB 204 requires 10 F flag bytes in the data field for input and output parameters:

FY x + 0:	transmitting CPU	input parameter
FY x + 1:	—	not used
FY x + 2:	condition code byte	output parameter
FY x + 3:	receiving capacity	output parameter
FY x + 4:	block ID	output parameter
FY x + 5:	block number	output parameter
FY x + 6:	address of the first	output parameter
FY x + 7:	received data word	output parameter
FY x + 8:	address of the last	output parameter
FY x + 9:	received data word	output parameter

2. ACCU-1-L:

No. of the first flag byte "x" in the data field,
permitted values: ACCU-1-LH: 0
ACCU-1-LL: 0 to 246

The RECEIVE function takes a block of data from the buffer of the 923C coordinator. It also indicates how many data blocks are still buffered and can still be received.
The RECEIVE function should be called in a loop until all the buffered blocks of data have been received.

6.1 Input Parameters

6.1.1 Transmitting CPU

The receive block receives data supplied by the transmitting CPU; the permitted value is between 1 and 4, but must be different from the CPU's own number.

6.2 Output Parameters

6.2.1 Condition Code Byte

This byte informs you whether the RECEIVE function was executed correctly and completely.

Errors

If an error occurs, you must change the program/parameters.

Error numbers (evaluation of the condition code byte as a byte)

- **(66)** The parameter "transmitting CPU" is illegal. The following errors are possible:

  The number of the transmitting CPU is greater than 4.

  The number of the transmitting CPU is less than 1.

  The number of the transmitting CPU is the same as the CPU's own number.

- **(67)** The special function organization block call is wrong. The following errors are possible

  a) Secondary error, since the INITIALIZE function could not be called or was terminated by an initialization conflict.

  b) Double call: the call for this function, SEND, SEND TEST, RECEIVE or RECEIVE TEST is illegal, since one of the functions INITIALIZE, SEND, SEND TEST, RECEIVE or RECEIVE TEST has already been called in this CPU in a lower processing level (e.g. cyclic program execution). (See "Calling and nesting the special function organization blocks".)

  c) The CPU's own number is incorrect (system data corrupted); following power down/power up the CPU number is generated again by the system program.

- **(68)** The management data (queue management) of the selected connections are incorrect; set up the buffer in the coordinator 923C again using the INITIALIZE function.

- **(69)** The block identifiers supplied by the transmitter is illegal. The following errors are possible

  The block ID is less than 1

  The block ID is greater than 2.

- **(70)** The block number supplied by the transmitter is illegal, since it is a data block with a special significance. The following errors are possible:

  If the block ID = 1 : DB 0, DB 1, DB 2

  If the block ID = 2 : DX 0, DX 1, DX 2

- **(71)** The block number provided by the transmitter is incorrect. The specified data block does not exist.

- **(73)** The data block is too small to receive the block of data supplied by the transmitter.

Warning

The function could be executed; the function call must be repeated, e.g. in the next cycle.

Warning numbers (evaluation of the condition code byte as a byte)

- **(130)** The RECEIVE function cannot receive data, since the receiving capacity was already zero when the function was called.

Initialization conflict

The "initialization conflict" number group cannot occur with the RECEIVE function.

6.2.2 Receiving Capacity

The "receiving capacity" parameter indicates how many blocks of data are still buffered and can still be received.

6.2.3 Block ID and Number

Block ID

ID = 1: DB data block
ID = 2: DX data block

Block number

The block number along with the block ID (see above) and the addresses of the first and last data word (see below) specifies the area in which the received data were stored by the RECEIVE function (and the area from which they were taken in the transmitting CPU by the SEND function).

Remember that the receive data blocks should be in a random access memory (RAM); using read-only memories (EPROM) might possibly serve a practical purpose for transmit data blocks.

6.2.4 Address of the First/Last Received Data Word

The difference between the addresses of the first and last data word transferred is a maximum of 31, since a maximum of 32 data words can be transferred per function call.

7 RECEIVE TEST Function (OB 205)

Call parameters

1. Data field:	Before calling OB 205, you must specify the input parameters in the data field. OB 205 requires 4 F flag bytes in the data field for input and output parameters:

FY x + 0:	transmitting CPU	input parameter
FY x + 1:	—	not used
FY x + 2:	condition code byte	output parameter
FY x + 3:	transmitting capacity	output parameter

2. ACCU-1-L:	No. of the first flag byte "x" in the data field,	
	permitted values:	ACCU-1-LH:0 ACCU-1-LL: 0 to 246

The RECEIVE TEST function determines the number of occupied memory fields in the buffer of the 923C coordinator. Depending on this number m, the RECEIVE function can be called m times to receive m blocks of data.

7.1 Input Parameters

7.1.1 Transmitting CPU

The CPU's own number and the number of the transmitting CPU identify the connection for which the receiving capacity (see above) is determined.

7.2 Output Parameters

7.2.1 Condition Code Byte

This byte indicates whether the RECEIVE TEST function was executed correctly and completely.

Errors

If an error occurs, you must change the program parameters.

Error numbers (evaluation of the condition code byte as a byte)

- **(66)** The parameter "transmitting CPU" is illegal. The following errors are possible:

  The number of the transmitting CPU is greater than 4.

  The number of the transmitting CPU is less than 1

  The number of the transmitting CPU is the same as the CPU's own number.

- **(67)** The special function organization block call is wrong. The following errors are possible:

  a) Secondary error, since the INITIALIZE function could not be called or was terminated by an initialization conflict.

  b) Double call: the call for this function, SEND, SEND TEST, RECEIVE or RECEIVE TEST is illegal, since one of the functions INITIALIZE, SEND, SEND TEST, RECEIVE or RECEIVE TEST has already been called in this CPU in a lower processing level (e.g. cyclic program execution). (See "Calling and nesting the special function organization blocks".)

  c) The CPU's own number is incorrect (system data corrupted); following power down/power up the CPU number is generated again by the system program..

- **(68)** The management data (queue management) of the selected connections are incorrect; set up the buffer in the coordinator 923C again using the INITIALIZE function.

Warning

The "warning" number group cannot occur with the RECEIVE TEST function.

Initialization conflict

The "initialization conflict" number group cannot occur with the RECEIVE TEST function.

7.2.2 Receiving Capacity

The "receiving capacity" parameter indicates how many blocks of data can be received and buffered.

8 Applications

When using one of the function blocks listed below and using interrupts (e.g. OB 2), make sure that the scratchpad flags are saved at the beginning of the interrupt handling and are written back again at the end.

REMEMBER

This also applies to the setting "interrupts at block boundaries", since the call of the special function organization blocks represents a block boundary.

8.1 Calling the Special Function OB Using Function Blocks

The following five function blocks (FB 200 and FB 202 to FB 205) contain the call for the corresponding special function organization block for multiprocessor communication (OB 200 and OB 202 to OB 205).
The numbers of the function blocks are not fixed and can be changed. The parameters of the special function OBs are transferred as actual parameters when the function blocks are called. The direct call of the special function organization blocks is faster, however, is more difficult to read owing to the absence of formal parameters.

FB no.	FB name	Function
FB 200	INITIAL	Set up buffer
FB 202	SEND	Send a block of data
FB 203	SEND-TST	Test the sending capacity
FB 204	RECEIVE	Receive a block of data
FB 205	RECV-TST	Test receiving capacity

The flag area from FY 246 to maximum FY 255 is used by the function blocks as a parameter field for the special function organization blocks.

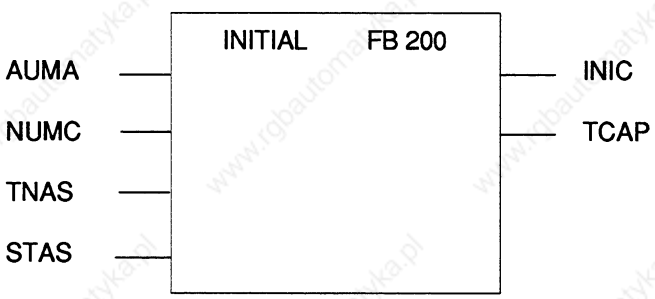
The exact significance of the input and output parameters is explained in the description of the special function organization blocks.

REMEMBER

The following examples of applications involve finished applications that you can program by copying them.

8.1.1 Setting Up a Buffer (FB 200)

Parameter name	Significance	Parameter type	Data type	Parameter field
AUMA	**Au**tomatic/**ma**nual	I	BY	FY 246
NUMC	**Num**ber of **C**PUs	I	BY	FY 247
TNAS	**T**ype (H byte) and **n**umber (L byte) of the data block containing the **as**signment list	I	W	FW 248
STAS	**St**art address of the **as**signment list	I	W	FW 250
INIC	**Ini**tialization **c**onflict	Q	BY	FY 252
TCAP	Total **cap**acity	Q	BY	FY 253

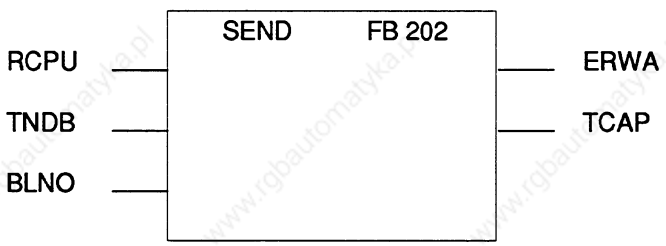


```
FB 200                                          LEN=45    ABS
SEGMENT 1
NAME:INITIAL
DECL :AUMA      I/Q/D/B/T/C:    I    BI/BY/W/D:    BY
DECL :NUMC      I/Q/D/B/T/C:    I    BI/BY/W/D:    BY
DECL :TNAS      I/Q/D/B/T/C:    I    BI/BY/W/D:    W
DECL :STAS      I/Q/D/B/T/C:    I    BI/BY/W/D:    W
DECL :INIC      I/Q/D/B/T/C:    Q    BI/BY/W/D:    BY
DECL :TCAP      I/Q/D/B/T/C:    Q    BI/BY/W/D:    BY
```

```
0017    :L    =AUMA          AUTOMATIC/MANUAL
0018    :T    FY 246
0019    :L    =NUMC          NUMBER OF CPUs
001A    :T    FY 247
001B    :L    =TNAS          DB TYPE, DB NO.
001C    :T    FW 248
001D    :L    =STAS          START ADDRESS OF THE
                             ASSIGNMENT LIST
001E    :T    FW 250
001 F   :
0020    :L    KB 246         SF OB:
0021    :JU   OB 200         INITIALIZE
0022    :
0023    :L    FY 252         INITIALIZATION CONFLICT
0024    :T    =INIC
0025    :L    FY 253         TOTAL CAPACITY
0026    :T    =TCAP
0027    :BE
```

8.1.2 Sending a Block of Data (FB 202)

Parameter name	Significance	Parameter type	Data type	Parameter field
RCPU	Receiving CPU	I	BY	FY 246
TNDB	Type (H byte) and number (L byte) of the source data block	I	W	FW 247
BLNO	Block number	I	BY	FY 249
ERWA	Error warning	Q	BY	FY 250
TCAP	Transmitting capacity	Q	BY	FY 251



```
FB 202                                          LEN=40     ABS
SEGMENT 1
NAME:SEND
DECL :RCPU      I/Q/D/B/T/C:      I     BI/BY/W/D:      BY
DECL :TNDB      I/Q/D/B/T/C:      I     BI/BY/W/D:      W
DECL :BLNO      I/Q/D/B/T/C:      I     BI/BY/W/D:      BY
DECL :ERWA      I/Q/D/B/T/C:      Q     BI/BY/W/D:      BY
DECL :TCAP      I/Q/D/B/T/C:      Q     BI/BY/W/D:      BY


0014      :L    =RCPU                   RECEIVING CPU
0015      :T    FY 246
0016      :L    =TNDB                   DB TYPE, DB NO.
0017      :T    FW 247
0018      :L    =BLNO                   BLOCK NUMBER
0019      :T    FY 249
001A      :
001B      :L    KB 246                  SF OB:
001C      :JU   OB 202                  SEND A BLOCK OF DATA
001D      :
001E      :L    FY 250                  ERROR/WARNING
001F      :T    =ERWA
0020      :L    FY 251                  TRANSMITTING CAPACITY
0021      :T    =TCAP
0022      :BE
```

8.1.3 Testing the Transmitting Capacity (FB 203)

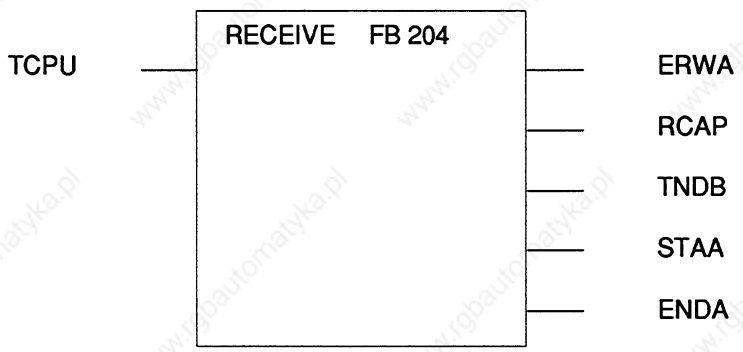
Parameter name	Significance	Parameter type	Data type	Parameter field
RCPU	Receiving **CPU**	I	BY	FY 246
ERRO	**Error**	Q	BY	FY 248
TCAP	Transmitting **cap**acity	Q	BY	FY 249

```
FB 203                                          LEN=30    ABS
SEGMENT 1
NAME:SEND-TST
DECL :RCPU      I/Q/D/B/T/C:    I    BI/BY/W/D:     BY
DECL :ERRO      I/Q/D/B/T/C:    I    BI/BY/W/D:     BY
DECL :TCAP      I/Q/D/B/T/C:    Q    BI/BY/W/D:     BY
```

```
000E    :L    =RCPU          RECEIVING CPU
000F    :T    FY 246
0010    :
0011    :L    KB 246         SF OB:
0012    :JU   OB 203         TEST TRANSMITTING CAPACITY
0013    :
0014    :L    FY 248         ERROR
0015    :T    =ERRO
0016    :L    FY 249         TRANSMITTING CAPACITY
0017    :T    =TCAP
0018    :BE
```

8.1.4 Receiving a Block of Data (FB 204)

Parameter name	Significance	Parameter type	Data type	Parameter field
TCPU	Transmitting **CPU**	I	BY	FY 246
ERWA	**Error warning**	Q	BY	FY 248
RCAP	**Receiving capacity**	Q	BY	FY 249
TNDB	**T**ype (H byte) and **n**umber (L byte) of the destination **d**ata **b**lock	Q	W	FW 250
STAA	Address of the first received data word (**start address**)	Q	W	FW 252
ENDA	Address of the last received data word (**end address**)	Q	W	FW 254

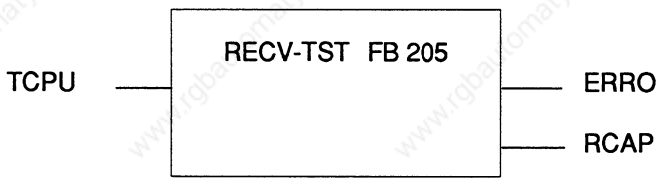


```
FB 204                                          LEN=45    ABS
SEGMENT 1
NAME:RECEIVE
DECL :TCPU      I/Q/D/B/T/C:    I    BI/BY/W/D:     BY
DECL :ERWA      I/Q/D/B/T/C:    Q    BI/BY/W/D:     BY
DECL :RCAP      I/Q/D/B/T/C:    Q    BI/BY/W/D:     BY
DECL :TNDB      I/Q/D/B/T/C:    Q    BI/BY/W/D:     W
DECL :STAA      I/Q/D/B/T/C:    Q    BI/BY/W/D:     W
DECL :ENDA      I/Q/D/B/T/C:    Q    BI/BY/W/D:     W


0017      :L    =TCPU                 TRANSMITTING CPU
0018      :T    FY 246
0019      :
001A      :L    KB 246                SF OB:
001B      :JU   OB 204                RECEIVE A BLOCK
                                      OF DATA
001C      :
001D      :L    FY 248                ERROR/WARNING
001E      :T    =ERWA
001F      :L    FY 249                RECEIVING CAPACITY
0020      :T    =RCAP
0021      :L    FW 250                DB TYPE, DB NO.
0022      :T    =TNDB
0023      :L    FW 252                START ADDRESS
0024      :T    =STAA
0025      :L    FW 254                END ADDRESS
0026      :T    =ENDA
0027      :BE
```

8.1.5 Testing the Receiving Capacity (FB 205)

Parameter name	Significance	Parameter type	Data type	Parameter field
TCPU	Transmitting **CPU**	I	BY	FY 246
ERRO	**Error**	Q	BY	FY 248
RCAP	Receiving **cap**acity	Q	BY	FY 249




```
FB 205                                          LEN=30    ABS
SEGMENT 1
NAME: RECV-TST
DECL :TCPU     I/Q/D/B/T/C:     I     BI/BY/W/D:     BY
DECL :ERRO     I/Q/D/B/T/C:     Q     BI/BY/W/D:     BY
DECL :RCAP     I/Q/D/B/T/C:     Q     BI/BY/W/D:     BY
```

```
000E    :L    =TCPU          TRANSMITTING CPU
000F    :T    FY 246
0010    :
0011    :L    KB 246         SF OB:
0012    :JU   OB 205         TEST RECEIVING CAPACITY
0013    :
0014    :L    FY 248         ERROR
0015    :T    =ERRO
0016    :L    FY 249         RECEIVING CAPACITY
0017    :T    =RCAP
0018    :BE
```

8.2 Transferring Data Blocks

8.2.1 Functional Description

The function block TRAN DAT (FB 110) transfers a selectable number of blocks of data from a data block in one CPU to the data block of the same type and same number in a different CPU. (For a description of the parameter list, function block and STEP 5 program, see the following page.)
The FB number has been selected at random and you can use other numbers.

8.2.2 Transferring a Data Block (FB 110)

The data area to be transferred is stipulated by the input parameter FIRB (= number of the first block of data to be transferred) and NUMB (= number of blocks of data to be transferred). A block of data normally consists of 32 data words. Depending on the data block length, the last block of data may be less than 32 data words.

The transfer is triggered by a positive-going edge at the start input STAR. If the output parameter REST is zero after the transfer, this means that the function block TRANDAT was able to send all the blocks of data (according to the NUMB parameter).

If, however, the REST output parameter has a value greater than zero, this means that the function block must be called again, for example in the next cycle. This means that you or the user program can only change the set of parameters (i.e. the values of all parameters) when the REST parameter indicates zero showing that the data transfer is complete.

You can call the function block TRANDAT several times with different parameters. In this case, various data areas are transferred simultaneously (interleaved in each other). The special function organization blocks for multiprocessor communication OB 202 to OB 205 can also be used "directly". This possibility is illustrated in the application example.

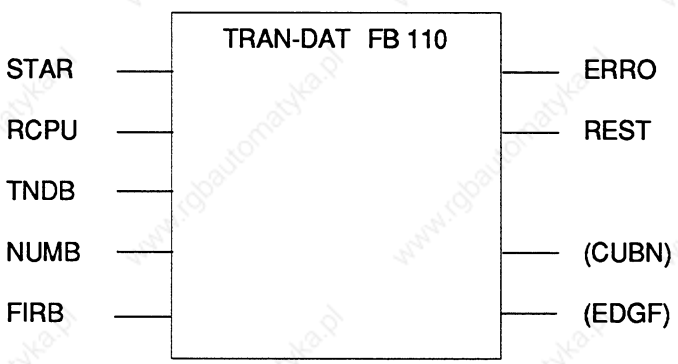
If the SEND function (OB 202) is not correctly executed within the TRANDAT function block, the error number is entered in the output parameter ERRO, the RLO = "1" and the output parameter REST is set to "0".

The TRANDAT function block uses flag bytes FY 246 to FY 251 as scratchpad flags. All other variables whose value is significant as long as the output parameter REST = "0" continue to have memory assigned to them using the mechanism of formal/actual parameters. This is necessary to allow various data blocks to be transferred simultaneously.

Note: data block and block of data are not synonymous. Data block is a DB or DX and a block of data is part of a DB from 1 to maximum 32 data words.

Parameter name	Significance	Parameter type	Data type
STAR	**Start** the transfer of the data block on a positive-going edge	I	BI
RCPU	**Receiving CPU**	I	BY
TNDB	Type (H byte) and number (L byte) of the **d**ata **b**lock to be transferred	I	W
NUMB	**Num**ber of **b**locks of data to be transferred	I	BY
FIRB	Number of the **first b**lock of data to be transferred	I	BY
ERRO	**Error**	Q	BY
REST	Number of blocks of data still to be transferred	Q	BY
CUBN [1)]	**C**urrent **b**lock **n**umber	Q	BY
EDGF [1)]	**Edge** flag	Q	BI

1) Internal scratchpad flag, not intended for evaluation.



```
FB 110                                              LEN=89     ABS
SEGMENT 1
NAME:TRAN-DAT
DECL :STAR      I/Q/D/B/T/C:    I    BI/BY/W/D:     BI
DECL :RCPU      I/Q/D/B/T/C:    I    BI/BY/W/D:     BY
DECL :TNDB      I/Q/D/B/T/C:    I    BI/BY/W/D:     W
DECL :NUMB      I/Q/D/B/T/C:    I    BI/BY/W/D:     BY
DECL :FIRB      I/Q/D/B/T/C:    I    BI/BY/W/D:     BY
DECL :ERRO      I/Q/D/B/T/C:    Q    BI/BY/W/D:     BY
DECL :REST      I/Q/D/B/T/C:    Q    BI/BY/W/D:     BY
DECL :CUBN      I/Q/D/B/T/C:    Q    BI/BY/W/D:     BY
DECL :EDGF      I/Q/D/B/T/C:    Q    BI/BY/W/D:     BI


0020        :L    =RCPU          ASSIGN PARAMETER FIELD FOR
0021        :T    FY 246         SF OB 202
0022        :L    =TNDB
0023        :T    FW 247
0024        :
0025        :L    =REST          FIRST SEND ANY REMAINING
0026        :L    KB 0           BLOCKS OF DATA
0027        :><F
0028        :JC   =TRAN
0029        :
002A        :AN   =STAR          POSITIVE EDGE AT START
002B        :RB   =EDGF          INPUT ?
002C        :ON   =STAR
002D        :O    =EDGF
002E        :JC   =GOOD
002F        :S    =EDGF
0030        :
0031        :L    =NUMB          INITIALIZE THE GLOBAL FLAGS
0032        :T    =REST          AFTER POSITIVE EDGE AT
0033        :L    =FIRB          START INPUT
0034        :T    =CUBN
0035        :
0036        :L    =REST          AS LONG AS REST ><0,
0038 LOOP   :L    KF+0           CONTINUE TO ATTEMPT
0039        :!=F                 TO SEND BLOCKS OF DATA
003A        :JC   =GOOD
003B TRAN   :L    =CUBN
003C        :T    FY 249
003D        :L    KB 246         SF OB:
003E        :JU   OB 202         SEND A BLOCK OF DATA
003F        :L    FY 250
0040        :JM   =ERRO          ABORT IF ERROR
0041        :JP   =GOOD          ABORT IF TRANS-CAP = 0
0042        :L    =CUBN          INCREMENT BLOCK NUMBER
0043        :I    1
0044        :T    =CUBN
0045        :L    =REST          DECREMENT NUMBER OF
0046        :D    1              REMAINING BLOCKS OF DATA
```

```
0047        :T    =REST
0048        :JU   =LOOP
0049        :
004A GOOD:A       F 0.0          REGULAR END OF PROGRAM
004B        :AN   F 0.0
004C        :L    KB 0           RLO = 0, ERRO = 0
004D        :T    =ERRO
004E        :BEU
004F        :
0050 ERRO :T      =ERRO          PROGRAM END IF ERROR
0051        :L    KB 0
0052        :T    =REST          RLO = 1, ERROR CONTAINS
                                 ERROR NUMBER
0053        :BE
```

8.2.3 Application Example (for the S5-135U)

You want CPU 1 to transfer data blocks DB 3 (blocks of data 2 to 5) and DB 4 (blocks of data 1 to 3) to CPU 2 during the cyclic user program. The RECEIVE function (OB 204) is also called in the cyclic user program.

The following blocks must be loaded in the individual CPUs:

Function	CPU 1	CPU 2
Cold restart block	OB 20	—
Cycle block [1)]	FB 0	FB 0
Send DB	DB 3; DB 4	—
Receive DB	—	DB 3; DB 4

[1)] Only OB 1 is permitted as the cycle block in the CPU 946/947.

OB 20 calls the INITIALIZE function (OB 200) and reserves several memory fields for the connection from CPU 1 to CPU 2.

The cyclic user program in function block FB 0 of CPU 1 contains two calls for the function block TRANDAT in each case with different sets of parameters. The transfer of the first data block DB 3 begins after a positive edge after input I 2.0. A positive edge at input I 2.1 starts the transfer of the second data block DB 4.

```
FB 0                                        LEN=66      ABS
SEGMENT 1
NAME:DEMO



0005        :L     KB 2                     TO CPU 2 ..
0006        :T     FY 0
0007        :L     KY 1,3                   .. FROM DATA BLOCK DB 3
0009        :T     FW 1
000A        :L     KB 4                     .. FOUR BLOCKS OF DATA
000B        :T     FY 3
000C        :L     KB 2                     .. SEND FROM 2ND BLOCK OF DATA
000D        :T     FY 4
000E        :
000F        :JU    FB 110
0010 NAME   :TRAN-DAT
0011 STAR   :      I 2.0
0012 RCPU   :      FY 0
0013 TNDB   :      FW 1
0014 NUMB   :      FY 3
0015 FIRB   :      FY 4
0016 ERRO   :      FY 5
0017 REST   :      FY 6
0018 CUBN   :      FY 7
0019 EDGF   :      F 8.0
001A        :
001B        :
001C        :JC    =HALT                    ABORT AFTER ERROR
001D        :
001E        :L     KB 2                     TO CPU 2 ..
001F        :T     FY 10
0020        :L     KY 1,4                   .. FROM DATA BLOCK DB 4
0022        :T     FW 11
0023        :L     KB 3                     .. THREE BLOCKS OF DATA
0024        :T     FY 13
0025        :L     KB 1                     .. SEND FROM 1ST BLOCK OF DATA
0026        :T     FY 14
0027        :
0028        :JU    FB 110
0029 NAME   :TRAN-DAT
002A STAR   :      I 2.1
002B RCPU   :      FY 10
002C TNDB   :      FW 11
002D NUMB   :      FY 13
002E FIRB   :      FY 14
002F ERRO   :      FY 5
0030 REST   :      FY16
0031 CUBN   :      FY17
0032 EDGF   :      F 8.1
0033        :
0034        :
0035        :JC    =HALT                    ABORT AFTER ERROR
0036        :BEU
```

```
0037        :
0038 HALT   :
```

The error handling takes place here (e.g stop, message output on the printer, ...)

```
0039        :BE
```

In CPU 2, the RECEIVE function (OB 204) called by FB 0 enters each transmitted block of data into the appropriate data block. It may take several cycles before a data block has been completely received.

```
FB 0                                        LEN=26     ABS
SEGMENT 1
NAME:RECV-DAT


0005        :L     KB 1                     RECEIVE DATA FROM CPU 1
0006        :T     FY 246
0007        :
0008 LOOP   :L     KB 246                   SF OB:
0009        :JU    OB 204                   RECEIVE
000A        :JM    =ERRO                    ABORT IF ERROR
000B        :L     FY 249                   THE RECEIVE FUNCTION IS
000C        :L     KB 0                     CALLED UNTIL THERE ARE NO
000D        :><F                            FURTHER BLOCKS OF DATA IN
000E        :JC    =LOOP                    THE BUFFER, I.E. THE RECEIVING
000F        :                               CAPACITY = 0.
0010        :BEU
0011 ERRO   :
```

The error handling takes place here (e.g. stop, message output on printer, ...)

```
0012        :BE
```

8.3 Extending the IPC Flag Area

8.3.1 The Problem

In the multiprocessor programmable controllers S5-135U and S5-155U, each of the 256 flag bytes of a CPU can become an input or output IPC flag by making an entry in data block DB 1. This, however, reduces the number of "normal" flag bytes. To transfer a data record (several bytes) other mechanisms are also required (semaphore variable or DX 0 parameter assignment "transfer IPC flags as a block") are necessary to prevent the receiver from receiving a fragmented data record.

8.3.2 The Solution

Consecutive data words of a DB or DX data block are defined from DW 0 onwards as "IPC data words". Each connection is assigned its own data block and is totally independent of the other connections.

At the beginning of the cycle block (CPU 946/947: OB 1, CPU 92x: OB 1 or FB 0), the IPC data words are received with the aid of the special function organization blocks for multiprocessor communication. This is followed by the "regular" cyclic program, that evaluates the received data and generates the data to be sent. At the end of the cycle, this data is then sent with the aid of the special organization blocks for multiprocessor communication. It can therefore be received by the other CPUs at the beginning of their cycles.

The following applies for each of the maximum 12 possible connections regardless of the other connections:

- The transmitting CPU is only active when the receiving CPU has read out all the "old" data from the 923C buffer.
- The receiving CPU is only active when the transmitting CPU has written all the "new" data in the 923C buffer.

This means that the receiving CPU can either receive a complete new data record or the old data record remains unchanged: **no mixing of "old" and "new" data**.

8.3.3 Data Structure

Which data words (for the data word area below) are to be transferred from which CPU to which CPU is described in the connection list (see table on the following page). This is located in an additional data block that must exist in all the CPUs involved.

The data word areas always begin from data word DW 0, and their lengths are specified in blocks of data. Remember the following points:

- A complete block of data consists of 32 data words.
- If the last block of a data block is "truncated", i.e. it contains between 1 and 31 data words, less data words are transferred.
- If a send data block is longer than the number of blocks of data specified in the connection list, the excess data words can be used in the corresponding CPU.
- If a receive data block is longer than the received data word area, the excess data words can be used in the corresponding CPU.

Structure of the connection list

	SUB-LIST 1			SUB-LIST 2		
Connection		DB type	DB number			No.of blocks of data
from CPU 1 to ...	DW 0	S 1		DW 16	S 1	
... CPU 2	DW 1	...	...	DW 17	2	...
... CPU 3	DW 2	...	...	DW 18	3	...
...CPU 4	DW 3	...	...	DW 19	4	...
from CPU 2 to ...	DW 4	S 2		DW 20	S 2	
... CPU 1	DW 5	...	...	DW 21	1	...
... CPU 3	DW 6	b	c	DW 22	3	a
... CPU 4	DW 7	...	...	DW23	4	...
from CPU 3 to ...	DW 8	S 3		DW 24	S 3	
... CPU 1	DW 9	...	...	DW 25	1	...
... CPU 2	DW 10	...	...	DW 26	2	...
... CPU 4	DW 11	...	...	DW 27	4	...
from CPU 4 to ...	DW 12	S 4		DW 28	S 4	
... CPU 1	DW 13	...	...	DW 29	1	...
... CPU 2	DW 14	...	...	DW 30	2	...
... CPU 3	DW 15	...	...	DW 31	3	...
		2^{15}	2^{0}	2^{15}		2^{0}

The connection consists of two similarly structured sub-lists, each with 16 data words. For each of the four sender CPUs (S1, S2, S3, S4) three entries are required to describe a connection.

- **Number of blocks of data**

  The number of blocks of data specifies the size (= the number of data words) of the data word area to be transferred. (If connections do not exist or you do not require them, enter 0 for the number of blocks of data, and for the DB type and DB number.)

- **DB type**

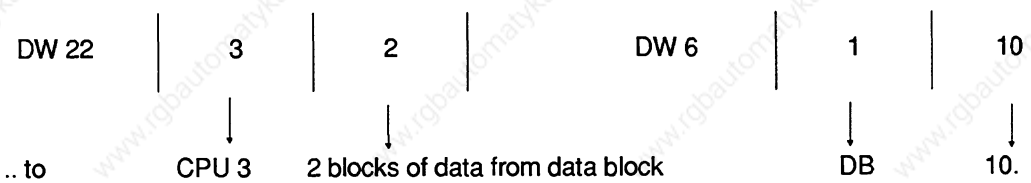
  Type of data block containing the data word area to be transferred.

- **DB number**

  Number of the data block containing the data word area to be transferred.

As shown in the table, these entries can be read in and completed in lines. If, for example, you want to transfer the first two blocks of data in data block DB 10 from CPU 2 (S2) to CPU 3, make the following entries:

CPU 2 (**S 2**) sends ..



Sub-list 2 is identical to the assignment ("manual" mode) required for the INITIALIZE function (OB 200). Within the data block, sub-list 1 must occupy data words 0 to 15 and sub-list 2 data words 16 to 31. You must not alter the entries shown in bold face.

8.3.4 Program Structure

During restart, one of the CPUs calls the INITIALIZE function (OB 200) to reserve exactly the same number of coordinator memory fields per connection as blocks of data to be transmitted on this connection.

To send and receive data word areas, each CPU uses two function blocks:

FB no.	Name	Function
FB 100	SEND-DAT	Send data word areas to the other CPUs
FB 101	RECV-DAT	Receive data word areas from the other CPUs

These FB numbers have been selected at random and you can use others.

The function blocks SEND-DAT and RECV-DAT read the connection list to determine which data word areas are to be sent from or received by which data blocks. The **whole** data word area is always sent or received. If this is not possible owing to insufficient transmitting or receiving capacity, the send or receive function is not executed.

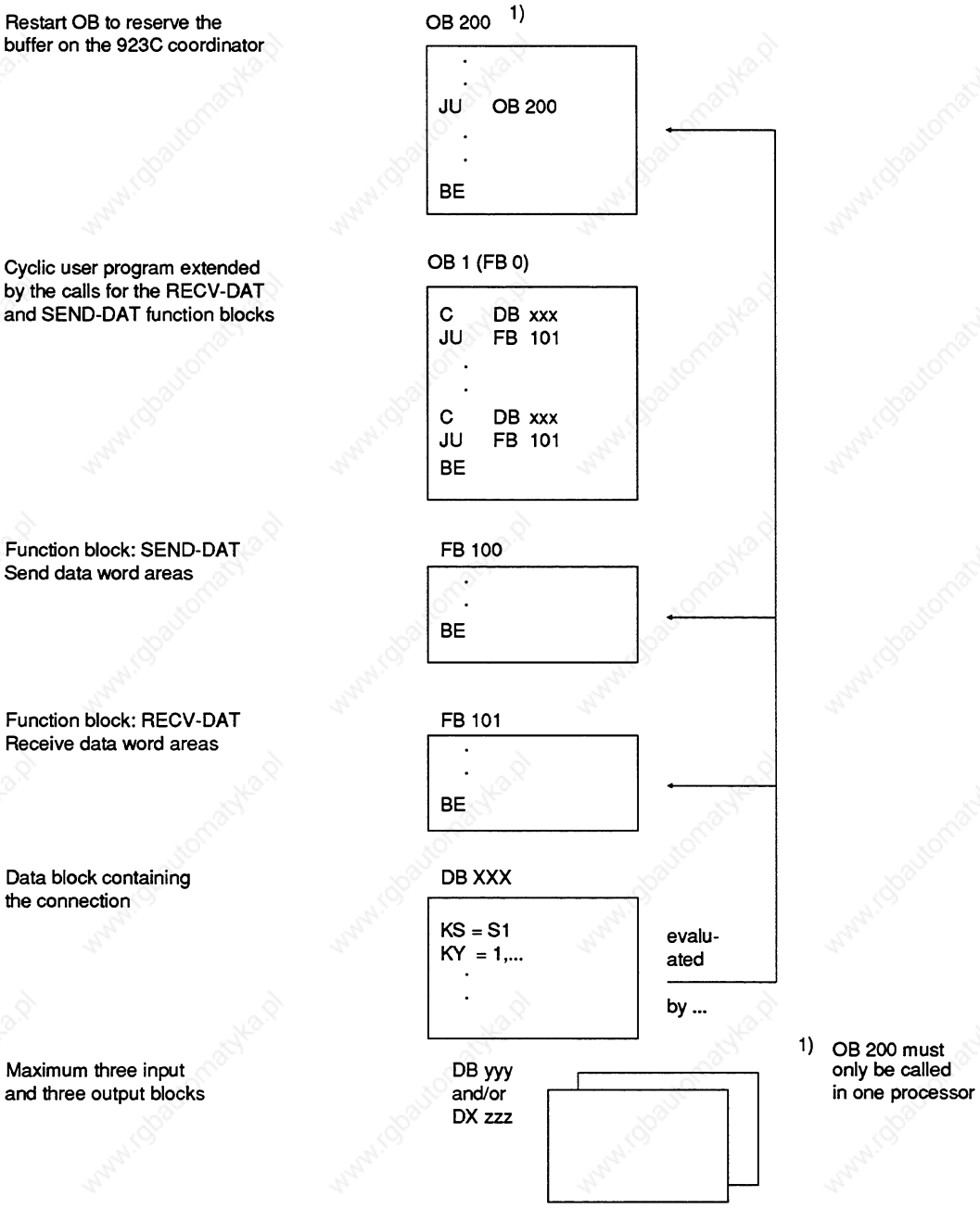


Fig. 3 Overview of the blocks required in each CPU

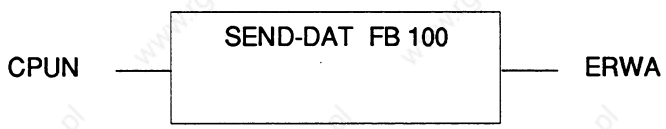
REMEMBER

The function blocks SEND-DAT and RECV-DAT contain the special function organization blocks for multiprocessor communication OB 202 to OB 205. You cannot call these organization blocks outside SEND-DAT / RECV-DAT.

8.3.5 Sending Data Word Areas (FB 100)

Before you call FB 100, the data block containing the connection list must be open. The function block SEND-DAT requires the number of the CPU on which it is called in order to evaluate the information contained in the connection list.
If the SEND function (OB 202) is not executed correctly in the function block, the error or warning number is transferred to the output parameter ERWA and RLO is set to 1.
If the input parameter CPUN (CPU number) is illegal, ERWA has the value 16 (bit $2^4 = 1$).
The function block SEND-DAT uses flag bytes FY 239 to FY 251 as scratchpad flags.

Parameter name	Significance	Parameter type	Data type
CPUN	Number of the **CPU** on which FB 100 is called. The numbers 1 to 4 are permitted.	D	KF
ERWA	Error/warning (see SEND function (OB 202))	Q	BY



```
FB 100                                          LEN=90
SEGMENT 1              0000
NAME:SEND-DAT
DECL :CPUN      I/Q/D/B/T/C:      D     KM/KH/KY/KS/KF/KT/KC/KG:   KF
DECL :ERWA      I/Q/D/B/T/C:      Q     BI/BY/W/D:          BY


000B        :LW   =CPUN                   CPUN = CPUN -1
000C        :L    KB 1                    ERROR IF:
000D        :-F
000E        :JM   =ERWA                   CPU NO. <1
000F        :L    KB 3
0010        :>F
0011        :JC   =ERWA                   CPU NO. >4
0012        :TAK
0013        :
0014        :SLW        2                 CPUN = CPUN * 4
0015        :T    FY 245                  BASE ADDRESS
0016        :
0017        :L    KB 1
0018        :T    FY 244                  CONNECTION COUNTER
0019        :
001A LOOP   :L    FY 245                  BASE ADDRESS
001B        :L    FY 244                  + COUNTER
001C        :+F
001D        :T    FW 240
001E        :ADD  BN+16                   + OFFSET
001F        :T    FW 242
0020        :
0021        :DO   FW 242
0022        :L    DR 0                    NUMBER OF RESERVED
0023        :T    FY 239                  FIELDS = 0 ?
0024        :L    KB 0
0025        :!=F
0026        :JC   =EMPT
0027        :
0028        :DO   FW 242
0029        :L    DL 0                    NO. OF THE RECEIVING CPU
002A        :T    FY 246
002B        :L    KB 246                  SF OB:
002C        :JU   OB 203                  TEST TRANSMITTING CAPACITY
002D        :L    FY 248                  ABORT IF ERROR
002E        :JC   =OBER
002F        :
0030        :L    FY 249                  TRANSMITTING CAPACITY >< NO.
0031        :L    FY 239                  OF RESERVED FIELDS ?
0032        :><F
0033        :JC   =EMPT
0034        :
0035        :L    KB 0                    FIELD COUNTER
0036        :T    FY 249
0037        :
0038        :DO   FW 240
0039        :L    DW 0                    TYPE AND NUMBER OF THE
003A        :T    FW 247                  SOURCE DB
```

```
003B        :
003C TRAN   :L    KB 246        SF OB:
003D        :JU   OB 202        SEND A BLOCK OF DATA
003E        :L    FY 250        ABORT IF ERROR/WARNING
003F        :JC   =OBER
0040        :
0041        :L    FY 249        BLOCK NO. = BLOCK NO. + 1
0042        :I    1
0043        :T    FY 249        ALL BLOCKS OF DATA TRANSFERRED ?
0044        :L    FY 239
0045        :<F
0046        :JC   =TRAN
0047        :
0048 EMPT   :L    FY 244        INCREMENT
0049        :I    1             CONNECTION COUNTER
004A        :T    FY 244
004B        :L    KB 4          ALL THREE CONNECTIONS
004C        :<F                 PROCESSED ?
004D        :JM   =LOOP
004E        :L    KB 0          REGULAR PROGRAM END:
004F        :T    =ERWA         RLO = 0, ERWA = 0
0050        :BEU
0051        :
0052 ERWA   :L    KB 16         PROGRAM END IF ERROR:
0053 OBER   :T    =ERWA         RLO = 1, ERWA CONTAINS
0054        :BE                 ERROR/WARNING NUMBER
```

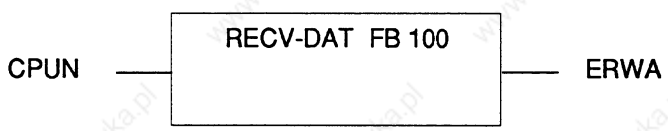
8.3.6 Receive Data Word Areas (FB 101)

Before you call FB 101, the data block containing the connection list must already be open. The function block RECV-DAT requires the number of the CPU in which it is called in order to evaluate the information contained in the connection list.

If the RECEIVE function (OB 204) is not correctly processed within the function block, the corresponding error or warning number is transferred to the output parameter ERWA and the RLO is set to 1. If the input parameter CPUN is illegal, ERWA has the value 16 (bit $2^4 = 1$).

The RECV-DAT function block uses flag bytes FY 242 to FY 255 as scratchpad flags.

Parameter name	Significance	Parameter type	Data type
CPUN	Number of the **CPU** on which FB 101 is called. The numbers 1 to 4 are permitted.	D	KF
ERWA	**Error/warning** (see RECEIVE function (OB 204)).	Q	BY



```
FB 101                                                    LEN=88
SEGMENT 1              0000
NAME:RECV-DAT
DECL :CPUN      I/Q/D/B/T/C:      D    KM/KH/KY/KS/KF/KT/KC/KG:   KF
DECL :ERWA      I/Q/D/B/T/C:      Q    BI/BY/W/D:          BY


000B        :LW   =CPUN                    ERROR IF:
000C        :L    KB 1
000D        :<F
000E        :JC   =ERWA                    CPU NO.<1
000F        :LW   =CPUN
0010        :L    KB 4
0011        :>F
0012        :JC   =ERWA                    CPU NO.>4
0013        :
0014        :L    KB 1                     CONNECTION COUNTER
0015        :T    FY 242
0016        :
0017        :L    KB 16
0018        :T    FW 244                   POINTER TO SUB-LIST 2
0019        :
001A SRCH   :L    FW 244                   SEARCH SUB-LIST 2 UNTIL THE
001B        :I    1                        NEXT ENTRY FOR THE RECEIVING
001C        :T    FW 244                   CPU WITH THE NUMBER "CPUN"
001D        :DO   FW 244                   IS FOUND
001E        :L    DL 0
001F        :LW   =CPUN
0020        :><F
0021        :JC   =SRCH
0022        :
0023        :DO   FW 244
0024        :L    DR 0                     NUMBER OF RESERVED
0025        :T    FY 243                   MEMORY FIELDS = 0 ?
0026        :L    KB 0
0027        :!=F
0028        :JC   =EMPT
0029        :
002A        :L    FW 244                   DETERMINE THE NUMBER OF THE
002B        :L    KM 00000000 00001100     TRANSMITTING CPU FROM THE
002D        :AW                            POINTER TO SUB-LIST 2
002E        :SRW 2
002F        :I    1
0030        :T    FY 246
0031        :
0032        :L    KB 246                   SF OB:
0033        :JU   OB 205                   TEST RECEIVING CAPACITY
0034        :L    FY 248
0035        :JC  = OBER                    ABORT IF ERROR
0036        :
0037        :L    FY 249                   RECEIVING CAPACITY = NUMBER
0038        :L    FY 243                   OF RESERVED MEMORY FIELDS?
0039        :><
003A        :JC  =EMPT
003B        :
```

```
003C RECV :L    KB 246      SF OB:
003D      :JU   OB 204      RECEIVE A BLOCK OF
003E      :L    FY 248      DATA
003F      :JM   =OBFE       ABORT IF ERROR/
0040      :                 WARNING
0041      :L    FY 249      IF RECEIVING CAPACITY = 0,
0042      :L    KB 0        PROCESS NEXT
0043      :><F              CONNECTION
0044      :JC   =RECV
0045      :
0046 EMPT :L    FY 242      INCREMENT
0047      :I    1           CONNECTION COUNTER
0048      :T    FY 242
0049      :L    KB 4        ALL CONNECTIONS
004A      :<F               PROCESSED ?
004B      :JM   =SRCH
004C      :L    KB 0        REGULAR PROGRAM END:
004D      :T    =ERWA       RLO = 0, ERWA = 0
004E      :BEU
004F      :
0050 ERWA :L    KB 16       PROGRAM END IF ERROR:
0051 OBER :T    =ERWA       RLO = 1, ERWA CONTAINS
0052      :BE               ERROR/WARNING NUMBER
```

8.3.7 Application Example (for S5-135U)

You want to exchange data between three CPUs:

- **From CPU 1 to CPU 2:** data block DB 3, DW 0 to DW 127 (= 4 blocks of data)
- **From CPU 1 to CPU 3:** data block DX 4, DW 0 to DW 63 (= 2 blocks of data)
- **From CPU 2 to CPU 1 and CPU 3:** data block DB 5, DW 0 to DW 95 (= 3 blocks of data)

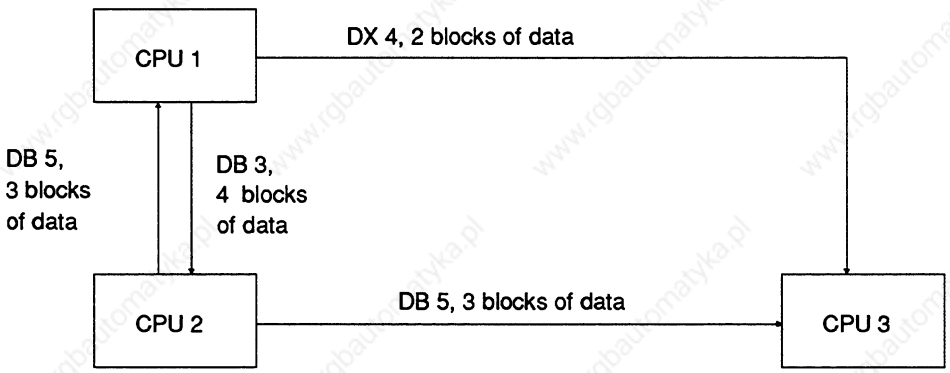


Fig. 4 Data exchange between 3 CPUs

Function block FB 0 is the interface for the cyclic user program on all three CPUs. CPU 1 calls the INITIALIZE function (OB 200) during the cold restart. The connection list is in data block DB 100.

The following blocks must be loaded in the individual CPUs:

Function	CPU 1	CPU 2	CPU 3
Restart OB	OB 20	—	—
User program	FB 0	FB 0	FB 0
FB: SEND-DAT	FB 100	FB 100	FB 100
FB: RECV-DAT	FB 101	FB 101	FB 101
Connection list	DB 100	DB 100	DB 100
Input DB	DB 5	DB 3	DB 5; DX 4
Output DB	DB 3; DX 4	DB 5	—

First of all the connection list (structure described in the section "Data structure") must be written and entered in DB 100:

```
DB100                                   LEN=37     ABS
                                              PAGE   1


—— Sub-list 1 ——

 0 :  KS   =S1                          Send from CPU 1 to ..
 1 :  KY   =001,003;                    .. CPU 2 (DB 3)
 2 :  KY   =002,004;                    .. CPU 3 (DX 4)
 3 :  KY   =000,000;
 4 :  KS   =S2                          Send from CPU 2 to ..
 5 :  KY   =001,005;                    .. CPU 1 (DB 5)
 6 :  KY   =001,005;                    .. CPU 3 (DB 5)
 7 :  KY   =000,000;
 8 :  KS   =S3
 9 :  KY   =000,000;
10 :  KY   =000,000;
11 :  KY   =000,000;
12 :  KS   =S4
13 :  KY   =000,000;
14 :  KY   =000,000;
15 :  KY   =000,000;

—— Sub-list 2——

16 :  KS   =S1                          Send from CPU 1 to ..
17 :  KY   =002,004;                    .. CPU 2 (four blocks of data)
18 :  KY   =003,002;                    .. CPU 3 (two blocks of data)
19 :  KY   =004,000;
20 :  KS   =S2                          Send from CPU 2 to ..
21 :  KY   =001,003;                    .. CPU 1 (three blocks of data)
22 :  KY   =003,003;                    .. CPU 3 (three blocks of data)
23 :  KY   =004,000;
24 :  KS   =S3
25 :  KY   =001,000;
26 :  KY   =002,000;
27 :  KY   =004,000;
28 :  KS   =S4
29 :  KY   =001,000;
30 :  KY   =002,000;
31 :  KY   =003,000;
```

Data words DW 16 to DW 31 contain the assignment list required for the manual INITIALIZATION function (OB 200). OB 200 is called by the OB 20 shown below in CPU 1 during the restart.

```
OB 20                                    LEN=23     ABS
SEGMENT 1


0000    :L    KB 2                       MANUAL INITIALIZATION OF
0001    :T    FY 246                     THE PAGES
0002    :
0003    :L    KY 1,100                   THE ASSIGNMENT LIST IS ENTERED
0005    :T    FW 248                     IN DB 100 FROM DATA WORD 16
0006    :L    KF+16                      ONWARDS
0008    :T    FW 250
0009    :
000A    :L    KB 246                     SF OB:
000B    :JU   OB 200                     INITIALIZE
000C    :
000D    :AN   F 252.5                    BLOCK END IF THERE IS NO
000E    :BEC                             INITIALIZATION CONFLICT
000F    :
```

The error handling routine is inserted here if an initialization conflict occurs (e.g. stop, output message on printer etc.)

```
0010    :BE
```

The user program on each CPU is extended by the RECV-DAT and SEND-DAT call. Function block FB 0 shown below is for CPU 1. For the other CPUs, the input parameter CPUN (CPU number) must be modified.

```
FB0                                    LEN=31    ABS
SEGMENT 1
NAME:PROG-1


0005        :C    DB100          CONNECTION LIST        DB 100
0006        :JU   FB101          RECEIVE THE INPUT
                                 DATA BLOCKS
0007 NAME   :RECV-DAT
0008 CPUN   :     KF+1
0009 ERWA   :     FY0
000A        :JC   =ERWA          ABORT IF ERROR/WARNING
000B        :
000C        :
```

The cyclic user program is inserted here and reads data from the input data blocks and writes data to the output data blocks.

```
000D        :
000E        :
000F        :
0010        :C    DB100          CONNECTION LIST        DB 100
0011        :JU   FB100          SEND THE OUTPUT
                                 DATA BLOCKS
0012 NAME   :SEND-DAT
0013 CPUN   :     KF+1
0014 ERWA   :     FY0
0015        :JC   =ERWA          ABORT IF ERROR/WARNING
0016        :BEU
0017        :
0018 ERWA   :                    AFTER ERROR/WARNING
0019        :BE                  EXECUTE ERROR HANDLING
```

The error handling is inserted here, (e.g. stop, output error message on printer or monitor, etc.)

REMEMBER

This example (IPC flag extension using function blocks SEND-DAT and RECV-DAT) can only be performed correctly if the special function organization blocks for multiprocessor communication OB 202 to OB 205 are not called outside these function blocks in any of the CPUs.

SIEMENS

SIMATIC S5
S5 135 U Programmable Controller
S Processor

Programming instructions Order No. C79000-B8576-C264-03

Contents

		Page
1	**Explanatory notes**	**3**
1.1	Application	3
1.2	STEP 5 programming language	3
1.3	Programming	4
1.3.1	Program structure	4
1.3.2	Program organization	6
1.3.3	Program storage	7
1.3.4	Program execution	7
1.4	Programming in multiprocessor operation	8
1.4.1	Flags for inter-processor communication (IPC)	9
1.4.2	Program distribution	9
1.4.3	Assignment of I/O's and IPC flags	10
1.5	General notes	13
1.5.1	Runtime optimization of the user progam	13
2	**Program blocks**	**15**
2.1	Programming program blocks	15
2.2	Calling program blocks	16
3	**Data blocks**	**17**
3.1	Programming data blocks	17
3.2	Calling data blocks	18
4	**Function blocks**	**19**
4.1	General	19
4.2	The structure of function blocks	20
4.3	Calling function blocks and parameter assignment	21
4.4	Programming function blocks	23
5	**Organization blocks**	**29**
5.1	General	29
5.2	Commissioning	31
5.2.1	Stop status	33
5.2.2	Overall reset	34
5.2.3	Test operation	34
5.3	Programming the start-up characteristics	35
5.4	Programming the cyclic execution	38
5.5	Programming interrupt-driven processing	41
5.6	Programming time-driven processing	43
5.7	Evaluating a device or programming error	44
6	**STEP 5 command set with programming examples**	**51**
6.1	General rules	51
6.2	Basic operation set	56
6.3	Supplementary operation set	78
7	**System program special functions**	**91**
7.1	Transmit data block (OB 255)	91
7.2	PID controller (OB 250 and OB 251)	91
7.2.1	PID algorithm	94
7.2.2	Data blocks for the PID controller	95
7.2.3	Initialization and call up of the PID controller in the STEP 5 program	99
7.2.4	Format of controller inputs and outputs	100
7.2.5	General notes	101
7.2.6	Controller characteristic quantities	102
7.2.7	Binary/decimal fractions	103
7.2.8	Abbreviations	104
7.3	Shift register	105
7.3.1	Mode of operation	105
7.3.2	Programming the shift register in the user program (OB 241 to 248)	108
7.3.3	Enabling the shift register memory areas (OB 221)	110
7.4	Triggering the scan time (OB 222)	110
7.5	Expanding the sign from a 16 to 32-bit fixed point number (OB 220)	110
7.6	Comparing start-up modes in multi-processor operation (OB 223)	110
7.7	Reading the cross-check sum of the system program EPROM (OB 227)	110
7.8	Block transfer of the IPC flags (OB 224)	111
7.9	Assigning parameters to the start-up characteristics (OB 225)	111
8	**STEP 5 special commands**	**112**
8.1	Generate data block	112
8.2	Set/enable semaphore	112
9	**Overview of STEP 5 operations**	**113**
10	**Information about disturbances**	**116**
	Appendix	**121**

These programming instructions describe the functional scope of the S processor 6ES5 921-3UA11, release 16 and 6ES5 921-3UA12, release 4.

With earlier releases, some functions may be limited.

Abbreviations

ACCU 1(2)-L(H)	Accumulator 1(2) low value (high value)
BARB	Program check
BARBEND	Finish program check
BCD	Binary coded decimal
COR	Coordination module (coordinator)
CP	Communications processor
CPU	Central processing unit
CSF	Control system flowchart
DB	Data block
DX	Extended data block
FB	Function block
FX	Extended function block
IP	Intelligent I/O modules
ISTACK	Interrupt stack
LAD	Ladder diagram
OB	Organization block
PC	Programmable controller
PI	Process image
PII	Process image inputs
PIO	Process image outputs
PG	Programmer
RLO	Result of logic operation
SB	Sequence block
STL	Statement list

Further reading

The following manuals contain an introduction to programming with STEP 5 and using standard function blocks:

Programming logic controls with STEP 5
Volume 1, Programming basic functions
Siemens AG, ISBN 3-8009-1407-7
Volume 2, Using standard function blocks
Siemens AG, ISBN 3-8009-1373-9
Volume 3, Programming function blocks yourself
Siemens AG, ISBN 3-8009-1366-6

1 Explanatory notes

1.1 Application

The S5 135 U programmable controller (PC) with a programmable memory is a high performance multiprocessor device for process automation (open loop control, signalling, monitoring, closed-loop control, logging). It can be used both to create the simplest logic controls with binary signals and to solve complex automation tasks. Its user programs are created with the programming language STEP 5.

The central controller of the S5 135 U can be equipped by the user with:

- **one** central processing unit (CPU) for single processor operation or
- one coordinator (COR) and up to 4 CPU's for multiprocessor operation:
- and also up to 8 communications processors (CP's) for single processor operation.

The remaining unoccupied module locations are available for input and output modules. In order to extend the peripherals, expansion units can be connected to the central controller.

In a multiprocessor PC, each individual CPU processes the user program transferred to it in a memory module, independent of the other CPU's. The COR manages the data traffic on the S5 bus. Inter-processor communication (IPC) flags for the data exchange between the individual CPU's are available on the COR (see section 1.4.1).

1.2 STEP 5 programming language

The use of the STEP 5 programming language makes it possible to program functions ranging from simple binary logic to complex digital processing and basic arithmetic operations.

The program can be written using any of three methods of representation: control system flowchart (CSF), ladder diagram (LAD) and statement list (STL). This means the programming method can be adapted to the particular application. The machine code generated by the programmers (PG's) is identical for all three methods of representation. If certain programming rules are followed, the PG can translate the user program from one method of representation to another.

Commands from the extended operation set can only be programmed in function blocks and are only shown in STL.

1.3 Programming

1.3.1 Program structure

The complete program of a PC consists of the system program and the user program. The system program contains all statements and declarations for internal functions (e.g. saving data in the event of a power failure, prompting operator reactions in particular situations etc.). This program is an integral part of the PC (EPROM) and cannot be changed by the user.

The user program consists of all statements and declarations programmed by the user for signal processing, through which the system (process) to be controlled will be influenced according to the automation task.

The S5 135 U enables the user to carry out structured programming, i.e., the complete program is divided into individual self-contained program sections (blocks). This method has the following advantages for the user:

- simple and clear programming, even of large programs,
- program sections can be standardized,
- simple program organization,
- easy program modification,
- simple program testing,
- simple commissioning.

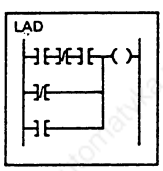
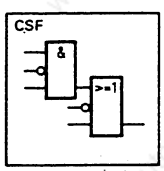
Ladder diagram	Statement list	Control system flowchart
Programming with graphic symbols as in circuit diagram to DIN 19239 (draft)	Programming with mnemonics of the function designation to DIN 19239 (draft)	Programming with graphic symbols to IEC 117-15 DIN 40700 DIN 40719 DIN 19239 (draft)
		

Fig. 1 Methods of representation in the STEP 5 programming language

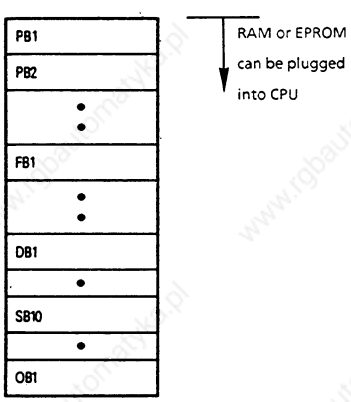


Fig. 2 Filing the blocks in the program memory (in any order)

Several types of software blocks, each with a different task, can be used to construct the user program:

- Organization blocks (OB)

These provide the interface between the system program and user program. There are special organization blocks which can be programmed by the user. These are intended for specific situations and are used to prompt a reaction from the user. There are also organization blocks, which the user cannot program, but can call and which contain system program special functions (see section 7).

- Program blocks (PB)

These are used to structure the user program in hardware oriented program sections (see section 2).

- Function blocks (FB)

These are used to program functions which are frequently repeated or complex functions (e.g. unit control, signalling functions, arithmetic and closed-loop control functions). Exception: FB 0 (see section 1.5.1).

- Sequence blocks (SB)

These are special types of program blocks for processing sequence cascades.

- Data blocks (DB)

These are used to store data and texts. The functions of these blocks are fundamentally different from those of the other blocks, since they do not contain a user program. DB 0 and 1 are reserved for special purposes (see section 3).

A maximum of 256 program, function and sequence blocks, 254 data blocks and 39 organization blocks can be programmed. One block may occupy a maximum of 4096 words in the CPU program memory. In the case of input/transmission of blocks with the PG, the memory size of the PG used must be taken into account.

All blocks which have been programmed can be stored in any order by the PG in the program memory (Fig. 2), which is implemented on the CPU as a plug-in RAM or EPROM.

1.3.2 Program organization

The program organization determines whether and in which sequence the blocks generated by the user will be processed (Fig. 3). Therefore corresponding calls (conditional or unconditional) for the blocks selected are programmed in organization blocks.

Additional program, function and sequence blocks can be called up in any desired combination by organization, program, function and sequence blocks.

The maximum permissible nesting depth is 24 blocks. This value can be seen as the total block nesting depth resulting from all the possible modes of operation (cyclic, interrupt-driven, time-driven and possibly also interrupt handling; see sections 5.4 to 5.7).

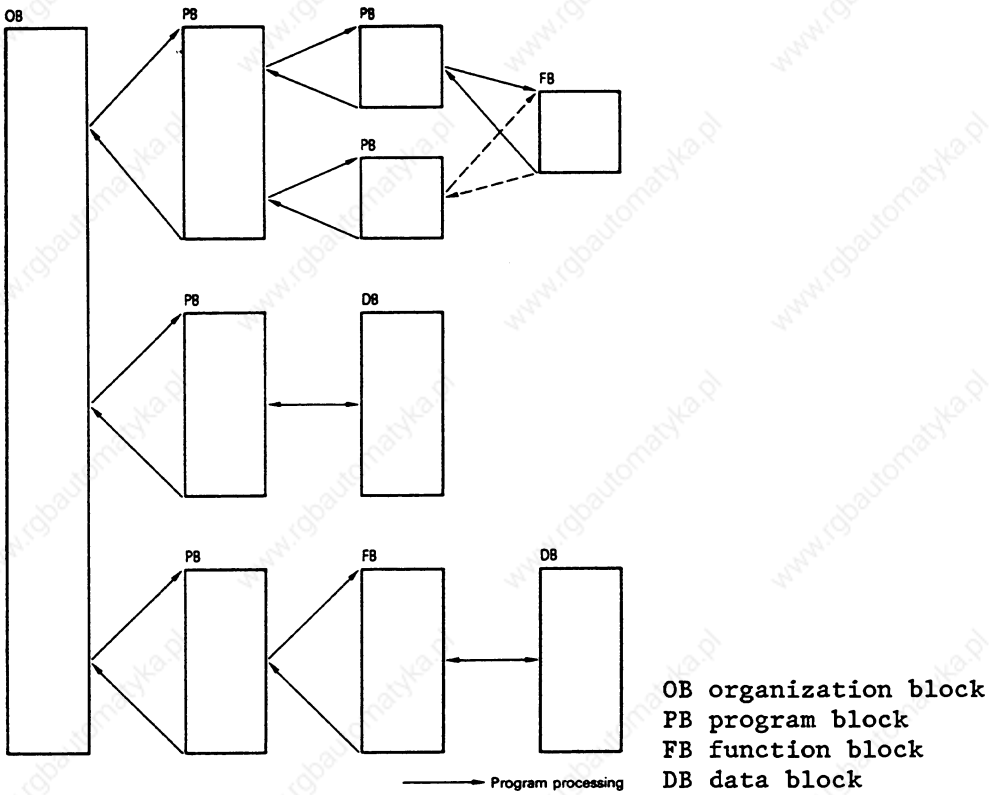


Fig. 3 Program organization in the STEP 5 programming language

1.3.3 Program storage

If a plug-in RAM is available in the CPU, the user program can be transferred directly from the PG to the CPU. All programmed blocks are stored in the RAM in any order. When the RAM is full, further **data** blocks are stored in the CPU RAM memory area (for data block RAM, see Fig. 14). The CPU RAM has enough space for 3792 words. If shift registers are used, this space is, however, reduced by 128 words per shift register called in; with the end address of the data block RAM shifting to lower addresses.

If an EPROM is used to store the user program, all programmed blocks will be stored in it. Data blocks which contain variable data - i.e. which are to be changed during the user program - must therefore be copied from the EPROM to the RAM memory area of the CPU during the cold restart (see section 7.1). Exception: DB 0 and 1 are managed by the system program.

1.3.4 Program execution

The user program can be executed in three different ways (Fig. 4):

- Cyclic program execution (see section 5.4)

In order to execute the user program cyclically, either the organization block OB 1 or the function block FB 0 can be used:

- OB 1 runs cyclically, calling the blocks programmed in the user program.
- FB 0 is executed like OB 1; in addition, however, it allows supplementary STEP 5 operations to be used. It is therefore especially suitable for the processing of small time-critical programs, which do not need structured programming nor the block calls it involves.

If OB 1 **and** FB 0 are programmed, only OB 1 will be run.

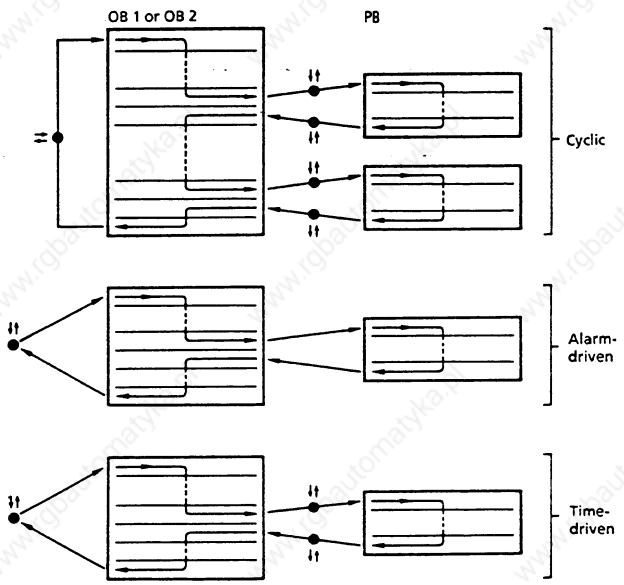
- Interrupt-driven program execution (see section 5.5)

With this type of program execution, the interruption of the cyclic program execution is initiated peripherally when there is a change of block. OB 2 is intended for calling interrupt routines.

- Time-driven program execution (see section 5.6).

With this type of program execution, certain program sections (called by OB 13) are automatically inserted into the cyclic program execution using a time base.

Time-driven program execution is necessary for the solution of closed-loop control tasks.



Interrupt point, at which interrupt-driven or time-driven program execution can be inserted into cyclic, interrupt-driven or time-driven program execution. Time-driven execution can only be interrupted by interrupt-driven execution, interrupt-driven execution only by disturbances.

Fig. 4 Types of program execution

1.4 Programming in multiprocessor operation

The programming of individual S5 135 U CPU's for multiprocessor operation corresponds to the programming for single processor operation described in section 1.3. In addition, the following aspects of multiprocessor operation should be noted:

- The individual CPU's can exchange data with each other via the flags for inter-processor communication (IPC flags) residing on the COR.
- The whole S5 135 U program can be distributed on the individual CPU's making each CPU responsible for certain aspects of the program.
- The peripheral inputs and outputs must be allocated to the individual CPU's.
- The assignment of the S5 bus to the individual CPU's is carried out by the COR. The number of CPU's used must be set on the COR (see COR operating instructions).

If there is more than one CPU in the PC, then the COR must be plugged in. As soon as the COR is used, the CPU system program is in multiprocessor operation, even if there is only one CPU. Data block DB 1 must then be programmed for each CPU.

1.4.1 Flags for inter-processor communication (IPC)

IPC flags are flag bytes which are designated by the user on the CPU as output or input. They are used for byte-serial, cyclic exchange of data between the CPU's.

A flag byte defined as an output on a CPU will be transferred, in the cylic operation of the S5 135 U, via the COR, to the CPU's, which have an input flag designated with this number.

The following rules result from this flag byte function:

- The flag designated on one or more CPU's as an IPC input flag must be defined on another CPU as an IPC output flag.
- If a flag is designated as an IPC output flag on one CPU, it cannot be defined on another CPU as an IPC output flag. It can, however, be defined on three more CPU's as an IPC input flag.
- The flag bytes designated as IPC flags on a CPU are only available on this CPU for the exchange of data. All other flags which have not been designated can be used for their normal application.

The designation of flags as IPC flags in data block DB 1 is described in section 1.4.3.

IPC flags can also be used for data transmission between CPU's and CP's. This function is possible both in single and multiprocessor operation. The IPC flag area with a maximum of 256 input and output bytes can be subdivided on the COR and/or the CP's into sub-areas of 32 bytes (see COR or CP operating instructions). All IPC flags specified in DB 1 must be set on the COR or CP's and acknowledge - if not, the system program detects an QVZ error (see section 5.7).

1.4.2 Program distribution

The CPU's (up to max. four), which process their particular user program in the multiprocessor PC simultaneously and independently of one another, allow the user to divide the whole S5 135 U program into individual, self-contained programs. As a result, multiprocessor operation offers the following advantages:

- Dividing the program among the CPU's, which then operate parallel to each other, improves the execution time of the whole program.

- Programs with short runtimes for handling processes which depend on fast responses can be put together on their own CPU's. The user program runtime in such a CPU can be further reduced if the user brings in FB 0 instead of OB 1 and makes use of the opportunity to specify a timer block length (see section 1.5.1).

- User programs with long runtimes for handling processes which are not time-critical can be programmed on their own "slow" CPU separate from the "fast" CPU's.

- Each CPU can be assigned to a particular part of the plant depending on its function.

1.4.3 Assignment of I/O's and IPC flags

In multiprocessor operation, the user **must** assign the peripheral input and output modules and the necessary IPC flags **in bytes** to the individual CPU's. DB 1 data block is provided for this purpose, in which the user enters the distribution of the I/O's and IPC flags in the form of address lists using the PG. During single processor operation, DB 1 can be programmed so that the runtime is optimized. The DB 1 has a fixed function and may not be used for other purposes.

- Structure of DB 1

The user must create DB 1 with the PG. There are two possible ways of doing this:

a) the data words 0, 1 and 2 must be preset with

KH = 4D41, 534B, 3031.

From data word 3 onwards the individual address lists are specified. Each address list begins with a keyword. Possible keywords are:

keyword for digital inputs	KH = DI00
keyword for digital outputs	KH = DQ00
keyword for IPC input flags	KH = CI00
keyword for IPC output flags	KH = CQ00
keyword for timer block length	KH = BB00

Following the keyword the relative byte addresses of the signed I/O's or IPC flags which belong to this address list are listed as data words in fixed point format. The order of the entries within an address list is as arbitrary as the order of the address lists themselves.

Following the last entry in DB 1,

KH = EE00

must be entered as an end identifier.

In multiprocessor operation, DB 1 must be generated for **each** CPU.

Example (DB 1 input with identifier)

```
KH = 4D41
KH = 534B          identifiers
KH = 3031

KH = DI00
KF = +00000
KF = +00002
KF = +00007        digital inputs
KF = +00012
KF = +00126
KF = +00127

KH = DQ00
KF = +00001        digital outputs
KF = +00003

KH = CQ00
KF = +00005        output IPC flags

KH = CI00
KF = +00126        input IPC flags

KH = EE00          end identifier
```

b) From the S0 A03 or S1 A01 software release for the PG 675 onwards DB1 can also be input supported by screen forms (using the softkeys **F1** and **F3**) in this case, the user has to enter the relative byte address in the screen form.

Example (DB 1 input with screen form)

PERIPHERAL ASSIGNMENT:

DIGITAL INPUTS	2	7	12	126	127						
DIGITAL OUTPUTS	1	3									
IPC INPUT FLAGS	126										
IPC OUTPUT FLAGS	5										
TIMER BLOCK LENGTH											

- Input/change DB 1

- On line via the PG when the CPU is in the stop status, and if it is equipped with a user RAM. DB 1 which has been input or changed will **only** be accepted by the system program in the form of internal address lists during a manual cold restart without memory recall (see section 5.3).

- By means of programming the user EPROM.

- I/O address lists for cyclic execution

  During cyclic execution the process image is updated but **only** for the digital I/O modules which are specified in both of these adress lists. During the start-up, the system program checks whether the input and output bytes specified in DB 1 acknowledge on the corresponding I/O modules. If they do not, the CPU stops and outputs a DB 1 error message.

In **multi**processor operation, **direct** I/O access bypassing the process image (loading/transfer commands L PB, T PB, L PW, T PW, see section 6.2) is possible:

- For all digital **in**puts available to the S5 135 U, independent of their entry in the address list of the CPU in question,

- For digital **out**puts only if they are allocated to the CPU in question, i.e. entered in the address list.

- However, with direct I/O access in multiprocessor operation, the PC always waits until the bus has been assigned (automatic, cannot be influenced by the user).

If an address list is transferred to a CPU in **single** processor operation, **direct** I/O access is possible to all existing inputs and outputs, irrespective of the address list.

Access to the process image using loading/transfer commands L I.., T Q.. and logic operations are permissible both in single and multiprocessor operation but only for those inputs and outputs specified in DB 1.

Note! Each output byte may only be allocated to one CPU.

- IPC flag address list

The flag bytes specified in these address lists are read (IPC input flags) or written (IPC output flags) while the CPU is in cyclic operation. For the allocation of IPC flags to the individual CPU's, the rules in section 1.4.1 should be followed.

1.5 General notes

For the purposes of the system program or the assignment of parameters to the individual CPU's, the data blocks DB 0 and 1 are already occupied.

1.5.1 Runtime optimization of the user program

- Program structure

In single processor operation - as well as in multiprocessor operation - the runtime of the user program can be reduced, if the user only makes use of structured programming when necessary.

As every block change requires additional runtime, structured programming can be avoided for short, time-critical programs, and only FB 0 should be programmed. In FB 0, the whole STEP 5 instruction set (see section 6) present in the S5 135 U is available.

- Cycle time

The runtime of the user program is the sum of the runtimes of the blocks which have been called. If a block is called n times, its runtime must be taken into account n times.

The sum of the runtimes of all user program parts (cyclic plus time-driven plus interupt-driven) is the cycle time. This is limited to 100 ms in the S5 135 U control processor. The cycle time is monitored by the system program; if it is exceeded, the CPU is stopped with the "CYC" error message (see section 5.7).

By calling the system program special function "cycle time triggering" (see section 7.4), the user can prolong the time of the cycle currently running by 100 ms, starting from the moment the special function was called.

- I/O assignment

In both single and multiprocessor operation, it is important that, in the address lists for I/O's and IPC flags, only those addresses are specified which the user program of the CPU in question accesses.

I/O addresses and addresses of IPC flags, which are not necessary for the particular user program, but which were cyclically updated due to the entry in DB 1, extend the runtime of the whole program.

- Timer block length

In DB 1 the user can specify the number of timer locations used as the timer block length. As a result the execution time for all the timer locations outside this timer block length will be saved.

However, this is only possible if the numbers of the timer locations used by the user are smaller than the specified timer block length. If 0 is specified as the timer block length, no timer locations will be processed. If **no** timer block length is specified, then **all** timer locations are permissible. If timer locations are processed which have numbers greater than or equal to the timer block length, the CPU detects an error and stops.

The timer block length can also be input in single processor operation. The user must, however, program the **complete** DB 1 address lists, i.e. he must also specify the address lists of the I/O's (see section 1.4.3).

Timer block length entry in DB 1 (example for 40 timer locations with permissible numbers 0 to 39):

```
   .
   .
   .
KH = B B 0 0          identifier
KF = + 0 0 0 4 0      timer block length
   .
   .
   .
   .
KH = E E 0 0          end identifier
```

2 Program blocks

2.1 Programming program blocks

The following description applies to the programming of organization blocks, program blocks and sequence blocks. These three types of blocks do not differ as far as programming is concerned. They can be programmed in all three methods of representation STL, LAD and CSF of the STEP 5 programming language. Programming is started by entering a block number:

- program blocks 0 to 255
- sequence blocks 0 to 255
- organization blocks 1 to 39 (see section 5)

This is followed by the actual logic control program which is completed with the statement BE. Only the STEP 5 basic operation set can be used.

The block is made up of the block program (STEP 5) and a block header. The block header is automatically generated by the PG and occupies 5 words in the program memory.

A block should always contain a complete program. Logic operations which go beyond the block limits are meaningless.

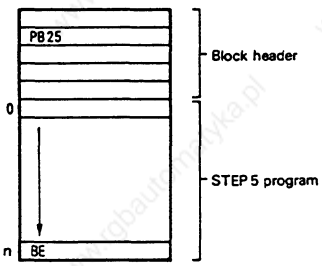


Fig. 5 Structure of an organization, program and sequence block

2.2 Calling program blocks

Block calls enable the blocks for processing (Fig. 6). These block calls can be programmed within an organization, program, function or sequence block. They are comparable with jumps to a subprogram and can be implemented both conditionally and unconditionally.

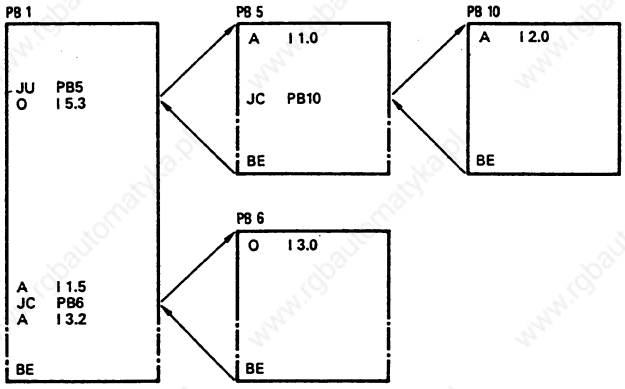


Fig. 6 Blocks calls which enable the processing of a program block

Following the BE statement (block end) in the block which has been called, a jump is made back to the block, in which the block call was programmed, and the STEP 5 command which follows the block call will be executed. Both following a block call and following BE, no further logic operations can be carried out on the result of the logic operation (RLO, see section 6.1), since both of these are RLO limiting commands. The RLO is, however, taken into the new block and can be evaluated there by means of RLO dependent commands.

- Unconditional call: JU xx

The program block addressed is processed independent of the result of the previous logic operation.

- Conditional call: JC xx

The program block addressed is processed dependent on the result of the previous logic operation. When RLO = 1, the jump statement is executed, when RLO = 0, it is not. In **both** cases, RLO is set to 1 by the jump statement. This dependence on the RLO and its influence also applies to the conditional block end statement BEC.

3 Data blocks

3.1 Programming data blocks

Data required within the user program are stored in data blocks. No STEP 5 operations are carried out in data blocks. Data may consist of:

- any desired bit pattern, e.g. for plant status,
- numbers (hexadecimal, fixed point, floating point) for times or results of calculations,
- alphanumeric characters e.g. for message texts.

The generation of a data block is started by specifying a data block number between 2 and 255 (e.g. DB 25). The data words (16 bits) must be input in ascending order, starting with data word 0. Data blocks DB 0 and DB 1 are reserved for specific functions and are not available to the user.

One memory word is reserved per data word in the program memory. A block header, occupying five more words in the program memory, is generated by the PG for each data block. A data block may occupy a maximum of 4096 words in the CPU program memory. When entering/transferring using the PG, the memory size of the PG must be taken into account.

Caution! With the L/T DW... load/transfer commands, access is only possible up to data word number 255.

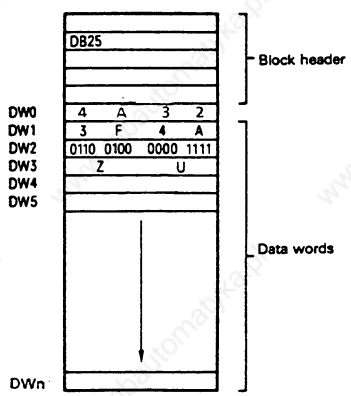


Fig. 7 Structure of a data block

3.2 Calling data blocks

Data blocks can only be called unconditionally. The selection remains valid until a new data block is selected. A DB data block can be called within an organization, program, function or sequence block with the command C DBxxx.

Caution! Before a data word is loaded/transferred, a data block must have been selected. The addressed data word must be contained in the selected data block (no check).

Example 1

The contents of data word 1 should be transferred from DB 10 to data word 1 of DB 20 (Fig. 8).

```
:C DB 10
:L DW  1
:C DB 20
:T DW  1
```

If a further program block is called by a program block in which a data block has already been addressed, and another data block is addressed in this second program block, then this second data block is only valid in the program block which has been called. Following the jump back to the first program block, the old data block is valid again (see Fig. 9).

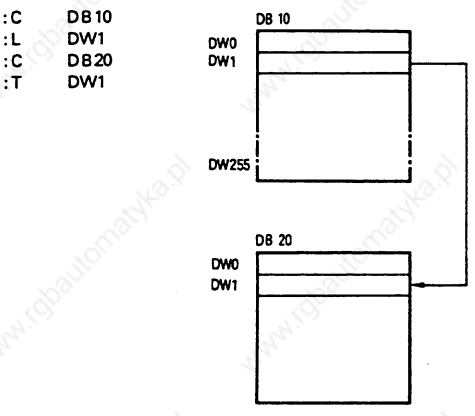
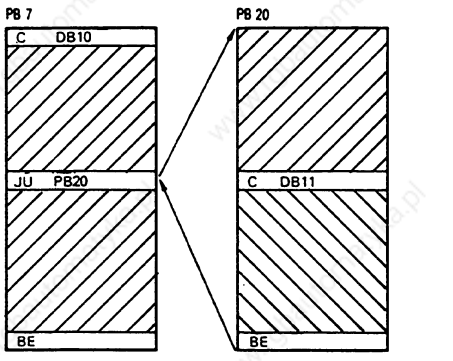


Fig. 8 Addressing a data block



/// area of validity of DB 10
\\\ area of validity of DB 11

Fig. 9 Area of validity of a selected data block

Example 2 (see Fig. 9)

In the program block PB 7, the data block DB 10 is selected. In the subsequent operation the data of this data block will be processed.

Following the call, the program block PB 20 is processed. DB 10, however, remains valid. Only when DB 11 is called will the data area be changed. DB 11 is now valid until the end of the program block PB 20.

After the block change back to the program block PB 7, the data block DB 10 will be valid again.

4 Function blocks

4.1 General

Function blocks are just as much a part of the user program as e.g. program blocks. Compared to the organization, program, and sequence blocks, the function blocks have four essential differences:

- Function blocks can have parameters assigned to them, i.e. the actual operands with which a function block is to operate, can be varied by using formal operands.

- In contrast to organization, program, and sequence blocks, the function blocks can be programmed with an extended operation set.

- The program of a function block can only be created and documented in a statement list (STL).

- A function block call will be represented graphically as a "black box".

Function blocks represent complex, self-contained functions within the user program. A function block can either be obtained as a software product (standard function blocks on mini diskette), or programmed by the user himself. The extended operations which are available in addition to the basic operations, can only be programmed in function blocks.

4.2 The structure of function blocks

A function block consists of a block header and a block body (Fig. 10).

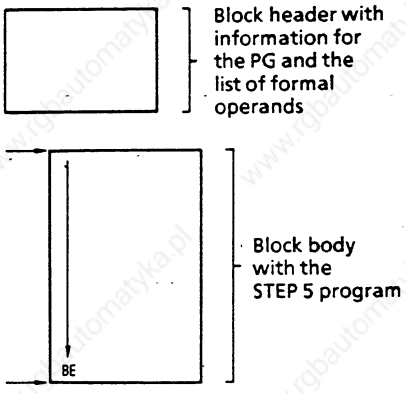


Fig. 10 The structure of a function block

- Block header

The block header contains all the information which the PG requires to be able to represent the function block graphically and to be able to check the operands during assignment of parameters to the function block. Before the function block is programmed this block header is input by the user (with the support of the PG). It is stored in the program memory of the CPU and contains a jump statement which is carried out during the function block call, but is not displayed when it is read out (jump over formal operand list).

- Block body

The block body contains the actual program of the function block. The function to be executed is written in the STEP 5 programming language and entered in the block body. When the function block is called, only the block body is processed. An extended operation set of greater scope than the basic operations set is available for programming function blocks (see section 6.3).

4.3 Calling function blocks and parameter assignment

Repetitive or very complex functions are implemented by function blocks. These exist only once in the program memory and are called in once or several times by a higher-ranking block. At each call, other parameters can be used.

Function blocks are stored in the program memory under a particular designation (FB 0 to FB 255). If standard function blocks are used (FB 1 to 199), then their numbers are no longer available for user function blocks.

FB 0 should only be used for the programming of cyclic program execution, instead of OB 1 (see section 5.4).

The function block call can be programmed within an organization, program, or sequence block or within another function block. The call consists of the call statement and the parameter list.

- Call statement

- Unconditional call (JU FBn):
  The function block addressed is processed independently of the result of the previous logic operation.

- Conditional call (JC FBn):
  The function block addressed will only be processed if the result of the previous logic operation is RLO = 1. If RLO = 0, the jump statement will not be executed. In **both** cases, RLO will be set to 1 by the conditional jump statement.

Following the unconditional and conditional call, the result of the logic operation can be evaluated but cannot, however, be further operated on. It is taken along with the jump into the function block called.

- Parameter list (see example)

The parameter list is in the block which is calling directly following the call statement. In the call statement, the input and output variables, as well as data, are defined (see "Classes of Block Parameters"). The parameter list can contain a maximum of 40 variables. It allocates the variables (actual operands), to the formal parameters (formal operands) of the function block.

When the function block program is executed, the variables from the parameter list will be used instead of the formal parameters. The PG monitors the order of the variables in the parameter list.

A jump statement is automatically inserted by the PG following the FB call, but is not displayed when the FB is read out. The FB call, the jump statement, and each parameter occupy one memory word each. (Exception: two words with floating point number).

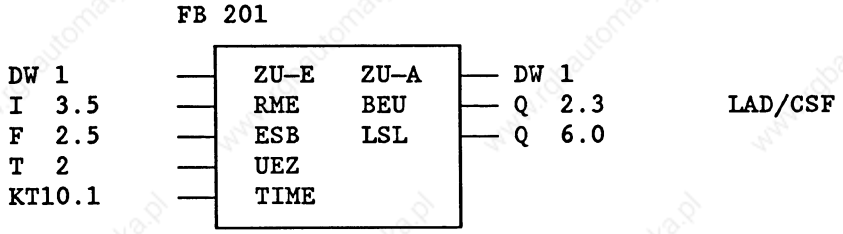
Example (calling a function block and transferring parameters with the STL and LAD/CSF methods of representation in a program block)

Method of representation

PB25

```
      :  JU FB 201
NAME  : E-ANTR
ZU-E  :     DW   1
RME   :     I    3.5
ESB   :     F    2.5                          STL
UEZ   :     T    2
TIME  :     KT   10.1    Parameter list
ZU-A  :     DW   1
BEU   :     Q    2.3
LSL   :     Q    6.0
```

Formal operands — Actual operands



LAD/CSF

The input/output identifiers of the function block as well as the name of the function block, which appear on the PG during programming, are stored in the function block itself. Before starting to program with the PG, all necessary function blocks must therefore be transferred onto the program diskette or input directly into the program memory of the PC (for more details, see operating instructions of the PG).

4.4 Programming function blocks

In keeping with the structure of a function block, the generation is divided into two parts:

Before the block body is input (STEP 5 program) the block header is entered. The block header contains:

- the library number,
- the name of the function block,
- formal operands (the names of the block parameters),
- the class of block parameter,
- the type of block parameter.

• Library number

A number from 0 to 99 999 can be assigned. This number is allocated to the function block, regardless of its symbolic or absolute parameter.

A library number should only be specified once in order to be able to identify a particular function block uniquely. Standard function blocks have a product number.

• Function block name

The name which designates the function block can be up to 8 characters long. It is not identical to the symbolic plant identifier.

• Formal operand (block parameter name)

The formal operand can be up to 4 characters long, and must start with a letter. Up to 40 parameters can be programmed per function block.

Class of block parameter

The class of block parameter may be either I, Q, D, B, T or C.

I = input parameter
Q = output parameter
D = data
B = command
T = timer
C = counter

I, D, B, T or C are parameters which, in the case of graphical representation, appear on the left-hand side of the function symbol. Parameters designated by Q appear on the right-hand side of the function symbol.

Type of block parameter

For the I, Q and D classes of parameter, the type of parameter must also be specified:

BI/BY/W/D	for I and Q parameter classes
KM/KH/KY/KS/KF/KT/KC/KG	for the D parameter class

With I and Q parameters the type of parameter specifies whether bit sizes, byte sizes, word sizes or doubleword sizes are used and which data format is valid for the D parameter.

Class of parameter	Type of parameter	Legal actual operands
I, Q	BI for an operand with bit address	I n.m inputs Q n.m outputs F n.m flags
	BY for an operand with byte address	IB n input bytes QB n output bytes FB n flag bytes DL n data byte left DR n data byte right PB n peripheral bytes OB n peripheral bytes from the extended peripherals
	W for an operand with word address	IW n input words QW n output words FW n flag words DW n data words PW n peripheral words OW n peripheral words from the extended peripherals
	D for an operand with doubleword address	ID n input doublewords QD n output doublewords FD n flag doublewords DD n data doublewords
D	KM for a binary pattern (16 bits) KY for two byte serial decimal numbers from 0 – 255 KH for a hexadecimal pattern up to 4 digits KS for a character (max. 2 alphanumeric characters)	constants

Class of parameter	Type of parameter	Legal actual operands
D	KT for a timer value (BCD coded) with a time base of 1.0 to 999.3 KC for a counter value (BCD coded) of 0 to 999 KF for a fixed-point number from −32768 to +32767 KG for a floating point number	constants
B	No type specification permitted	DB n data blocks; the C DB n command is executed FB n function blocks (only permissible without parameters) are called unconditionally (JU ..n) PB n program blocks are called unconditionally (JU ..n) SB n sequence blocks are called unconditionally (JU ..n)
T	No type specification permitted	T 0 to 127 timer [1])
C	No type specification permitted	C 0 to 127 counter [1])

[1]) The timer or counter value should have parameters assigned to it as data or should be programmed as a constant in the function block.

Example (programming a function block)

FB 202

```
NAME : EXAMPLE
DECL : MIKE     I/Q/D/B/T/C : I  BI/BY/W/D :  BI Formal
DECL : BERT     I/Q/D/B/T/C : I  BI/BY/W/D :  BI operand list
DECL : MAUD     I/Q/D/B/T/C : Q  BI/BY/W/D :  BI
```

```
     : A = MIKE                                  STEP 5 program
     : A = BERT
     : = = MAUD
```

Formal operands — Parameter class — Parameter type

Substitution commands

Example (function block call in a program block)

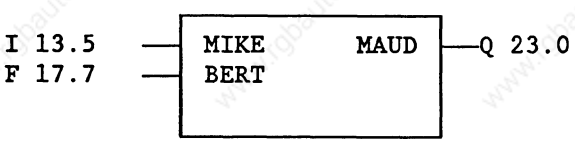
Method of representation

STL

```
    : JU FB 202
NAME : EXAMPLE
MIKE :   I 13.5
BERT :   F 17.7
MAUD :   Q 23.0
```

Formal operands — Actual operands

LAD/CSF



- Program executed

```
     : A  I 13.5
     : A  F 17.7
     : =  Q 23.0
```

Operations, (substitution commands) to which parameters are to be assigned, are programmed in the function block with the formal operands. It is also possible to reference the formal operands several times at different places in the function blocks.

During the function block call, the formal operands are substituted by the actual operands of the parameter list.

Caution! If the order or number of formal operands in the function block header is changed, the substitution commands in the function block program and the parameter list in the block which is calling must be modified accordingly.

Example (standard function block)

```
EXTR:                                FB 6  for 115 A
      floating point root extractor  FB 6  for 135 A
GP                                   FB 19 for 150 A
```

The ROOT:GP function block extracts the root of a floating point number (8-bit exponent and 24-bit mantissa). The result is also a floating point number (8-bit exponent and 24-bit mantissa) whereby the LSB of the mantissa will not be rounded.

If applicable the function block sets the identifier "radicand negative" for further processing.

Numerical range:

radicand	-0.1469368 exp. -38 to +0.1701412 exp. +39
root	+0.3833234 exp. -19 to +0.1304384 exp. +20

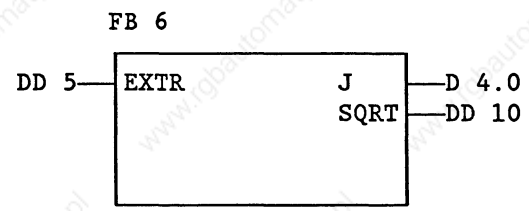
Function: $Y = \sqrt{A}$
Y = SQRT; A = EXTR

Function block call:

- STL representation

```
        : JU  FB 6
NAME    : ROOT: FP
EXTR    : DD 5
J       : D 15.0
SQRT    : DD 10
```

- LAD representation



DD = data doubleword

In the above example, the root of a floating point number, which is available in DD 5 with an 8-bit exponent and a 24-bit mantissa, is extracted. The result, which is again a 32-bit floating point number, is stored in DD 10. The appropriate data block must be selected beforehand. The J parameter (class of parameter: Q, type of parameter: BI) specifies the sign of the radicand: J = 1 with negative radicands. Occupied flag words: FW 238 to 254.

The catalogue ST 57 shows the standard function blocks for the S5 135 U, their runtimes, their memory space requirements and the variables occupied by them.

General notes

If standard function blocks are used, the flag bytes 200 to 255 are occupied and are no longer available to the user.

The timer 0, the counter 0, and data blocks DB 0 and 1 are also occupied.

Standard function blocks occupy the numbers 1 to 199. User function blocks can therefore only be created with the numbers 200 to 255.

The function block FB 0 is called in cyclically by the system program instead of the organization block OB 1, if OB 1 is not programmed.

5 Organization blocks

5.1 General

The organization blocks constitute the interface between the system program and the user program (see Fig. 12). The organization blocks OB 1 to OB 39 are a part of the user program, just like program, function or sequence blocks. Organization blocks are called by the system program. The user can program the organization blocks OB 1 to 39 and thus have an indirect influence on the system program. For testing purposes these organization blocks can also be called by the user with (JU/JC OB xxx).

By appropriately programming the organization blocks the following modes of operation can be set:

- cyclic processing (OB 1 or FB 0)
- interrupt-driven processing (OB 2)
- time-driven processing (OB 13)
- manual cold restart without memory recall (OB 20)
- manual cold restart with memory recall (OB 21)
- automatic cold restart with memory recall (OB 22)

Device error handling:

- if one of the device errors described in section 5.7 occurs, organization block OB 28 will be called.

● OB for special functions

Apart from organization blocks OB 1 to 39, system program special functions can be called as organization blocks in the S5 135 U using numbers greater than 39. These organization blocks for special functions cannot be programmed by the user - they can only be called. They do not contain a STEP 5 program. The special functions are described individually in section 7.

Special function OB's can also be called within organization blocks 1 to 39 (from software release S0 A03 and S1 A01 of the PG 675).

- Significance of the organization blocks

Absolute parameter	Designation or reason for initiation
OB for cyclic processing:	
OB 1	Program start (also FB 0)
OB for interrupt-driven processing:	
OB 2	Process-interrupt
OB for time-driven processing:	
OB 13	Time base with 0.1 s
OB's for cold restart:	
OB 20	Manual cold restart without memory recall
OB 21	Manual cold restart with memory recall
OB 22	Automatic cold restart with memory recall
OB's for error handling:	
OB 28	Reaction to error
OB's for special functions:	
OB 220	Fixed point expansion from 16 bits to 32 bits
OB 221	Delete all shift register user memory locations
OB 222	Trigger cycle time
OB 223	Stop if CPU's do not all operate in same start up mode during multiprocessor operation
OB 224	Transmit IPC flags in blocks during multiprocessor operation
OB 225	Assigning parameters to start-up characteristics
OB 226	Reading a byte from the system program EPROM
OB 227	Reading the cross-check sum of the system program EPROM
OB 240	Initializing the shift register
OB 241	Calling shift register no. 1 during the cycle
OB 242	Calling shift register no. 2 during the cycle
OB 243	Calling shift register no. 3 during the cycle
OB 244	Calling shift register no. 4 during the cycle
OB 245	Calling shift register no. 5 during the cycle
OB 246	Calling shift register no. 6 during the cycle
OB 247	Calling shift register no. 7 during the cycle
OB 248	Calling shift register no. 8 during the cycle
OB 250	Initializing the PID controller
OB 251	Calling the PID controller during the cycle
OB 255	Transmit data block from the user program memory into the data block RAM

For a description of the organization blocks for special functions, see section 7.

5.2 Commissioning

Both with single processor and multiprocessor operation, there are different operating statuses:

- stop status
- start-up
- program execution

Each operating status can be divided into three types (see Fig. 11).

Stop	Start-up	Program execution
"STOP" LED flashes quickly	Manual cold restart without memory recall	Cyclic
"STOP" LED is continuously lit	Manual cold restart with memory recall	Interrupt-driven
"STOP" LED flashes slowly	Automatic cold restart with memory recall	Time-driven

Fig. 11 Operating statuses

Commissioning with a RAM or EPROM submodule is described in the central controller operating instructions. After the supply voltage has been switched on, the CPU runs through an initialization routine, in which the following functions are executed irrespective of the preceding operating status:

- initialization of all memory modules,
- setting up of the block address list (DB 0) showing all blocks present in the user program,
- recognition of single or multiprocessor operation.

If errors such as

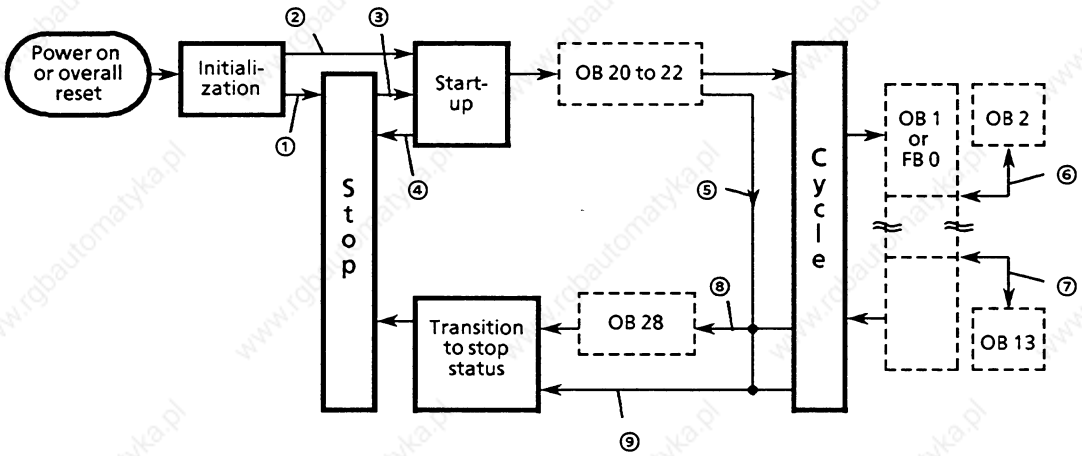
- defective memory contents,
- the absence of a user program memory submodule or if an EPROM is empty,
- interrupted buffering of CPU RAM or user program RAM

are detected during initialization, then the CPU stops, and the "STOP" LED flashes quickly. The CPU must be overall reset (see section 5.2.2), this also applies whenever the module is plugged into the central controller.

Fig. 12 shows the structure underlying the system program. The organization blocks constitute the interface between the system program and the user program. They do not necessarily have to be programmed because the system program makes its services available even without them.

Examples:

- Only OB 1 is programmed. The CPU starts the cyclic program execution using the selected start-up mode, even if no user start-up OB is programmed.
- Only OB 13 is programmed. The system program carries out its cyclic operations (see section 5.4) and the time-driven program execution.



——— system program
– – – user program

1 a) Errors have been detected during initialization or b) operating mode switch at "STOP" or c) status was previously stop or d) automatically following overall reset
2 Automatic cold restart with memory recall (see section 5.3)
3 Manual cold restart with/without memory recall (section 5.3)
4 Error during start-up before user start-up OB called, (e.g. DB 1 error, selection of an illegal start-up mode)
5 Causes of trouble during processing of user start-up OB's (see section 5.7)
6 Interrupt-driven program execution (see section 5.5)
7 Time-driven program execution (see section 5.6)
8 Call OB 28 when certain causes of trouble arise (see section 5.7).
9 Direct transition to stop status without calling an OB when certain causes of trouble arise (see section 5.7)

Fig. 12 The structure underlying the system program in the S processor

The different types of operating statuses will now be described.

5.2.1 Stop status

The stop status of the PC is indicated by the red "STOP" LED. There are three different types of stop status:

- The "STOP" LED is lit continuously

Stop status following termination of the user program execution
a) in single processor operation by switching the mode selector to "STOP";
b) in multiprocessor operation as with a), and if another CPU or the COR has caused the termination;
c) in multiprocessor operation by means of the PG function "PC stop"
d) in single or multiprocessor operation because of device errors which are not allocated to an individual CPU (NAU, BAU, PEU);
e) after the PG function, "program check" has ended, and following an overall reset.

o The "STOP" LED flashes quickly (overall reset request)

a) During the initializatiobn, errors have been detected which occurred owing to interruption of the buffering of the user program RAM or the CPU RAM before the power was switched on.
b) The user program EPROM is empty or not plugged. A programmed EPROM must be plugged in.
c) An overall reset has been requested by the user (see section 5.2.2).

In a) and b), the system program requests an overall reset. The CPU **must** be overall reset (see section 5.2.2). In c), the user can either implement an overall reset or avoid doing this by selecting a start-up mode (see section 5.3).

o The "STOP" LED flashes slowly (error)

a) The CPU caused an error (see section 5.7), in single or multiprocessor operation, which resulted in the PC stopping.
b) The CPU has been incorrectly operated (selection of an illegal start-up mode, DB 1 error), even if afterwards the operating mode switch is switched to "STOP". The CPU had not yet started processing the user program (corresponds to point 4 in Fig. 12).
c) A stop command has been programmed in the start-up OB or the cyclic user program.
d) In single processor operation by means of the PG function "PC stop".

Reaction to stop status

The BASP signal (inhibit command output) is output (exception: test operation), which disables all digital outputs and is also indicated by the "BASP" LED. In multiprocessor operation, if one CPU goes into the stop status, then the others stop automatically (exception: test operation; see section 5.2.3). With stops caused by the PG, the stop switch, a stop command and with certain device errors, the system program calls organization block OB 28 in which the user can program the reactions desired (see section 5.7).

Possible ways of leaving the stop status

- overall reset, then manual cold restart without memory recall
- test operation (see section 5.2.3),
- selection of a start-up mode (see section 5.3):
  manual cold restart without memory recall or
  manual cold restart with memory recall.

5.2.2 Overall reset

Function:

- reset the CPU
- delete all RAM's

To initiate an overall reset:

1 Transition to stop status ("STOP" LED continuously lit).
2 Hold selector switch in "OVERALL RESET" position; at the same time, switch mode selector from "STOP" to "RUN" and then back to "STOP" again.
3 Result: the "STOP" LED flashes quickly, warning "overall reset is requested" [1]).
4 Hold selector switch in "OVERALL RESET" position; at the same time, switch stop switch from "STOP" to "RUN" and then back to "STOP" again.
5 Result: the "overall reset" function is carried out, then the processor stops and the "STOP" LED is lit continuously.
6 Following an overall reset, the only start-up mode permitted is manual cold restart without memory recall.

If the overall reset is requested by the system ("STOP" LED flashes quickly, see above), points 1 to 3 can be omitted. The overall reset can also be carried out with the PG (see PG operating instructions).

5.2.3 Test operation

With the test operation it is possible to start up individual CPU's in a multiprocessing system (or any desired combination of CPU's) without the CPU's in the stop status blocking the whole PC. The following special features should be noted here:

- The start-up of the individual CPU's is not synchronized. Depending on the length of the organization blocks called at the start-up (OB 20, 21, 22) the CPU's start the cycle at different times.
- The BASP signal is not output. In the event of an error, the digital outputs are not disabled (exceptions see below).
- If during test operation an error is detected in a CPU which is in cyclic operation, then only the CPU concerned is stopped. Exception: with the PG functions "program check" and "PC stop", DB 1 errors, and during an overall reset, the PC stops and BASP is output.

[1]) At this stage, an overall reset can be avoided by "STOP" ---> "RUN" ---> "STOP" without moving the selector switch: the CPU goes back to stop status. After this, a start-up mode should be selected.

Initiating the test function

The test function must be enabled at the COR (see COR operating instructions).

On the COR, the selector must be switched from "STOP" to "TEST"; the "BASP" LED must then go out.

The start-up mode must be selected on the CPU's which are to go into cyclic operation (see section 5.3).

5.3 Programming the start-up characteristics

The system program of the CPU has three different PC start-up modes:

- manual cold restart without memory recall (e.g. after programming the CPU, following overall reset),
- manual cold restart with memory recall (flags and IPC flags are not erased),
- automatic cold restart with memory recall (only following power failure).

For each start-up mode, the system program calls an organization block which the user can program to determine the events during start-up. If this is not required, these organization blocks do not need to be programmed.

The permissible start-up mode is displayed by the PG during an error analysis in the stop status (see section 5.7, control bits NEU-ZUL, MWA-ZUL).

The BASP signal is output during the start-up. The digital outputs are only enabled when the cyclic program execution starts. If e.g. cyclic program execution is terminated by an error, it will be continued at the start of a new cycle after a start-up has been run through.

With every start-up mode, all I/O outputs and the process image of the outputs are erased by the system program before the user start-up OB is called (OB 20 to 22) and the process image of the inputs is updated. Errors which occur in the user start-up program (OB 20 to 22) are recognised and dealt with in the same way as in the cycle (see section 5.7); however, no **scan** time monitoring takes place. "Manual cold restart without memory recall" is the only start-up mode permitted following a terminated start-up.

In **multiprocessor operation**, the following must also be taken into account:

- A start-up will only be executed if a data block DB 1 is present in **every** CPU.
- **Following** the individual CPU's, the COR must be started by switching the mode selector from "STOP" to "RUN". Exceptions: with test operation, the COR start is omitted. When starting with the PG function "PC start", the COR can be started automatically.

- The start-up of the individual CPU's is chronologically synchronised, i.e. they all go into cyclic operation together even if their start-up times are of different lengths. The CPU's remain in a wait loop until they have all finished their start-up procedures (does not apply with test operation).
- When the COR is started, the operating mode switches on all CPU's must be in the "RUN" position. The start-up mode of each individual CPU depends on what the operator has input during the stop status. It is possible for some CPU's to execute a manual cold restart **with** memory recall, and others a manual cold restart **without** memory recall. If there has been **no** operator input, then they will execute a manual cold restart **with** memory recall.
- It is only possible to start the PC solely by using the COR start if the COR alone was responsible for the stop status (the COR mode selector switched from "RUN" to "STOP"). In this case, when the COR is started, all CPU's carry out a manual cold restart **with** memory recall.

• Manual cold restart without memory recall

To initiate the manual cold restart without memory recall:

- hold selector switch in the "RESET position,
- change the mode selector from "STOP" to "RUN",
- in multiprocessor operation, following the cold restart of an individual CPU:
  start the COR
- or use the PG function "PC start" provided that the mode selectors on all CPU's and on the COR are still in the "RUN" position, (see PG operating instructions).

The system program then:

- resets all flags, timers and counters.

Calling the organization block OB 20:

In the organization block OB 20, the user can store a program which carries out particular activities before the start of the cyclic program execution, e.g. sets flags, starts timers, sets outputs and, if necessary prepares the data exchange between the PC and I/O devices. OB 20 must be completed with BE (block end). After OB 20 has been processed, the cyclic execution starts by calling OB 1 or FB 0.

A cold restart without memory recall is essential if DB 1 has been overall reset, input, or changed and following program execution which was terminated during the start-up.

• Manual cold restart with memory recall

To initiate the manual cold restart with memory recall:

- the selector switch must be in the middle position;
- change the mode selector from "STOP" to "RUN";
- in multiprocessor operation, following cold restart of an individual CPU:
  start COR
- or use the PG function "PC start" provided that the mode selectors on all the CPU's and on the COR are still in the "RUN" position.

During manual cold restart with memory recall the results acquired before the PC stopped and the previous operating statuses are taken into account, i.e. flags and IPC flags are not erased. The time values and counts are erased (see note!).

With the manual cold restart with memory recall, OB 21 is called, in which the user can program particular presettings, before the cyclic program is executed.

• Automatic cold restart with memory recall

Initiation of the automatic cold restart with memory recall:

- during the **cycle**, switch power off and then on again,
- leave switch positions as they are.

When there has been a power failure, the PC tries automatically to carry out a cold restart when the power returns. In this case, the system program first calls the organization block OB 22, in which the user can program the presettings of particular statuses. Otherwise, the function of the automatic cold restart is identical to the manual cold restart with memory recall. If the PC is not intended to carry out an automatic cold restart, the "stop" instruction must be programmed in OB 22.

```
OB 22 : STP (stop)
      : BE (block end)
```

Note: during manual and automatic cold restarts with memory recall, the user has the option of retaining the status of the timer and counter locations by calling the special function OB 225. OB 225 must be called **before** OB 21 or 22 has finished being processed. It becomes active during the next cold restart with memory recall. It should be noted that following each cold restart without memory recall, the standard setting "erase timer and counter locations" is valid until OB 225 is called.

5.4 Programming the cyclic execution

Cyclic program execution is the normal type of execution with programmable logic controllers (Fig. 13). After running through the operating mode once, the CPU automatically begins cyclic processing at the start of the user program. It works through the STEP 5 instructions in turn until the end of the program, and then begins processing again at the start of the program.

The organization block OB 1 or the function block FB 0 is the interface between the system program and the cyclic execution of the user program. The first STEP 5 instruction in OB 1 is also the first instruction of the user program, i.e. synonymous with the program start. If OB 1 **and** FB 0 are programmed, only OB 1 will be processed by the system program.

In OB 1 or FB 0, the program, function and sequence blocks of the cyclic program are called. In these blocks there may be further block calls, i.e. the blocks can be nested up to a depth of 24 blocks. This value is arrived at by taking the sum of the nesting depth resulting from all three possible operating modes (cyclic, interrupt-driven, time-driven) and if applicable interrupt handling (OB 28, see section 5.7).

The runtime of the user program is the sum of the runtimes of the blocks which have been called. If a block is called n times, its runtime must be taken into account n times (see section 1.5.1)

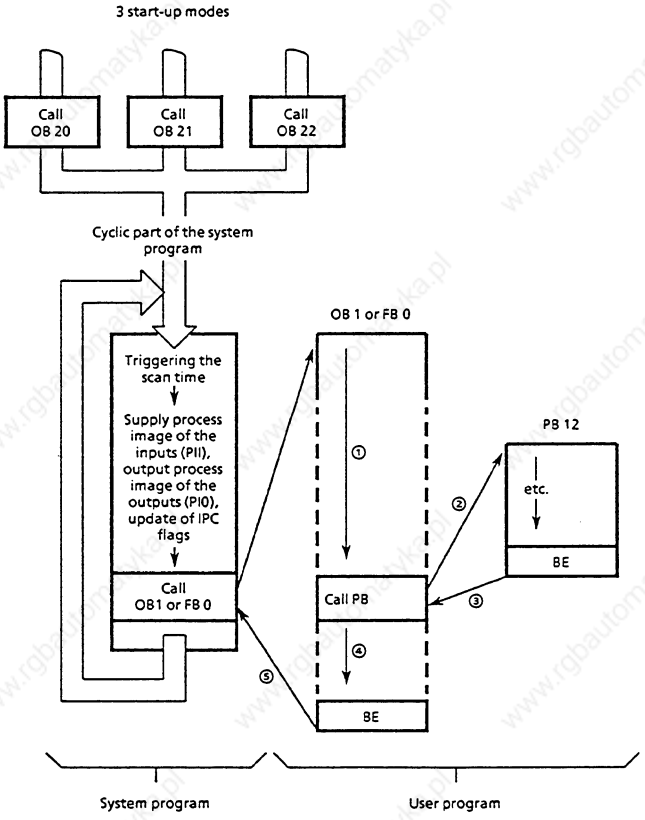


Fig. 13 Cyclic program execution

- General structure of the user program

OB 1 or FB 0 contains the general structure of the user program. The documentation of this block is intended to show the basic program structures (Fig. 14) or emphasize the parts of the system which are connected in terms of the program (Fig. 15).

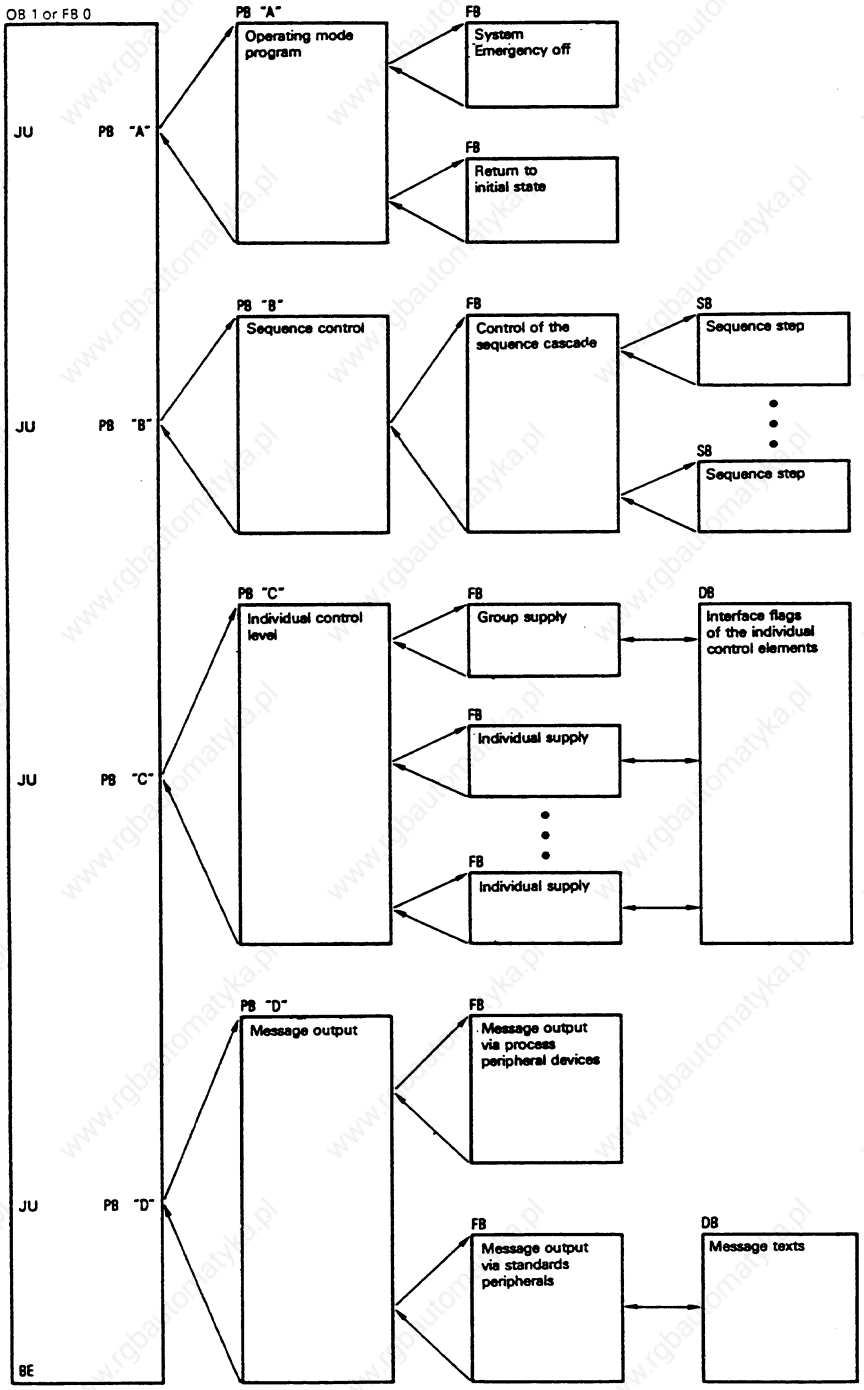


Fig. 14 Basic structure of the user program related to program structure

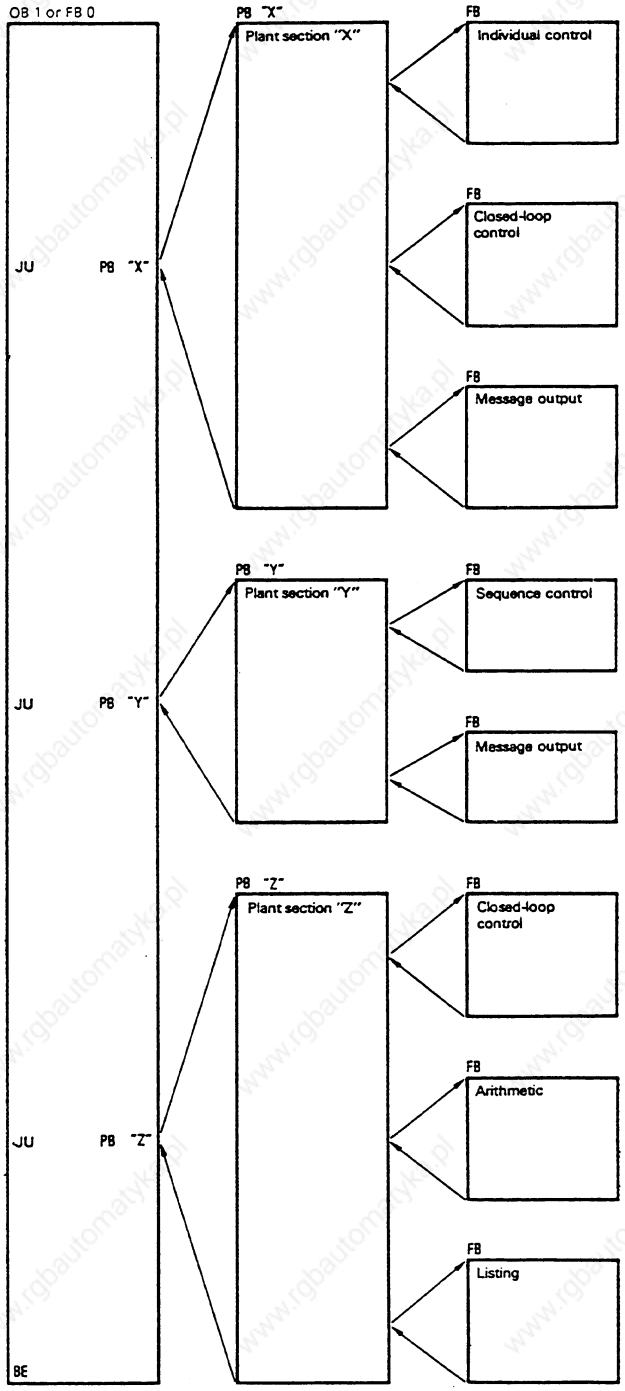


Fig. 15 Basic structure of the user program related to system structure

- Cyclic program execution interrupts

The cyclic program execution can be interrupted by:

- interrupt-driven program execution (see section 5.5),
- time-driven program execution (see section 5.6),
- programming and device errors (see section 5.7),
- signals from the S5 bus (PEU, NAU, BAU),
- switching the mode selectors on the CPU's or on the COR from "RUN" to "STOP", stop command, stop using the "PC stop" PG function.

Following the interrupt or time-driven execution, the cyclic program execution is automatically continued from the interrupt point. All other types of interrupt result in the CPU stopping.

- Indicators

During error-free cyclic operation, the green "RUN" LED lights up. The red "STOP" LED and the error LED's are unlit.

Note! The contents of the arithmetic registers, accu's 1 to 4 and the values of the result indicators (see section 6.1) are not guaranteed beyond the limits of the cycle, i.e. they must be set up at the beginning of a new cycle and must not be transferred from the previous cycle or from the start-up.

5.5 Programming interrupt-driven processing

Interrupt-driven processing can be carried out with the S5 135 U. In this operating mode, cyclic processing is interrupted at the block boundaries by a signal on an S5 bus interrupt line (see S processor and central controller operating instructions). The system program then calls the organization block OB 2 in which the user can program required reactions. After this program has been executed the processor goes back to the point of interruption and continues the cyclic processing from there.

Process-interrupt processing means the user can react directly to process signals.

- Points of interruption

Cyclic program execution cannot be interrupted at random by interrupt-driven processing. This is normally only possible at the block boundaries. When a change is made from one block to another (by calling in a new block, or by returning to the higher ranking block following a block end statement) the system program can call in an organization block for interrupt-driven processing.

Interrupt-driven processing can only be interrupted by device disturbances, not by time-driven processing, or a renewed request from the interrupt-driven processing. Further interrupt requests will only be accepted after OB 2 has been processed. During the next block change, the cyclic processing is interrupted again by the interrupt-driven processing.

- Reaction time

There can be no interrupt-driven processing while a block is being processed. If an interrupt occurs, it will only be processed during a block change, i.e. when a block is called or ended. Therefore the maximum reaction time between the occurrence and the processing of an interrupt corresponds to the processing time of the longest block.

- Disabling interrupt-driven processing

An interrupt-driven program is inserted into the cyclic program at a block boundary . At this point, the cyclic program will be interrupted. This interruption can have a negative effect, if a cyclic program section is time-critical, when e.g. a particular reaction time must be achieved.

If a program section must not be interrupted by interrupt-driven processing, the following programming options are possible:

- The program does not contain a block change. Therefore it cannot be interrupted.
- The program itself is in an interrupt-driven program. Here, too, it cannot be interrupted during a block change.
- Interrupt processing is disabled with the STEP 5 command IA. With the RA command the interrupt processing is enabled again. The program section between IA and RA cannot be interrupted by interrupt-driven processing.

5.6 Programming time-driven processing

The S5 135 U can also carry out time-driven program processing. Time-driven processing is carried out if a signal coming from an "inner clock" causes the CPU to interrupt normal cyclic processing and to call organization block OB 13, by means of which the user can bring about time-driven processing of a program.

After this program has been executed, the processor returns to the point of interruption in the cyclic program and continues processing from there. If no OB 13 is programmed, the cyclic program will not be interrupted.

- Interface between system program and time-driven processing

OB 13 is the interface between the system program and the time-driven processing. It is called every 100 ms by the system program.

- Points of interruption

Cyclic program processing can only be interrupted at the block boundaries by time-driven processing. Time-driven processing can be interrupted both by interrupt-driven processing at the block boundaries as well as by device faults, at any time, but never by a renewed time-driven processing request.

If after 100 ms a second request comes from OB 13, before the first has finished being processed, the CPU stops and outputs an error message (WECKFE, see section 5.7). The execution of the whole time-driven program, including process interrupts if applicable, must be completed within this time.

- Reaction time

There can be no time-driven processing while a block is being processed. Therefore a time-driven program will only be called when a block is called or ended. Thus, the maximum reaction time between the occurrence and the processing corresponds to the processing time of the longest block. If there are still process interrupts waiting at this point, the time-driven program will only be processed when all the outstanding process interrupts have been serviced. The maximum reaction time between the occurrence and the execution of the time-driven program increases in this case by the processing time required by the process interrupts.

5.7 Evaluating a device or programming error

The system program can recognise when the CPU is operating incorrectly, errors in the system program or the effects of incorrect programming by the user:

- Calling in a block which is not loaded
- Acknowledgement delay (QVZ) with direct access to I/O modules in the P or O areas or other S5 bus addresses
- Acknowledgement delay during updating of process image or during IPC flag transmission
- Addressing error (ADF)
- Scan time exceeded (ZYK)
- Substitution error
- Command code error
- Time interrupt error with time-controlled program processing
- Special function error
- DB 1 error
- Block stack or interrupt stack overflow

The CPU stops and the following happens:

- The "STOP" LED on the CPU which caused the error flashes slowly. In multiprocessor operation, the **other** CPU's are put into stop status, and their "STOP" LED's are lit continuously (exceptions, see section 5.2.3).
- The digital outputs are disabled by the output of the BASP signal (exceptions, see section 5.2.3).
- Organization block OB 28 is called (not if there is a stack overflow or a DB 1 error).

The transition to the stop status is carried out irrespective of whether and how OB 28 has been programmed.

Device errors which occur while OB 28 is being processed are registered by the system program in the same way as in the cyclic program. The CPU then stops immediately without calling OB 28. When the CPU is in the stop status the OB 28 call is also implemented by means of the PG function "PC stop", switching the mode selector (from "RUN" to "STOP"), a stop command or a stop signal. In addition, QVZ, ADF and ZYK are indicated by the LED's on the CPU front panel.

During the stop status which then follows, the cause of interruption can be analysed via the PG with the aid of the indications (control bits and interrupt stack) shown on the following pages. The control bits indicate the sequence which has been run through or the current operating status, causes of errors if applicable, and the permitted start-up mode. In the interrupt stack, the point of interruption in the program is specified along with the current statuses and accumulator contents.

In addition to the information in the interrupt stack, disturbances occurring during the initialization, start-up or program execution phases are each defined in more detail in system data 3 and 4 (see section 10).

C O N T R O L B I T S (output with the PG)

PRISTP	<======	MAFEHL	BARBEND	PGSTP	STPS	STPBEF	HALT
ANLAUF	<======	NEUST	M W A X	A W A		NEU-ZUL	MWA-ZUL
ZYKLUS X	<======	SIPROZ X	BARB	OB1GEL X	FBOGEL	OBPROZA	OBWECKA X
32KRAM	16KRAM	8KRAM X	EPROM	KM-AUS	KM-EIN	DIGEIN X	DIGAUS X
URGELO	URLOIA	VERURS	ANL-ABB	UA-PG	UA-SYS		1. UA
CHS-FE	BAT-FE	AWM-FE	RAM-FE	DBO-FE	DB1-FE	DB2-FE	KOR-FE
N A U	P E U	B A U	P A A	ZYK	Q V Z	A D F	WECKFE
OPC-FE	PAR-FE	BSTNIZ	BSTNIG	SF-NIG	SF-LZF		TI-OUT

PRISTP Processor in stop status

MAFEHL Machine error

BARBEND Stop status following PG function "program check end"

PGSTP Stop by PG function "PC stop"

STPS Mode selector on "STOP"

STPBEF Stop command executed

HALT Halt active = CPU stopped by the COR or by another CPU

ANLAUF CPU starting up

NEUST Cold restart carried out or active

M W A Manual warm restart carried out or active

A W A Automatic warm restart following NAU carried out or active

NEU-ZUL Cold restart permissible

MWA-ZUL Manual warm restart permissible

ZYKLUS Cyclic program execution active

SIPROZ Single processor operation

BARB PG function "program check" active

OB1GEL OB 1 loaded. Cyclic program execution is determined by OB 1

FBOGEL FB 1 loaded. The cyclic program execution is determined by FB 0, if **no** OB 1 is loaded. If FB 0 **and** OB 1 are loaded, only OB 1 is processed cyclically

OBPROZA	OB 2 process interrupt organization block loaded, i.e. process interrupt processing possible (see section 5.5)
OBWECKA	OB 13 time-interrupt organization block loaded, i.e. 100 ms time-interrupt processing possible (see section 5.6)
32KRAM	User memory module is a RAM with 32 x 2^{10} words
16KRAM	User memory module is a RAM with 16 x 2^{10} words
8KRAM	User memory module is a RAM with 8 x 2^{10} words
EPROM	User memory module is an EPROM
KM-AUS	Address list for IPC flag outputs present
KM-EIN	Address list for IPC flag inputs present
DIGEIN	Address list present for digital inputs
DIGAUS	Address list present for digital outputs
URGELO	Overall reset of CPU carried out
URLOIA	Overall reset being carried out
VERURS	CPU has caused the PC to stop
ANLABB	Termination during the start up
UA-PG	Overall reset request by the PG
UA-SYS	Overall reset request by the system program, (overall reset must be carried out)
1. UA	First overall reset request (= preparation for the overall reset by switch operation)
CHS-FE	Cross-checksum error occurred during checking of the system program PROM's
BAT-FE	Fault in battery back-up (start up not possible)
AWM-FE	Contents of the user memory module not correct (overall reset necessary)
RAM-FE	Contents of system program RAM not correct (overall reset necessary)
DB0-FE	Error during setting up of the block address lists
DB1-FE	Error during setting up of the address lists for process image updating: DB 1 not programmed in multiprocessor operation; or the inputs and outputs specified in DB 1 do not acknowledge on the corresponding modules

DB2-FE	Not occupied
KOR-FE	Error during data exchange with the COR
NAU	Mains power failure
PEU	I/O not operable
BAU	Battery back-up not available
PAA	Acknowledgement delay during updating of process images or during IPC flag transmission
ZYK	Scan time exceeded (see section 1.5.1)
WECKFE	Time-interrupt processing (= time-controlled processing) requested, while the last time-interrupt processing is still active (see section 5.6)
QVZ	Acknowledgement delay
ADF	Addressing error
OPC-FE	Error in command code
PAR-FE	Parameter illegal for this command
BSTNIG	Block called is not loaded
BSTNIZ	Block call illegal
SF-NIG	Special function called does not exist
SF-LZF	Error while a special function was being executed
TI-OUT	Not occupied

```
I N T E R R U P T - S T A C K

TIEFE:    01

BEF-REG:  34C8      SAZ:       004F      DB-ADR:    002C      BA-ADR:    005A
BST-STP:  EAD8      OB-NR.:      28      DB-NR.:       5      PB-NR.:       1
                    REL-SAZ:   0002      DBL-REG:   0000
VEK-ADR:  0000      UAMK:      0200      UALW:      0000

AKKU1: 0000 6789    AKKU2: 0000 6789    AKKU3: 0000 0000    AKKU4: 0000 0000


ERGEBNISANZEIGE:    ANZ1 ANZO OVFL OVFLS ODER STATUS VKE ERAB
                                                      X

STOERUNGSURSACHE:   NAU  PEU  BAU  HALT  ZYK   QVZ   ADF STPS

                    BCF  SUF  TRAF SFF   STUEB STUEU ZFE
                     X


TIEFE:    02

BEF-REG:  6500      SAZ:       005A      DB-ADR:    002C      BA-ADR:    0024
BST-STP:  EADC      PB-NR.:       1      DB-NR.:       5      OB-NR.:       1
                    REL-SAZ:   0004      DBL-REG:   0000
VEK-ADR:  005A      UAMK:      0000      UALW:      0000

AKKU1: 0000 6789    AKKU2: 0000 6789    AKKU3: 0000 0000    AKKU4: 0000 0000


ERGEBNISANZEIGE:    ANZ1 ANZO OVFL OVFLS ODER STATUS VKE ERAB

STOERUNGSURSACHE:   NAU  PEU  BAU  HALT  ZYK   QVZ   ADF STPS
                                                      X
                    BCF  SUF  TRAF SFF   STUEB STUEU ZFE
```

TIEFE Level of the interrupt stack, indicates chronological order of the interrupts which have occurred

BEF-REG Command register, contains machine code (first word) of the next command to be executed or command just executed

BST-STP Block stack pointer

VEK-ADR With PAA errors, indicates the address of the area which is not acknowledging (otherwise like SAC):
E200H to E27FH: digital inputs 128 to 0 (entry 7FH to 0)
E280H to E2FFH: digital outputs 128 to 0 (entry FFH to 80H)
E300H to E3FFH: IPC flag inputs 256 to 0 (entry FFH to 0); to system program version A7 on E000H to E0FFH
E400H to E4FFH: IPC flag outputs 256 to 0 (entry FFH to 0); to system program version A7 on E100H to E1FFH

SAZ Step address counter, specifies absolute address of the command to be executed next or command just executed in the program memory.

FB-NR. Block type and block number of the current block (OB, DB, SB, PB or FB)

REL-SAZ Relative step address counter, specifies address of the command to be processed next or command just executed relative to the start of the current block (relative addresses are displayed by the PG in the "input disable" operating mode, see operating instructions of the PG).

UAMK Interrupt displays mask word.

UALW Interrupt displays erase word

DB-ADR Absolute start address of the currently selected data block in the program memory (address of first data word; 7FFFH, if no DB has been addressed)

DBL-REG Not occupied

BA-ADR Absolute address in the program memory of the last command to be executed in the last block to call (jump command)

OB-NR. Block type and block number of the last block to call (OB, DB, SB, FB, PB)

Result bit: see section 6.1

AKKU1..4 Contents of the accumulators

NAU Mains power failure

PEU I/O's not operable (= power failure in the expansion unit).

BAU Battery back-up not available

HALT Halt signal line activated (CPU stopped by the COR, or by another CPU)

ZYK Scan time exceeded

QVZ Acknowledgement delay

ADF Addressing error

STPS Mode selector set to "STOP" or stop command or stop caused by PG

BCF Command code error

SUF Substitution error

TRAF Not occupied

SFF Special function group error (corresponds to SF-NIG or SF-LZF with control bits)

STUEB Block stack overflow - nesting depth of 24 blocks has been exceeded

STUEU Interrupt stack overflow

ZFE Time error (time-interrupt error or clock pulse failure at the CPU)

- Error description

QVZ Acknowledgement delay: addresses which were addressed via the S5 bus do not acknowledge on the correspoding modules, e.g.:

  - direct access using load/transfer commands L/T, PB, PW, OB, OW to inputs or outputs which do not acknowledge
  - acknowledgement delay during updating of process image (see section 6.2) or during IPC flag transmission (see section 1.4.1)

ADF Addressing error: the process image (see section 6.2) has been addressed (load/transfer commands L/T IB, IW. ID, QB, QW, QD or binary logic or memory operations) with inputs or outputs which did not acknowledge on the corresponding I/O modules at the last start-up or with outputs which were not specified during the last cold restart without memory recall.

SUF Substitution error: a substitution command is to be executed for which an operation, which is illegal under these circumstances, has been substituted (see section 6.3).

BCF Command code error: a STEP 5 command is to be processed which has an illegal operations list or parameter (also if blocks are called which are not loaded). Example: a timer or counter location is to be processed, which was masked out by means of the timer block length or which has a number greater than 127.

SFF Special function group error: the special function called does not exist, has been assigned incorrect parameters or has been processed incorrectly.

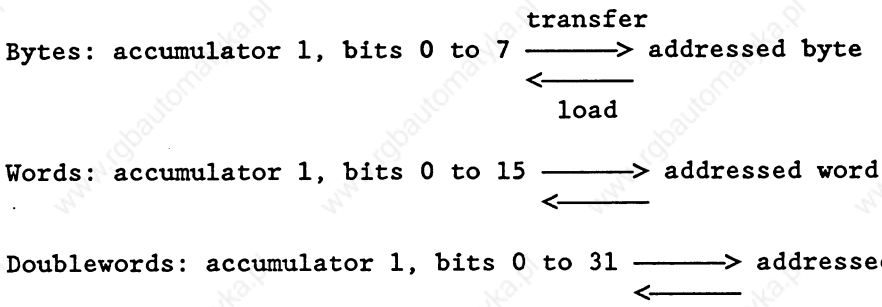
ZFE Time error: with time-driven processing, a second request is recognised after 100 ms, before the first one has finished being processed (see section 5.6). Also occurs if the processor clock pulse fails on the CPU.

6 STEP 5 command set with programming examples

6.1 General rules

The majority of STEP 5 operations use two registers (32 bits) as source for the operands and as destination for the results. These are accumulators 1 and 2.

Depending on the method of addressing (in bytes, words or doublewords), load and transfer commands use the contents of accumulator 1 as follows:



Accumulator 1 is always the destination of the load operation and source of the transfer operation.

With **byte** or **word load** operations, the more significant bit positions of the least significant word in accu 1, which are not used, are always filled with zeros. Before the contents of the address which has been referenced are loaded in accu 1, the "old" contents of the least significant word are transferred from accu 1 to accu 2.

With **byte** or **word load** operations, the most significant word in accumulators 1 and 2 remains unchanged. With **doubleword load** operations, the whole content of accu 1 is transferred to accu 2 before loading.

With transfer instructions accumulator 1 and accumulator 2 remain unchanged. The auxiliary registers (accus 3 and 4) remain unchanged during all load and transfer instructions.

- Numeric notation

Numbers in various notations can be used as operands for the STEP 5 commands, which logically operate on, change or compare the contents of accumulators 1 and 2. Depending on the operation to be executed, the contents of accumulators 1 or 2 will be interpreted as one of the following notations:

a) **Fixed point number**: is interpreted as a 16-bit binary number in two's complement notation (fixed point expansion from 16 to 32 bits, see section 7.5). Range of numbers: -32768 to +32767. Example (loading a fixed point number): L KF -12876.

b) **BCD number**: with sign and 3 figures

Assignment in accu 1:

Bits	15 to 12	11 to 8	7 to 4	3 to 0
	sign	hundreds	tens	ones

The individual figures are positive 4-bit binary numbers in two's complement notation.

Sign: 0000 if the number is positive
1111 if the number is negative

Range which can be shown: -999 to +999

c) **Floating point number**: is interpreted as a 32-bit binary number with an 8-bit exponent and a 24-bit mantissa. With the +G, -G, xG and :G floating point operations, a 16-bit mantissa is recognised in the S processor; the 8 least significant bits are set to zero.

Examples: (input of N floating point numbers with the PG)

N = 12.34567

L KG + 1234567 + 02

mantissa; exponent (base 10)

$N = +0.1234567 \times 10^{+2} = 12.34567$

N = -0.005

L KG - 500000000 - 02

mantissa; exponent (base 10)

$N = -0.5 \times 10^{-2} = -0.005$

Range which can be represented: $+0.1469368 \times 10^{-38}$ to $+0.1701412 \times 10^{39}$ and $-0.1469368 \times 10^{-38}$ to $-0.1701412 \times 10^{39}$

Note: the internal notation need not correspond to the format in which the numbers are input during the creation of a program using the PG. (See operating instructions of the PG). The PG generates the notations shown above.

• Result bits

There are commands for processing information consisting of individual bits and commands for processing information consisting of words (8, or 16 bits).

In both groups there are commands which set condition codes and commands which interpret these codes (see section 8 operation list, influencing condition codes). There are bit condition codes and word condition codes which correspond to the command groups. The condition code byte can be displayed by the PG and appears as follows:

	Word condition codes				Bit condition codes			
	CNC1	CNC0	OV	OS	OR	STA	RLO	$\overline{\text{ERAB}}$
Bit	7	6	5	4	3	2	1	0

Jump operations are available for the immediate interpretation of the codes (section 4.3).

Explanation of the bit condition codes:

ERAB First scanning cycle; a logic operation starts. At the end of a logic operation chain (e.g. memory operations) ERAB is set to 0. Commands which set ERAB to 0 (e.g. result assignment = Ax.x), limit the RLO (see column in operations list, section 8), i.e. the logic operation result can be further interpreted (e.g. by RLO-dependent commands), but not however further operated on. Only after the first logic operation statement (= first scanning cycle) will the RLO be re-established.

RLO Result of logic operation; the result of bit-wide logic operations. Truth statement in the case of compare commands (see appendix: operation list, binary logic operations or compare operations).

STA Status; with bit commands specifies the logical status of the bit which has just been scanned or set. The status is updated with binary logic operations [except for A(, O(,), O] and with memory operations.

OR Or; informs the CPU that the following AND logic operations must be handled before an OR logic operation (AND before OR).

Explanation of the word condition codes;

OV Over; specifies whether during the arithmetic operation just completed, the permissible numerical range was exceeded.

OS Over latching; the over bit is latched. This is used to indicate whether at some stage during several arithmetic operations an error has occurred caused by overflow.

CNC1 and CNC0 are coded result bits, which are interpreted according to the following table.

Word result bits		Result of fixed point calculation	Digital logic opera-tions	Comparison of contents of accu 1 with accu 2	Shifting: last bit shifted
CNC1	CNC0				
0	0	result = 0	= 0	accu 2 = accu 1	0
0	1	result < 0	–	accu 2 < accu 1	–
1	0	result > 0	≠ 0	accu 2 > accu 1	1

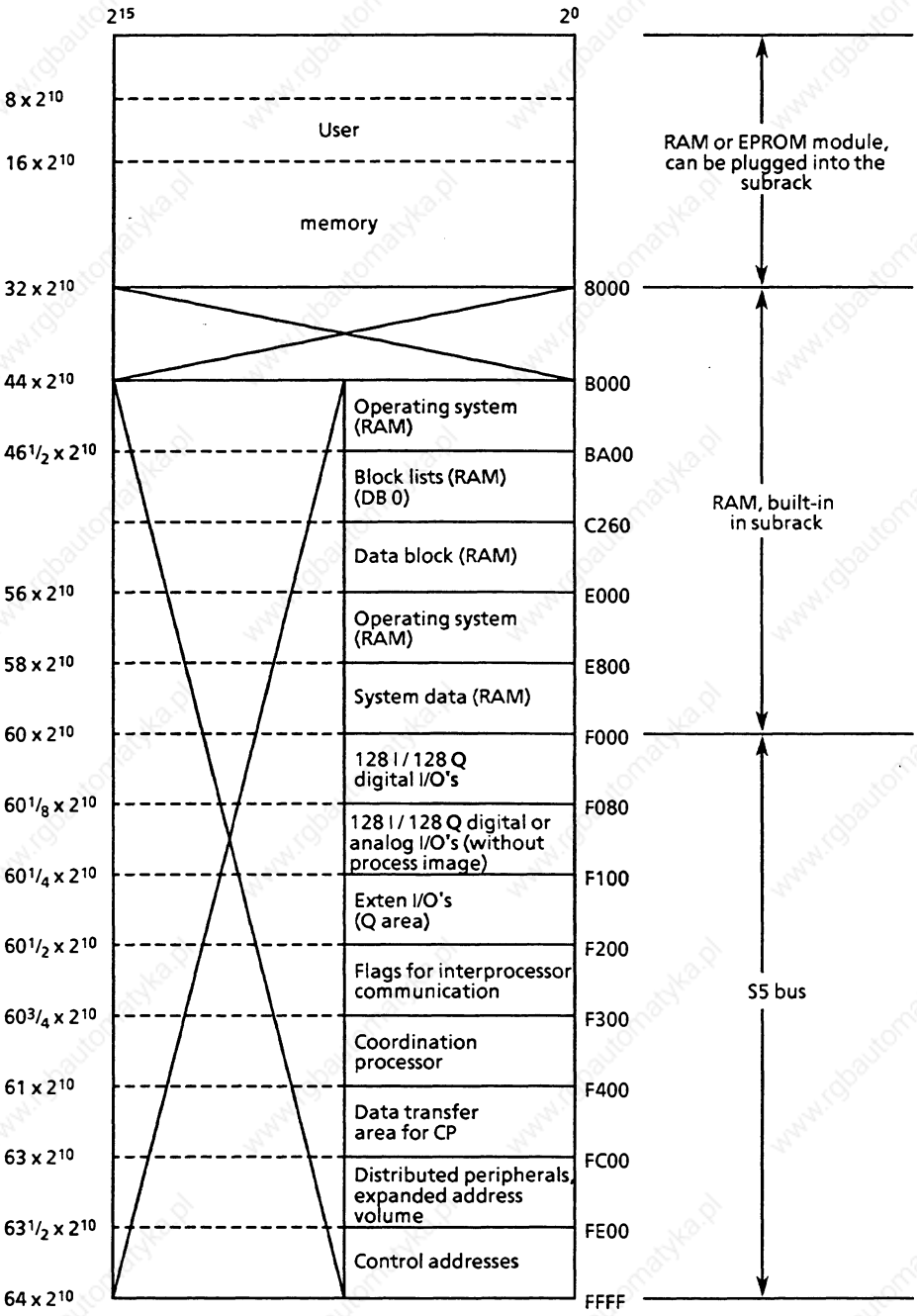


Fig. 16 Memory distribution in the S processor

- Address areas for I/O/programming

Area (absolute address)		Is addressed with	Parameters	Address area
EF00 EF7F	PII (process image inputs)	L IB/T IB L IW/T IW L ID/T ID A I/AN I/O I/ON I S I/R I/= I	0 to 127 0 to 126 0 to 124	CPU RAM
EF80 EFFF	PIO (process image outputs)	L QB/T QB L QW/T QW L QD/T QD A Q/AN Q/O Q/ON Q S Q/R Q/= Q	0 to 127 0 to 126 0 to 124	
F000 F07F	Digital I/O's	L PB/T PB L PW/T PW	0 to 127 0 to 126	
P-I/O's with process image				
F080 EF7F	Digital or analog I/O's	T PB/T PB T PW/T PW	128 to 255 128 to 254	S5 bus
P-I/O's without process image				
F100 F1FF	Extended I/O area	L OB/T OB L OW/T OW	0 to 255 0 to 254	
Q-I/O's				

Fig. 17 Address area allocation in the S5 135 U

6.2 Basic operation set

- Binary logic operations

Operation	Parameter	Function
) A (O (O		Close brackets ANDing expressions in brackets ORing expressions in brackets ORing AND functions
A O		AND function OR function
I	0.0 to 127.7	with scanning of an input for signal status 1
Q	0.0 to 127.7	with scanning of an output for signal status 1
F	0.0 to 255.7	with scanning of a flag for signal status 1
D	0.0 to 255.15	with scanning of a data word for signal status 1
NI	0.0 to 127.7	with scanning of an input for signal status 0
NQ	0.0 to 127.7	with scanning of an output for signal status 0
NF	0.0 to 255.7	with scanning of a flag for signal status 0
ND	0.0 to 255.15	with scanning of a data word for signal status 0
T	0 to 127	with scanning of a timer for signal status 1
NT	0 to 127	with scanning of a timer for signal status 0
C	0 to 127	with scanning of a counter for signal status 1
NC	0 to 127	with scanning of a counter for signal status 0

Binary logic operations generate the result of logic operation (RLO) as their result.

At the start of a logic operation sequence the results from the first logic operation (first scan) are only dependent on the status of the scanned signal and whether or not it is negated (N = negation); they are not, however, dependent on the type of logic operation (O = OR, A = AND).

During a logic operation sequence, the RLO is formed from the type of logic operation, the previous RLO and the status of the scanned signal. A logic operation sequence is completed by an RLO-limiting (ERAB = 0) command (e.g. memory operations).

The RLO remains unchanged until the next "first scan". It can be interpreted, but cannot be further operated on. In order to interpret the RLO, RLO-dependent commands can be used (see column in operations list).

Program	Status	RLO	$\overline{\text{ERAB}}$	
:				RLO limited, end of logic operation sequence
= Q 0.0	0	0	0	
A I1.0	1	1	1	first scan
A I1.1	1	1	1	
A I1.2	0	0	1	
= Q 0.1	0	0	0	RLO limited, end of logic operation sequence

- Memory operations

Operation	Parameters	Function
S		Set
R		Reset
=		Assign
I	0.0 to 127.7	an input in the PII
Q	0.0 to 127.7	an output in the PIO
F	0.0 to 255.7	a flag bit
D	0.0 to 255.15	a data word bit

- Loading, transfer and compare functions

Operation	Parameters	Function
L		Load
T		Transfer
IB	0 to 127	an input byte from/to the PII
IW	0 to 126	an input word from/to the PII
ID	0 to 124	an input doubleword from/to the PII
QB	0 to 127	an output byte from/to the PIO
QW	0 to 126	an output word from/to the PIO
QD	0 to 124	an output doubleword from/to the PIO
FB	0 to 255	a flag byte
FW	0 to 254	a flag word
FD	0 to 252	a flag doubleword
DR	0 to 255	data (right byte) from a DB
DL	0 to 255	data (left byte) from a DB
DW	0 to 255	a data word from a DB
DD	0 to 254	a data doubleword from a DB
PB	0 to 127	a peripheral byte of the digital inputs or outputs (P area)
PB	128 to 255	a peripheral byte of the analog or digital inputs or outputs (P area)
OB	0 to 255	a byte of the extended I/O area (O area)

• Loading, transfer and compare functions (continued)

Operation	Parameters	Function
L T		Load Transfer
PW	0 to 126	a peripheral word of the digital inputs or outputs (P area)
PW	128 to 254	a peripheral word of the analog or digital inputs or outputs (P area)
OW	0 to 254	a word of the extended peripherals (O area)
L		Load
KM	16 bit pattern	a constant as bit pattern
KH	0 to FFFF	a constant in the hexadecimal code
KF	−32 768 to +32 767	a constant as a fixed point number
KY	0 to 255 for each byte	a constant, 2 bytes
KB	0 to 255	a constant, 1 byte
KS	2 alphanumeric characters	a constant, 2 ASCII characters
KT	0.0 to 999.3	a timer value (constant)
KC	0 to 999	a counter value (constant)
KG	[1]	a constant as a floating point number (32 bit)
T	0 to 127	a timer value
C	0 to 127	a counter value
LD T	0 to 127	BCD loading of a timer value
LD C	0 to 127	BCD loading of a counter value
! =		compare for equal to
> <		compare for not equal to
>		compare for greater than
> =		compare for greater than or equal to
<		compare for less than
< =		compare for less than or equal to
F		two fixed point numbers (16 bit)
D		two fixed point numbers (32 bit)
G		two floating point numbers (32 bit)

The loading and transfer operations do not influence the condition codes. The compare commands generate the RLO and the CNC1 and CNC0 word condition codes as the result. The contents of accumulators 1 and 2 are always compared (see program examples and operations list).

For loading and transfer operations the instructions in section 6.1 should be noted. The I/O's can be addressed directly by loading and transfer operations - with L/T PB, PW, OB, OW or by means of

[1] $\pm 0.1469368 \times 10^{-38}$ to $\pm 0.1701412 \times 10^{39}$

Process image - with L/T IB, IW, ID, QB, QW, QD and with logic operations. With T PB 0 to 127 and T PW 0 to 126 the PIO will be maintained at the same time. (PII/PIO = process image of the inputs/outputs for 128 input/output bytes of the P I/O's with byte addresses from 0 to 127).

The process image represents a memory area, the contents of which are only output to the peripherals (PIO) or read in by the peripherals (PII) once per user program cycle (see Fig. 13). This avoids frequent changing of the logic status of a bit within a program cycle, which leads to "chattering" of the corresponding peripheral output.

The O area can only be addressed via the 300 and 301 interface modules, so that I/O modules with addresses in the O area can only be plugged into expansion units. For the whole O area and P area with relative byte addresses from 128 to 255, there is no process image.

With word loading and word transfer operations to address areas organized in bytes (PII, PIO, flags, S5 bus), byte n and byte n + 1 will be loaded/transferred; with doubleword operations byte n to byte n + 3 will be loaded/transferred.

Example

L IW 5 bytes 5 and 6 of the PII will be loaded into accu 1.

L FD 10 flag bytes 10 to 13 will be loaded.

- Timer and counter operations

In order to load a timer using a start command, or a counter using a set command, the value must be loaded into accumulator 1 beforehand.

The following loading operations are recommended:

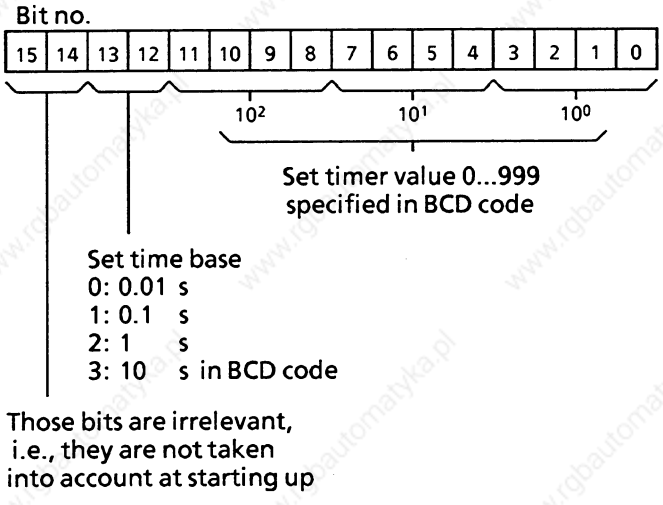
for timers: L KT, L IW, L QW, L FW, L DW
for counters: L KC, L IW, L QW, L FW, L DW.

Operation		Parameters	Function
SP	T	0 to 127	starting a timer as a pulse
SE	T	0 to 127	starting a timer as an extended pulse
SR	T	0 to 127	starting a timer as an "ON" delay
SS	T	0 to 127	starting a timer as a latching "ON" delay
SF	T	0 to 127	starting a timer as an "OFF" delay
R	T	0 to 127	resetting a timer
S	C	0 to 127	setting a counter
R	C	0 to 127	resetting a counter
CU	C	0 to 127	incrementing a counter
CD	C	0 to 127	decrementing a counter

When the SP, SR, SE, SS, SF and S timer or counter operations are carried out, the value in accumulator 1 will be fetched into the timer or counter location (corresponds to the transfer command) and the corresponding operation will be started.

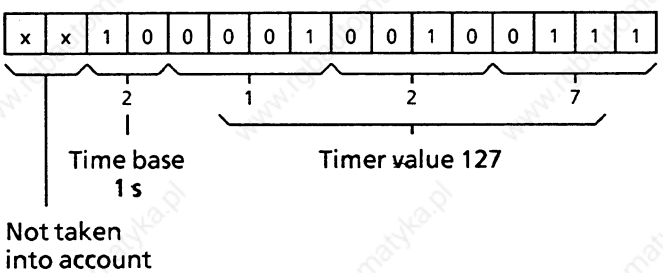
If the time value or count value is loaded using IW, QW, FW or DW, the corresponding word must have the following structure:

For the timer value

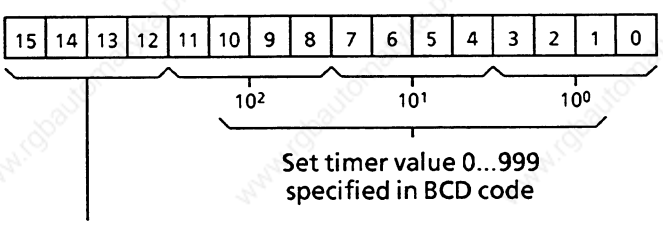


Example: setting a time of 127 s

Bit assignment:

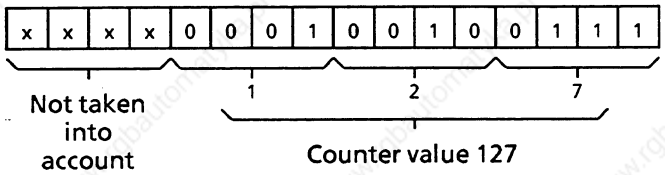


For the counter value



Example: setting a count value of 127.

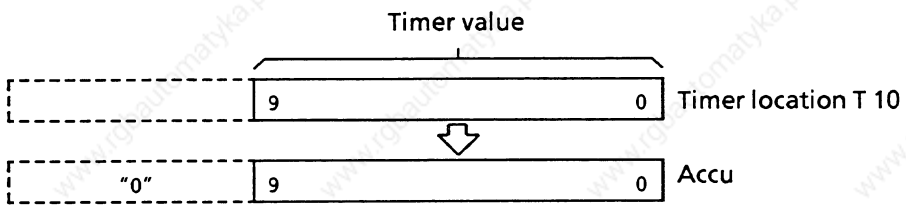
Bit assignment:



The timer or counter value is stored in the timer or counter location and is binary coded. In order to scan the timer or the counter, the value in the timer or counter location can be loaded into accumulator 1 directly or in BCD.

Example:

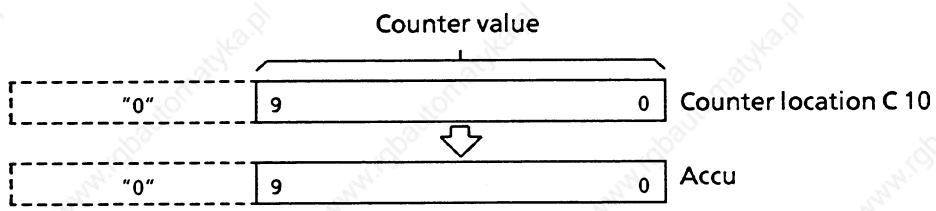
Direct loading of timer values



L T 10 Directly loading the binary value of the T 10 timer into the accumulator

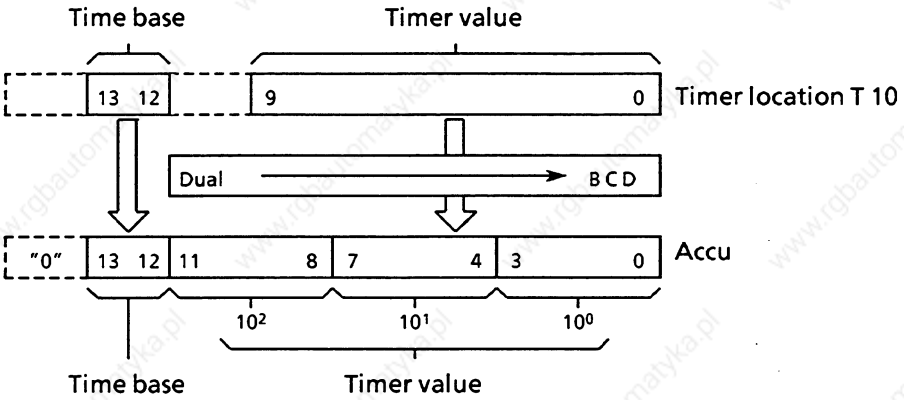
The time base is not loaded here

Direct loading of counter values:



L C 10 Loading the counter value of the C 10 counter directly into the accumulator

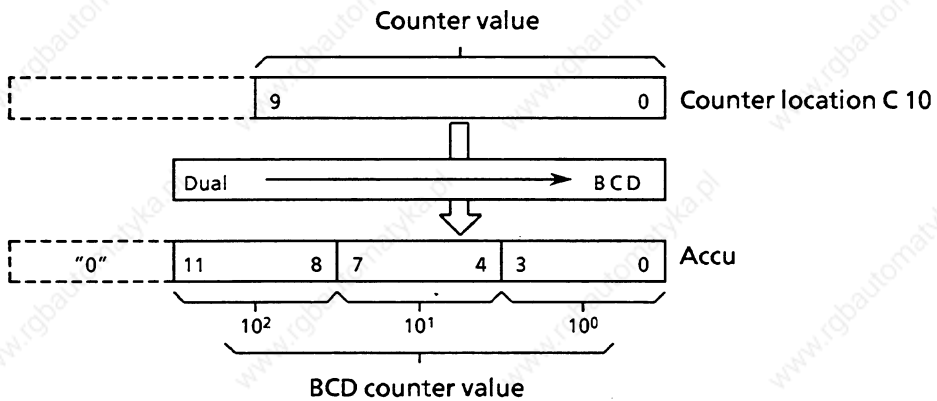
Coded loading of timer values:



LD T 10 Coded loading of the timer value and the time base of the T 10 timer into the accumulator

The time base is also loaded.

Coded loading of counter values:



LD C 10 Coded loading of the counter value of the C 10 counter into the accumulator

With coded loading, status bits 14 and 15 of the timer locations or 12 to 15 of the counter locations are not loaded. In their place, there is a 0 in accumulator 1. The value now in the accumulator can be processed further.

- Arithmetic operations

Operation	Parameters	Function
+ F		addition of 2 fixed point numbers
− F		subtraction of 2 fixed point numbers
x F		multiplication of 2 fixed point numbers
: F		division of 2 fixed point numbers
+ G		addition of 2 floating point numbers
− G		substraction of 2 floating point numbers
x G		multiplication of 2 floating point numbers
: G		division of 2 floating point numbers

The arithmetic operations refer to the contents of accumulators 1 and 2 (see operations list, section 8). The result is then available in accumulator 1. The arithmetic registers are changed by an arithmetic operation as follows:

<accu 1>: = result
<accu 2>: = <accu 3>
<accu 3>: = <accu 4>
<accu 4>: = <accu 4>

The previous contents of accumulator 2 are lost.

- Block calls

Operation	Parameters	Function
JU		jump absolute
JC		jump conditional (dependent on the RLO)
OB	1 to 39	to an organization block [1]
OB	40 to 255	to a system program special function [1]
PB	0 to 255	to a program block
FB	0 to 255	to a function block
SB	0 to 255	to a sequence block
C DB	2 to 255	DB data block call
BE		block end
BEC		conditional block end (dependent on the RLO)
BEU		unconditional block end

[1]) From PG 675 software release S0 A03 or S1 A01

- No operation

Operation	Parameters	Function
NOP	0	no operation
NOP	1	no operation
BLD	0 to 255	display construction statement for the PG (is treated as a no operation by the CPU)

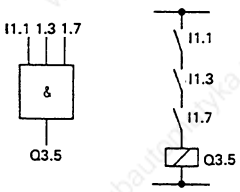
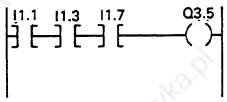
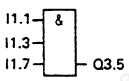
- Stop statement

Operation	Parameters	Function
STP		CPU goes into stop status

Programming examples for logic, memory, timer, counter and compare functions

• Logic functions

AND logic

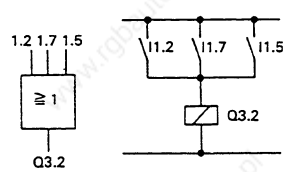
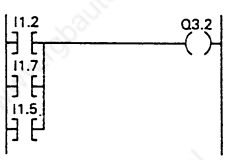
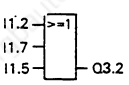
Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.1 A I 1.3 A I 1.7 = Q3.5		

A "1" signal appears at output Q 3.5 when all the inputs have "1" signals simultaneously.

A "0" signal appears at output Q 3.5 if at least one of the inputs has a "0" signal.

There are no restrictions imposed on the number of scans and the programming sequence.

OR logic

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	O I 1.2 O I 1.7 O I 1.5 = Q3.2		

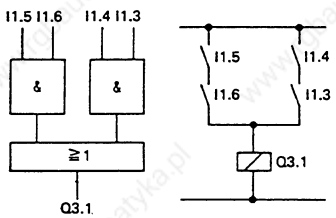
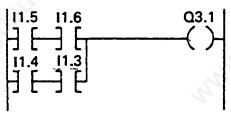
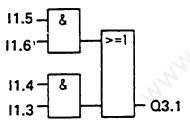
A "1"signal appears at output Q 3.2 if at least one of the inputs has a "1" signal.

A "0" signal appears at output Q 3.2 when all the inputs have "0" signals simultaneously.

There are no restrictions imposed on the number of scans and the programming sequence.

• Logic functions (continued)

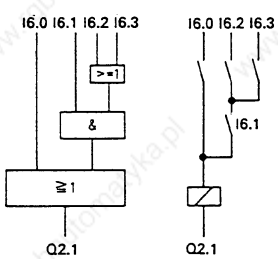
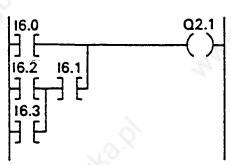
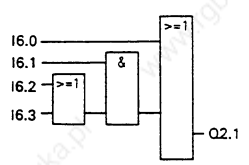
AND before OR logic

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.5 A I 1.6 O A I 1.4 A I 1.3 = Q3.1		

A "1" signal appears at output Q 3.1 when the output of at least one of the AND gates is "1".

A "0" signal appears at output Q 3.1 when neither of the AND gates has a "1" at its output.

OR before AND logic

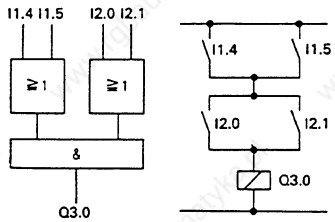
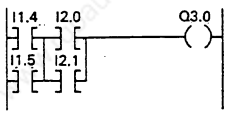
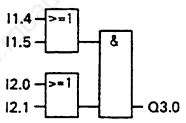
Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	O I 6.0 O A I 6.1 A(O I 6.2 O I 6.3) = Q2.1		

A "1" signal appears at output Q 2.1 when input I 6.0 or input I 6.1 and one of the inputs I 6.2 or I 6.3 have a "1" signal.

A "0" signal appears at output Q 2.1 when input I 6.0 has a "0" signal and the AND gate has a "0" at its output.

• Logic functions (continued)

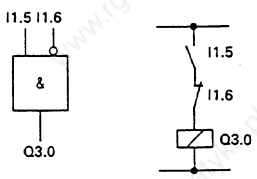
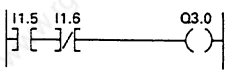
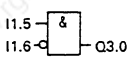
OR before AND logic

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A(O I 1.4 O I 1.5) A(O I 2.0 O I 2.1) = Q3.0		

A "1" signal appears at output Q 3.0 when both OR gates have "1" signal at their outputs.

A "0" signal appears at output Q 3.0 when at least one of the OR gates has a "0" signal at its output.

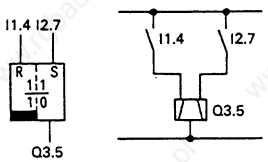
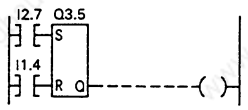
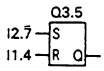
Scanning for "0" signal status

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.5 AN I 1.6 = Q3.0		

A "1" signal appears at output Q 3.0 only when input I 1.5 has a "1" signal (normally open contact actuated) and input I 1.6 has a "0" signal (normally closed contact actuated).

• Memory functions

RS flip-flops for holding signal output

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 2.7 S Q3.5 A I 1.4 R Q3.5		

A "1" signal at input I 2.7 sets the flip-flop, (signal "1" at output Q 3.5).

If the signal at input I 2.7 at input I 2.7 changes to "0", the flip-flop status remains unchanged, i.e. the signal is latched.

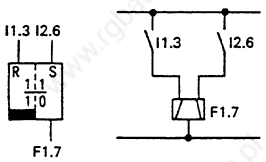
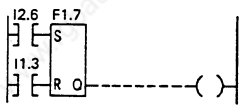
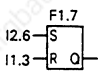
A "1" signal at input I 1.4 resets the flip-flop, signal "0" at output Q 3.5).

If the signal at input I 1.4 changes to "0", the flip-flop status remains unchanged.

If the set signal (input I 2.7) and the reset signal (input I 1.4) appear simultaneously, the scan operation programmed last (in this case A I 1.4) is effective during the processing of the remaining program (reset has priority).

• Memory functions (continued)

RS flip-flop with flags

Original	STEP 5 operation		
	Statement list	Ladder diagram	Control system flowchart
	A I 2.6 S F 1.7 A I 1.3 R F 1.7		

A "1" signal at input I 2.6 sets the flip-flop.

If the signal at input I 2.6 changes to "0" the flip-flop status remains unchanged, i.e. the signal is latched.

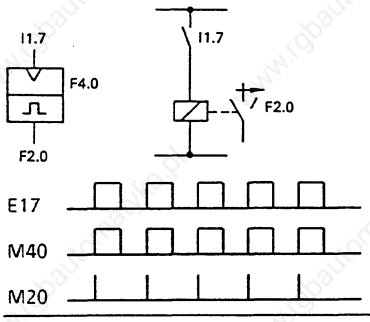
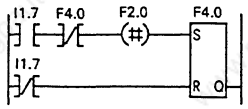
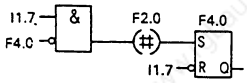
A "1" signal at input I 1.3 resets the flip-flop.

If the signal at input I 1.3 changes to "0", the flip-flop status remains unchanged.

If the set signal (input I 2.6) and the reset signal (input I 1.3) appear simultaneously, the scan operation programmed last (in this case A I 1.3) is effective during the processing of the remaining program (reset has priority).

• Memory functions (continued)

Implementation of an impulse contact

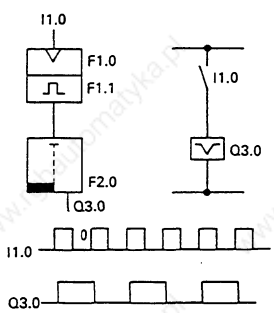
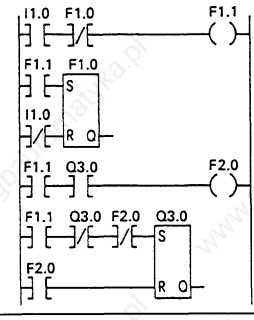
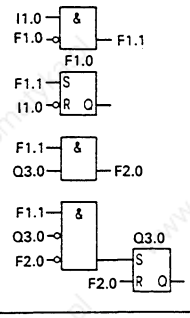
Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.7 AN F 4.0 = F 2.0 A F 2.0 S F 4.0 AN I 1.7 R F 4.0		

The AND logic condition (A I 1.7 and AN F 4.0) is fulfilled at each positive-going edge of the signal at input I 1.7 and flags F 4.0 ("pulse edge flag") and F 2.0 (pulse flag) are set if the RLO = "1"

The AND logic condition A I 1.7 and AN F 4.0 is no longer fulfilled during the next processing cycle since flag F 4.0 has been set.

Flag F 2.0 is reset, i.e. it is "1" during a single program run.

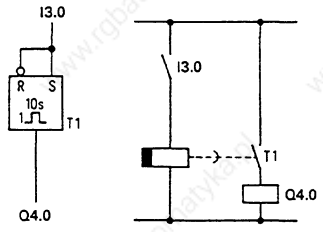
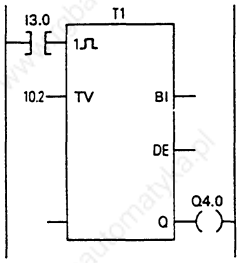
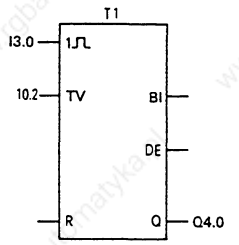
Binary scaler

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.0 AN F 1.0 = F 1.1 A F 1.1 S F 1.0 AN I 1.0 R F 1.0 A F 1.1 A Q 3.0 = F 2.0 A F 1.1 AN Q 3.0 AN F 2.0 S Q 3.0 A F 2.0 R Q 3.0		

Output Q 3.0 of the binary scaler changes its state at each positive-going edge of the signal at input I 1.0, i.e. when input I 1.0 changes from "0" to "1". Consequently, half the input frequency appears at the binary scaler output.

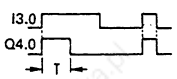
• Timer functions

Pulse

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 3.0 L KT 10.2 SI T 1 A T 1 = Q 4.0		 TV = time value

The timer is started during the first processing cycle if the result of the logic operation is "1". The timer remains unaffected during subsequent processing resulting in a "1" signal.
The timer is set to "0" (reset) if the result of the logic operation is "0".
The AT and OT scans result in a "1" signal as long as the timer is running.

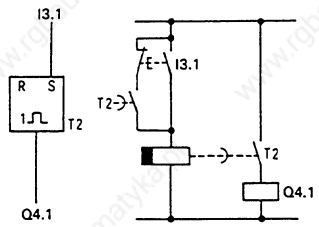
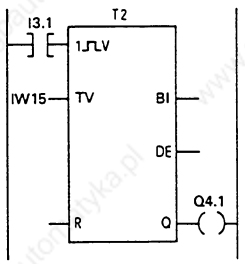
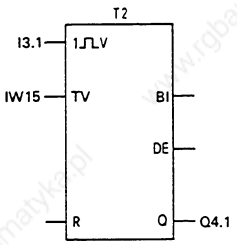
KT 10.2:
The timer is loaded with the specified value (10). The number to the right of the decimal point indicates the time base:
0 = 0.01 s 2 = 1 s
1 = 0.1 s 3 = 10 s



BI and DE are digital outputs of the timer location. The time at output BI is binary code and at DE in BCD with time grid.

• Timer functions (continued)

Extended pulse

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 3.1 L IW15 SE T 2 A T 2 = Q 4.1		

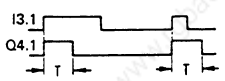
The timer is started during the first processing cycle if the result of the logic operation is "1"

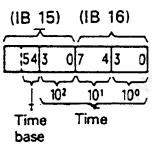
The timer remains unaffected if the result of the logic operation is "0".

The AT or OT scans result in a "1" signal as long as the timer is running.

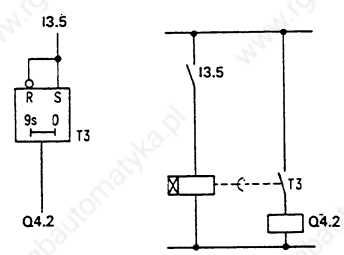
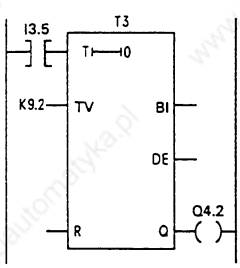
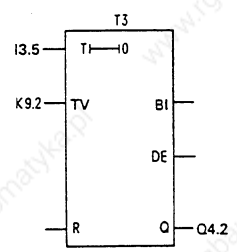
IW 15:

Setting the time with the BCD value of the operands I, Q, F or D (input word I 15 in the example)





On-delay

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 3.5 L KT9.2 SR T 3 A T 3 = Q 4.2		

The timer is started during the first processing cycle if the result of the logic operation is "1". The timer remains unaffected during subsequent processing if the result of the logic operation remains "1".

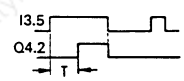
The timer is set to "0" (reset) if the result of the logic operation is "0".

The AT or OT scans result in a "1" signal when the time has elapsed and the result of the logic operation is still present at the input.

KT 9.2:

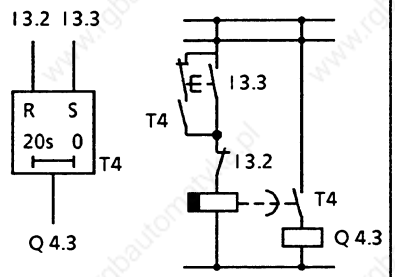
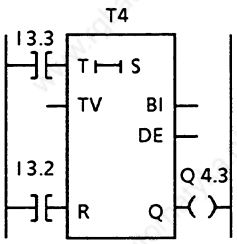
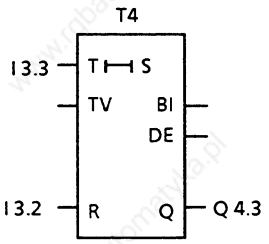
The timer is loaded with the specified value (9). The number to the right of the point indicates the time base:

0 = 0.01 s 2 = 1 s
1 = 0.1 s 3 = 10 s



• Counter function (continued)

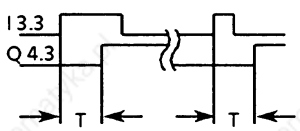
Latching "On" delay

Orginal	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 3.3 L KT 10.2 SS T 4 A I 3.2 R T 4 A T 4 = Q 4.3		

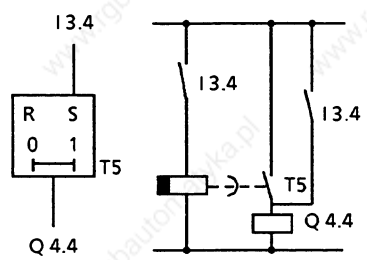
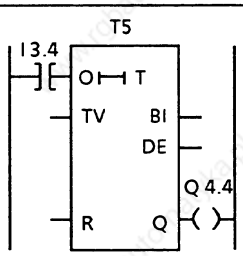
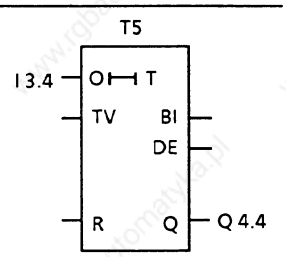
The timer is started during the first processing cycle if the result of the logic operation is "1"

The timer remains unaffected if the result of the logic operation is "0".

The AT o OT scans result in a "1" signal when the time has elapsed. The signal status only changes to "0" when the timer is reset by the RT function.



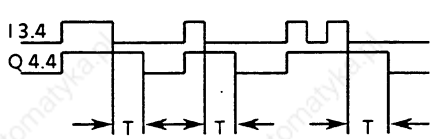
"Off" delay

Orginal	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 3.4 L KT 10.2 SA T 5 A T S = Q 4.4		

The timer is started when the result of the logic operation at the start input changes from "1" to "0". It runs for the time programmed.

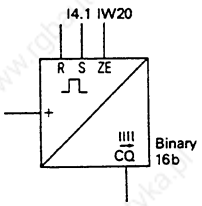
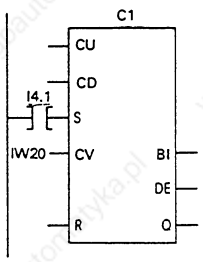
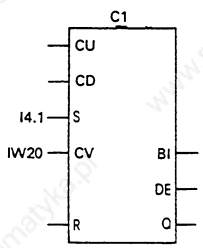
The timer is set to zero (reset) if the result of the logic operation is "1".

The AT or OT scans result in a "1" signal if the timer is running **or** the result of the logic operation is till present at the input



• Counter functions

Set counter

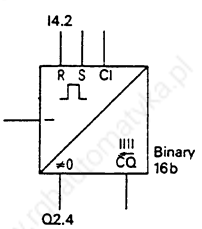
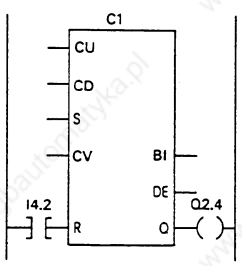
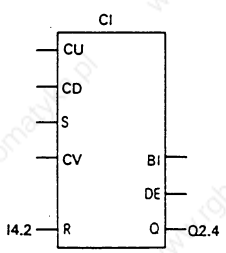
Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 4.1 L IW20 S C 1		

The counter is set during the first processing cycle if the result of the logic operation is "1". The counter remains unchanged during subsequent processing (no matter whether the result of the logic operation is "1" or "0"). The counter is set again (pulse edge evaluation) at the next processing cycle if the result of the logic operation is "1".

The flag necessary for pulse edge evaluation of the set input is included in the counter word.

BI and DE are digital outputs of the counter location. The count values are binary coded at output BI and BCD at output DE.

Reset counter

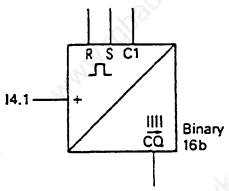
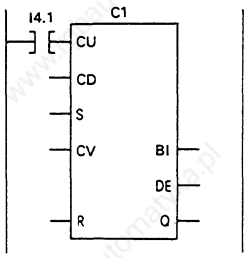
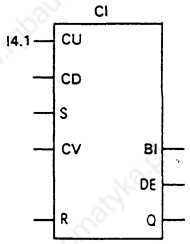
Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 4.2 R C 1 A C 1 = Q 2.4		

The counter is reset when the result of the logic operation is "1".

The counter remains unchanged even if the result of the logic operation becomes "0".

• Counter functions (continued)

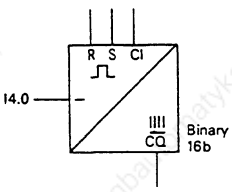
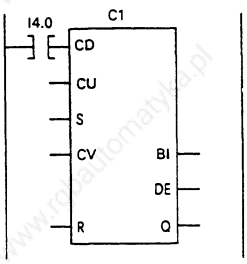
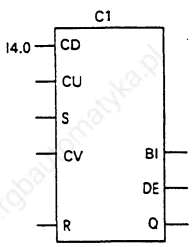
Counting up

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 4.1 CU C 1		

The value of the addressed counter is incremented by 1 up to a maximum of 999. The CU function is effective only on a positive-going pulse edge (from "0" to "1") of the logic operation programmed before CU. The flags necessary for pulse edge evaluation of the counter inputs are included in the counter word.

A counter with two different inputs can be used as an up/down counter by means of the two separate pulse-edge flags for CU and CD.

Counting down

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 4.0 CD C 1		

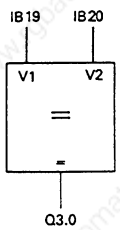
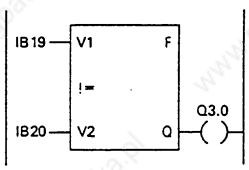
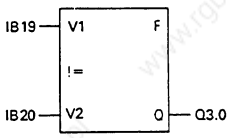
The value of the addressed counter is decremented by 1 to a minimum 0. The CD function is only effective with a positive-going edge (from "0" to "1") of the logic operation programmed before CD.

The flags necessary for pulse edge evaluation of the counter inputs are included in the counter word.

A counter with two different inputs can be used as an up/down counter by means of the two separate pulse-edge flags for CU and CD.

• Compare functions

Comparing for equal to

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	L IB19 L IB20 ! = F = Q 3.0		

The first operand specified is compared with the following operand according to the comparison function.

The comparison produces a binary logic operation result:

RLO = "1": the condition is fulfilled if
accu 1-L = accu 2-L

RLO = "0": the conidtion is not fulfilled, if
Accu 1-L ≠ accu 2-L

The condition codes CNC1 and CNC0 are set as explained in 6.1.

Accu 2-H and accu 1-H remain unaffected during the 16-bit fixed point comparison.

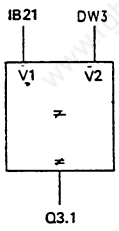
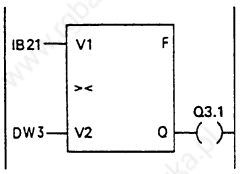
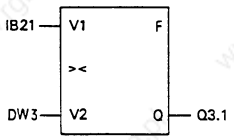
During fixed point comparison (! = F) and floating point comparison (! = G) the total contents of accu 1 and accu 2 (32-bit) are compared with each other.

During the comparison the numerical representation of the operands is taken into account, i.e. the contents of accu 1-L and accu 2-L are interpreted as a fixed point number.

O	IB 19	Accu 2-L
O	IB 20	Accu 1-L

• Compare functions (continued)

Comparing for not equal to

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	L IB21 L IB ><F = Q3.1		

The first operand specified is compared with the following operand according to the comparison function.

The comparison produces a binary logic operation result:

RLO = "1": the condition is if fulfilled if
accu 1-L ≠ accu 2-L

RLO = "0": the conidtion is not fulfilled, if
Accu 1-L = accu 2-L

The condition codes CNC1 and CNC0 are set according to the table on page 54.

Accu 2-H and accu 1-H remain unaffected during the 16-bit fixed point comparison.

During the 32-bit fixed point comparison and the floating point comparison accu 2-H and accu 1-H are involved.

This also applies to comparing for greater than, greater than or equal to, less than and less than or equal to (see operations list).

Then comparing, the numerical representation of the operands is taken into account, i.e. the contents of accu 1-L and accu 2-L are interpreted as a fixed point number.

0	IB 19	Accu 2-L
DW 3		Accu 1-L

6.3 Supplementary operation set

In contrast to the other blocks, function blocks can be programmed with an extended operation set. The entire operation set for function blocks consists of the basic operations and the supplementary operations.

Together with the basic functions and the supplementary functions, the system functions complete the operation set of the STEP 5 programming language.

With the system functions it is possible to intervene in the running of the system program; the memory can be overwritten at any point, and the contents of the working register of the central processor can be changed. Therefore, the system functions should only be used (if at all) with the utmost caution.

The system functions are clearly indicated in the following lists:

Function block operations are only represented in STL. The programs of the function blocks cannot therefore be programmed in graphic form (LAD or CSF).

The following description shows the supplementary operations and system functions which can only be used with function blocks. The possible combinations of substitution operations with actual operands are also given.

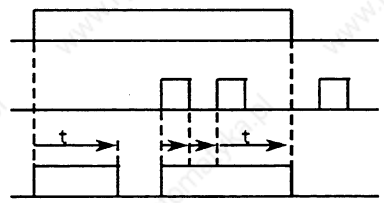
- Binary logic operations

Operations	Description
A =□	AND operation, scanning a formal operand for signal status "1".
AN =□	AND operation, scanning a formal operand for signal status "0"
O =□	OR operation, scanning a formal operand for signal staus "1"
ON =□	OR operation, scanning a formal operand for signal status "0"
	Assign formal operand (see p. 8)
	Inputs, outputs, data and flags addressed in binary code (parameter class I, Q; parameter type BI) and also timers and counters (parameter class T, C) are permitted as actual operands.

- Memory functions

Operation	Description
S =□	Set (binary) formal operand.
RB =□	Reset (binary) formal operand.
= =□	Assign result of logic operation to formal operand.
	Assign formal operand.
	Inputs, outputs, data and flags addressed in binary code (parameter class I, Q, parameter type BI) are permitted as actual operands.

• Timer and counter functions

Operation	Description
FR T 0 to 127	Enabling a timer for restart The operation is only carried out on the leading edge of the result o the logic operation. The timer is restarted if the RLO is "1" at the time of the start operation. 
FR C 0 to 127	Enabling a counter The operation is only carried out on the leading edge of the result of the logic operation. The counter is set (counting up or down) if the result of the logic operation is "1" at the corresponding operation.
FR = ☐	Enabling a formal operand for a restart (for description see FRT or FRC depending on formal operand; parameter class: T, C).
RD = ☐	Resetting (digital) a formal operand (parameter class: T, C).
SP = ☐	Starting a timer, specified as a formal operand, as a pulse with the value stored in the accumulator (parameter class: T).
SR = ☐	Starting a timer, specified as a formal operand, as an on-delay with the value stored in the accumulator (parameter class: T).
SEC = ☐	Starting a timer, specified as a formal operand, as an extended pulse with the value stored in the accumulator or setting a counter specified as a formal operand for the count value stored in accu 1 (parameter class: T, C)
SSU = ☐	Starting a timer, specified as a formal operand, as a latching on-delay with the value stored in the accu or incrementing a counter specified as a formal operand (parameter class: T, C).
SFD = ☐	Starting a timer, specified as a formal operand, as an off-delay with the value stored in the accu or decrementing a counter specified as a formal operand (parameter class: T, C).
	Enter formal operand
	Timers and counters are permitted as actual operand. Exceptions: SP and SR (only timers). The timer or counter value can be assigned as with basic operations: or as a formal operand it can be assigned as follows: Set the timer or counter value with the BCD value, of the IW, QW, FW, DW operands specified as formal operands (parameter class: I, parameter type: W) or as a constant (parameter class: D, parameter type: KT, CC).

Examples

Function block call	Program in function block	Executed program
: JU FB203 NAME : EXAMPLE ANNE : I 10.3 BERT : T 17 FRED : Q 18.4	:A =ANNE :L KT 010.2 :SSU =BERT :A =BERT := =FRED	:A I 10.3 :L KT 010.2 :SS T 17 :A T 17 := Q 18.4
: JU FB204 NAME : EXAMPLE RUTH : I 10.5 PETE : I 10.6 MAUD : I 10.7 DORA : C 15 EMMA : F 58.3	:A =RUTH :SSU =DORA :A =PETE :SFD =DORA :A =MAUD :L KC100 :SEC =DORA :AN =DORA := =EMMA	:A I 10.5 :CU C 15 :A I 10.6 :CD C 15 :A I 10.7 :L KC 100 :S C 15 :AN C 15 := F 58.3
: JU FB205 NAME : EXAMPLE BILL : I 10.4 CARL : T 18 EGON : IW20 DAVE : F 100.7	:A =BILL :L =EGON :SEC =CARL :A =CARL := =DAVE	:A I 10.4 :L IW 20 :SF T 18 :A T 18 := F 100.7

- Loading and transfer functions

Operation	Description
L = ☐	Loading of a formal operand The value of the operand specified as a formal operand is loaded into the accumulator (parameter class: I, T, C, Q; parameter type: BY, W, D)
LD = ☐	Coded loading of a formal operand. The value of the timer or counter location specified as a formal operand is loaded in BCD into the accumulator (parameters: T, C).
LW = ☐	Loading the bit pattern of a formal operand. The bit pattern of the formal operand is loaded into the accumulator (parameter class: D; parameter type: KF, KH, KM, KY, KS, KT, KC).
xxxx = ☐	Loading the bit pattern of a formal operand. The bit pattern of the formal operand is loaded into the accumulator (parameter class: D; parameter type: KG).
T = ☐ Enter formal operands	Transferring to a formal operand. The accumulator contents are transferred to the operand specified as a formal operand (parameter class: I, Q; parameter type: BY, W, D).

Operands corresponding to the basic operations are permitted as actual operands. For LW, data is permitted in the form of a binary (KM) or hexadecimal (KH) pattern, 2 numbers in bytes (KY), chararacters (KC), fixed point number (KF), time values (KT) and count values (KC). For LD, a floating point number is permitted as data.

Operation	Parameters	Description
L RI	0 to 255	Loading a word into accu 1 from the "interface data" area
L RS [5]	0 to 255	Loading a word into accu 1 from the "system data" area
T RI [1]	0 to 255	Transferring accu 1 to a word from the "interface data" area
T RS [5]	0 to 255	Transferring accu 1 to a word from the "system data" area
LIR [1]	0 to 15	Load the register (indirect): with the contents of the memory word [2] addressed by accu 1
TIR [1]	0 to 15	Transfer the register contents (indirect): into the memory word [2] addressed by the contents of accu 1
TNB [1]	0 to 255	Block transfer in bytes: source in accu 2, destination in accu 1 [3]
TNW [1]	0 to 255	Block transfer in words: source in accu 2, destination in accu 1 [4]

[1] system function

[2] LIR, TIR
- a) Registers 4, 7, 8, 13, 14 not present —> NOP
- b) Registers 5, 6, 15: TIR loads the register addressed into the accu 1 L word. LIR loads the accu 1 L word into the register addressed. No transfer from/to the memory!
- c) If the address area FF00H to FFFFh is addressed —> NOP.
- d) Access to the 8-bit memory:
  TIR: the high byte of the register is lost.
  LIR: FFH is written into the high byte of the register.

[3] The command TNB is limited to the following types of block transfer:
- a) byte memory address area B000 to EFFF —> byte memory address area B000 to EFFF
- b) byte memory address area B000 to EFFF —> I/O address area F000 to FDFF
- c) I/O address area F000 to FDFF —> byte memory address area B000 to EFFF.

[4] The command TNW is limited to the following types of block transfer:
- a) user memory —> user memory
- b) user memory —> byte memory address area B000 to EFFF
- c) byte memory address area B000 to EFFF —> user memory

[5] The system data 3 and 4 contain information about disturbances if any have occurred (see section 10). Only the system data 60 to 63 can be written into by the user by means of T RS and are not used by the system program.

- Arithmetic operations

Operation	Description
ENT	Enter data in the arithmetic memory The ENT command results in the loading of accus 3 and 4 which are also used in arithmetic operations: accu 4: = accu 3 accu 1: = accu 1 accu 3: = accu 2 accu 2: = accu 2

The former contents of accu 4 are lost.

Example

The following fraction is to be calculated: (30 + 3 x 4)/6 = 7

	Accu 1	Accu 2	Accu 3	Accu 4
Accumulator defaults prior to execution of arithmetic operations	a	b	c	d
L KF 30	30	a	c	d
L KF 3	3	30	c	d
ENT	3	30	30	c
L KF 4	4	3	30	c
* F	12	30	c	c
+ F	42	c	c	c
L KF 6	6	42	c	c
/ F	7	c	c	c

Operation	Parameters	Description
ADD BF	−127 to +127	Add byte constant (fixed point) to accu 1 [1]
ADD KF	−32 768 to +32 767	Add fixed point constant (word) to accu 1 [1]
TAK		Exchange the contents of accus 1 and 2 [2]

[1]) Accus 2, 3 and 4 are not changed
[2]) Accus 3 and 4 are not changed

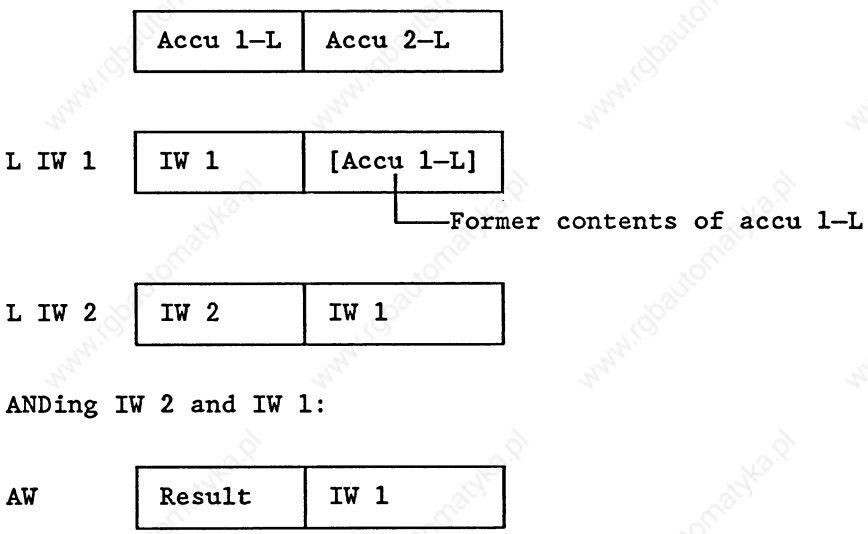
- Digital logic operations

Operation	Description
AW	Digital ANDing of accus 1 and 2
OW	Digital ORing of accus 1 and 2
XOW	Exclusive digital ORing of accus 1 and 2

Accumulators 3 and 4 are not affected, but the condition codes CNC1 and CNC0 are (see section 6.1).

By means of two loading operations, accumulators 1 and 2 can be loaded corresponding to the operands of the loading operation. Then, the contents of both accumulators can be operated on digitally.

Example



Organizational functions

- Jump functions

The destination of unconditional and conditional jumps is specified symbolically (a maximum of 4 characters beginning with a letter). The symbolic parameter of the jump instruction is identical to the symbolic address of the statement to be jumped to. When programming, it should be taken into account that the absolute jump distance does not cover more than ±127 words and that a STEP 5 statement can consist of more than one word. Jumps can only be carried out within a block; jumps across segments are not permissible.

Note: jump statement and jump destination must be in one segment. Per segment only one symbolic address is permitted for jump destinations. These conditions do not apply to the JR jump, for which an absolute jump distance is specified as a parameter.

Operation	Description
JU = addr	Jump unconditional. An unconditional jump is carried out under all conditions.
JC = addr	Jump conditional. A conditional jump will be carried out if RLO = 1. If RLO = 0, the statement will not be carried out and the result of the logic operation will be set to RLO = 1.
JZ = addr	Jump condition: CNC1, CNC0. A jump will only be carried out if CNC1 = 0 and CNC0 = 0. The logic operation result is not changed.
JN = addr	Jump condition: CNC1, CNC0. A jump will only be carried out if CNC1 ≠ CNC0. The logic operation result is not changed.
JP = addr	Jump condition: CNC1, CNC0. A jump will only be carried out if CNC1 = 1 and CNC0 = 0. The logic operation result is not changed.
JM = addr	A jump will only be carried out if CNC1 = 0 and CNC0 = 1. The logic operation result is not changed.
JO = addr	Jump on overflow. A jump will be carried out if the condition code OV = 1. If there is no overflow, (OV = 0) the jump will not be carried out. The logic operation result is not changed. An overflow occurs if the permissible area for the numerical representation involved is exceeded by an arithmetic operation.

addr = symbolic address (a maximum of 4 characters)

Operation	Description
JS = addr	Jump if the condition code OS (latching overflow) is set (OS = 1).
JR [1]) −32 768 to +32 767	Jump over the system software.

addr = symbolic address (a maximum of 4 characters)

o Shift functions

Operation	Description
SLW 0 to 15	Shifting to the left (zeros are filled in from the right).
SRW 0 to 15	Shifting to the right (zeros are filled in from the left).
SLD 0 to 32	Shifting a doubleword to the left (zeros are filled in from the right).
SSW 0 to 15	Shifting to the right with sign.
SSD 0 to 32	Shifting a doubleword to the right with sign (sign is filled in from the left).
RLD 0 to 32	Rotating to the left.
RRD 0 to 32	Rotating to the right.
Parameters	

With the shift functions only accu 1 is used. The parameter part of the commands specifies up to how many positions the accu contents are shifted or rotated. With SLW, SRW and SSW, only the less significant word is involved with the shift functions, with SLD, SSD, RLD and RRD the entire contents of accu 1 (32 bits) are used.

Shift functions are carried out unconditionally. The last bit shifted out can be interrogated by means of jump functions. The CNC0 and CNC1 condition codes are affected (see section 6.1).

With JZ, a jump can be carried out if the bit is 0. With JN, a jump can be carried out if the bit is 1.

[1]) System function

Examples

```
STEP 5 program:      Contents of the data words

:L    DW52           H = 14AF
:SLW  4
:T    DW53           H = 4AF0


STEP 5 program:      Accu 1 contents (hexadecimal):

:L    ED0            2348 ABCD
:SLW    4            2348 BCD0
:SRW    4            2348 0BCD
:SLD    4            3280 BCD0
:SSW    4            3480 FBCD
:SSD    4            0348 0FBC
:RLD    4            3480 FBC0
:RRD    4            0348 0FBC
:BE
```

- Conversion functions

Operation	Meaning
CFW	Forming of one's complement of accu 1 (16 bit)
CSW	Forming of two's complement of accu 1 (16 bit)
CSD	Forming of two's complement of accu 1 (32 bit)
CBW	Fixed point conversion (16 bit) from BCD to binary
BDW	Fixed point conversion (16 bit) from binary to BCD
DED	Doubleword conversion (32 bit) from BCD to binary
BDD	Doubleword conversion (32 bit) from binary to BCD
FDG	Conversion of a fixed point number (32 bit) to a floating point number (32 bit)
GFD	Conversion of a floating point number to a fixed point number (32 bit)

Examples

The contents of data word 64 are to be inverted bit by bit and stored in data word 78.

```
STEP 5 program:      data word assignment:

:L   DW64            BP = 0011111001011011
:CFW
T    DW78            BP = 1100000110100100
```

The contents of data word 207 are to be interpreted as a fixed point number and should be stored in data word 51 with the opposite sign.

```
STEP 5 program:      data word assignment

:L   DW207           F: + 51
:CSW
:T   DW51            F: - 51
```

- Decrementing/incrementing

Operation	Description
D 1 to 255	decrementing
I 1 to 255	incrementing
parameters	

The contents of accu 1 are decremented or incremented by the number specified as a parameter. The execution of the operation is unconditional. It is limited to the right byte (without carry).

Example

```
STEP 5 program:      data word assignment

:L   DW7             H = 1010
:I     16
:T   DW8             H = 1020
:D     33
:T   DW9             H = 10FF
```

- Processing functions

Operation	Description
DO DW 0 to 255 (operation)	Process data word The following specified operation will be combined with the parameter specified in the data word and executed.
DO FW 0 to 254 (operation	Process flag word The following specified operation will be combined with the parameter specified in the flag and executed.
DO = □ enter formal operand	Process formal operand (parameter class: B): Only Q DB, JU, PB, JU FB, JU SB can be substituted.
DOI [1] [2]	Process using a formal operand (indirect). The number of the formal operand to be executed is in accu 1.
DO RS [1], [2]	The command in the system data area (RS) is to be executed.

All operations, with the exception of those listed below, can be combined with DO DW or DO FW:

- C DB0, C DB1,
- all two and three word commands
- operations with formal operands
- JU OBxx and JC OBxx.

The PG does not check whether the combinations are permissible.

[1] System function
[2] The value, which is in the system data or in the formal operand in the function block, is interpreted as the operation code of a STEP 5 operation which will then be executed. Permissible operations are as with DO FW and DO DW. Only the system data 60 to 63 are reserved for the user and are not used by the system program.

Example ("process data word")

The contents of data words DW 20 to DW 100 are to be set to signal status "0". The index register for the parameter of the data words is DW 1.

```
        :L   KF 20      supply index register
        :T   DW1
F001    :L   KF 0       reset
        :DO  DW1
        :T   DW0
        :L   DW1        increment index register
        :L   KF 1
        :+F
        :T   DW1
        :L   KF 100
        :<=F
        :JC  =F001      jump if index is in the range
        ...             further STEP 5 program
```

- Disable/enable command output

Operation	Description
IA	Disable process interrupts
RA	Enable process interrupts

The "disable/enable interrupts" function can be used, for example, if interrupt-driven processing is to be suppressed during time-driven processing.

7 System program special functions

7.1 Transmit data block (OB 255)

The system program special function "transmit data block" is initiated by calling organization block OB 255.

OB 255 expects the number of the data block to be transmitted in the accumulator. The transmission can **only** be carried out from the 16 bit user memory into the 8 bit data block RAM (see section 6.1).

OB 255 checks independently whether the data block to be transmitted is present in the user memory and transmits it (with the appropriate correction of the address entry in DB 0) to an empty data block RAM memory area.

DB 0 is managed by the system program. It contains the start addresses of all blocks.

If the specified data block is not present in the user memory or already present in the data block RAM, the CPU stops and outputs the error message SFF (see section 5.7). The same applies if there is insufficient space for the transmission in the data block RAM.

A partial transmission of the data block does not occur in any of the possible error types.

Note! If a data block called with C DBX is to be transmitted, it must be created again following transmission with DBX. If it is not, then all further accesses to data words using loading and transfer commands will be related to the old data block. Only data blocks with a maximum of 256 data words (DW 0 to 255) may be transmitted.

7.2 PID controller (OB 250 and OB 251)

The user can call one or more PID controllers in the processor of the S5 135 U.

Each controller must be initialized in the initial start organization block. A data block is used for the transfer of parameters.

The actual control algorithm is integrated in the operating system and can only be called as an organization block by the user. A data block is once again used as data interface between control algorithm and the user program.

- **PID controller functions**

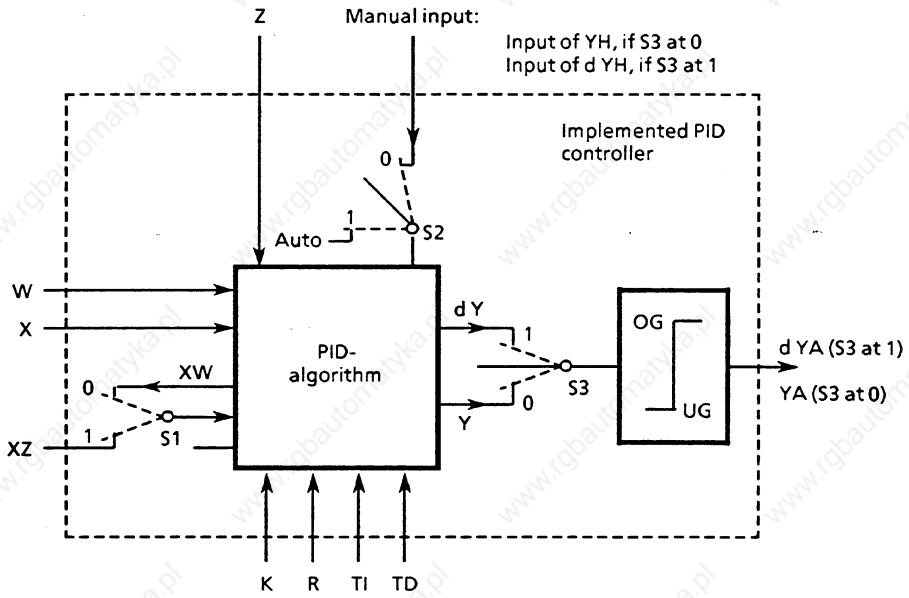


Fig. 18 Block diagram of the PID controller

Index k: kth scan

Switch	Position	Effect
S1 STEU bit 1	0 1	The control difference XW_k is supplied to the derivative unit. Via XZ, another signal can be supplied to the derivative unit
S2 STEU bit 0	0 1	Manual operation Automatic operation
S3 STEU bit 3	0 1	Position algorithm Velocity algorithm
S4 STEU bit 4	0 1	With feed forward control Without feed forward control

A function corresponding to the switch positions of this block diagram is achieved during parameter assignment of the PID controller, by setting the control bits appropriately in the control word STEU (see section 7.2.1).

The continuous controller is intended for high speed control systems e.g. in process engineering to control pressure, temperatures or flow rates.

The controller itself is based on a PID algorithm. Its output signal can either be output as a manipulated variable (position algorithm) or as a manipulated variable change (velocity algorithm).

The individual P, I and D components can be switched off by means of their respective parameters R, TI and TD (see section 7.2.8) by presetting the locations concerned with zero (see section 7.2.5). This means that all required controller structures e.g. PI, PID or PD controllers can easily be implemented.

The control deviation XW or (via the XZ output) any influencing quantity or the inverted actual value x can be supplied to the derivative unit.

To compensate for the influence of disturbances a feed forward control on the control element without time response is provided. A process-related disturbance signal Z is fed forward to the control algorithm. In manual operation, the preselected manipulated variable YN is accepted at this point (see page 98).

If an inverted control direction is required, a negative K value should be preset.

If the control information (dY or Y) is at a limit, the I component will be switched off automatically, in order to avoid deterioration of the controller response.

The controller program can be supplied with fixed values or adaptive (dynamic) parameters (K, R, TI, TD). They are entered via the memory locations assigned to the individual parameters.

7.2.1 PID algorithm

The PID controller is based on a velocity algorithm, according to which the corresponding control increment dY_k is calculated at a particular point in time t = k.TA according to the following formula:

$$dY_k = K[(XW_k - XW_{k-1})R + TA/TN \times XW_k + \frac{1}{2}(TV/TA(XU_k - 2XU_{k-1} + XU_{k-2}) + dD_{k-1})]$$

$$= K \quad \underbrace{[dPW_k}_{\text{P component}} \quad + \quad \underbrace{dI_k}_{\text{I component}} \quad + \quad \underbrace{dD_k]}_{\text{D component}}$$

$dXXX_k$: change in the quantity XXX at time t.

U can be W or Z, depending on whether XW or XZ is supplied to the derivative unit. The following applies:

With XW_k supply:	With XZ supply:
$PW_k = W_k - X_k$	
$PW_k = XW_k - XW_{k-1}$	$PZ_k = XZ_k - XZ_{k-1}$
$QW_k = PW_k - PW_{k-1}$	$QZ_k = PZ_k - PZ_{k-1}$
$QW_k = XW_k - 2XW_{k-1} + XW_{k-2}$	$QZ_k = XZ_k - 2XZ_{k-1} + XZ_{k-2}$

$$dPW_k = (XW_k - XW_{k-1})R$$

$$dI_k = TI.XW_k \qquad TI = \frac{TA}{TN}$$

$$dD_k = \frac{1}{2}(TD.QU_k + dD_{k-1}) \qquad TD = \frac{TV}{TA}$$

If the manipulated variable Y_k is required as controller output at time t_k, it is formed according to the following formula:

$$Y_k = \sum_{m=o}^{m=k} dY_m$$

With most process control designs it is assumed that R = 1, if a P response is required.

The quantity R can be used to set the proportional component of the PID controller.

7.2.2 Data blocks for the PID controller

Controller specific data are entered using a transfer data block (for initialization and call of PID controller see section 7.2.3).

The following data should be entered by the user into the transfer data block x:

K, R, TI, TD, W, STEU, YH, BGOG, BGUG

The structure of the transfer block is explained in more detail below.

• Structure of the transfer data block

Word no.	Name	I/O	Number format	PG format	Remarks
0	–	–	–	–	Reserve
1	K	I	GP	KG	Proportional coefficient K > 0: positive control direction i.e. change of setpoint and manipulated variable in same control direction K < 0: negative control direction; floating point number range
3	R	I	GP	KG	R parameter,usually = 1 for controllers with P component; floating point number range
5	TI	I	GP	KG	TI = TA/TN; floating point number range
7	TD	I	GP	KG	TD = TV/TA; floating point number range
9	W_k	I	GP	KG	Entry of setpoint here, if STEU bit 6=1, otherwise in word no. 19 ($-1 \leq W_k < 1$)
11	STEU	I	BM	KM	Control word (see page 77)
12	YH_k	I	GP	KG	Manual entry here if STEU bit 6 = 1; otherwise in word no. 18 ($-1 \leq YH_k < 1$). With velocity algorithms, manipulated variable increments must be specified here.
14	BGOG	I	GP	KG	Upper limit value $0 \leq BGOG < 1$ ($YA_{k\ max.}$)
16	BGUG	I	GP	KG	Lower limit value $-1 \leq BGUG \leq 0$ ($YA_{k\ min}$)
18	YH_k	I	LP	KF	Manual entry here if STEU bit 6 = 0 ($-1 \leq YH < 1$). With velocity algorithms, manipulated variable increments must be specified here.
19	W_k	I	LP	KF	Entry of setpoint here, if STEU bit 6=0 ($-1 \leq W_k < 1$)

Word no.	Name	I/O	Number format	PG format	Remarks
20	MERK				Reserve
21	X_k	I	LP	KF	Entry of actual value for STEU bit 7=0 ($-1 \leq X_k < 1$)
22	X_k	I	GP	KG	Entry of actual value for STEU bit 7=1 ($-1 \leq X_k < 1$)
24	Z_k	I	LP	KF	Influencing quantity ($-1 \leq Z_k < 1$)
25	Z_k	I	GP	KG	Influencing quantity input here, if STEU bit 7 = 1 ($-1 \leq Z_k < 1$)
27	Z_{k-1}		GP	KG	Previous value of influencing quantity
29	XZ_k	I	LP	KF	Value supplied to the derivative unit via input XZ ($-1 \leq XZ_k < 1$); entry here if STEU bit 7 = 0
30	XZ_k	I	GP	KG	XZ entry here, if STEU bit 7 = 1 ($-1 \leq XZ_k < 1$)
32	XZ_{k-1}		GP	KG	Previous limit of XZ_k
34	PZ_{k-1}		GP	KG	$XZ_{k-1} - XZ_{k-2}$
36	dD_{k-1}		GP	KG	Derivative component
38	XW_{k-1}		GP	KG	Previous value of control deviation
40	P_{wk-1}		GP	KG	$XW_{k-1} - XW_{k-2}$
42	–	–	–	–	Reserve
44	Y_{k-1}		GP	KG	Previous value of the calculated manipulated variable Y_{k-1} or dY_{k-1} before the limiter
46	YA_k	Q	GP	KG	Output quantity
48	YA_k	Q	LP	KF	Output quantity BGUG ≤ YA ≤ BGOG

Recommended format (KH, KM also possible)

GP = floating point LP = binary or decimal fractions

I = input, Q = output

- Assignment of the control word STEU (word 11 in the transfer data block)

Bit	Name	Meaning
0	AUTO	= 1: automatic operation = 0: manual operation
1	XZ_ON	= 1: a different quantity, which must not be XW_k, is supplied to the derivative unit via the XZ input. = 0: XW_k is supplied to the derivative unit. The XZ input is ignored.
2	REG_AUS	= 1: when calling the controller (OB 251) all values in the controller DB (DW 20 to DW 48) are erased once (except K, R, TI, TD, BGOG, BGUG, STEU, YH_k and W_k). The controller is switched off. The previous value of the influencing quantity is updated. = 0: controller on
3	VELOC	= 1: velocity algorithm = 0: position algorithm
4[1])	MAN	= 1: for GESCHW = 0 (position algorithm) the manipulated variable output last is retained. For GESCHW = 1 (velocity algorithm), positioning increment $dY_k = 0$ is set. = 0: for GESCHW = 0, after switching over to manual operation, the value of the manipulated variable YA output is brought exponentially to the manually set value in 4 sampling steps. After this, further manually set values are immediately acepted at the controller output. For GESCHW = 1, the manually set values are immediately switched to the controller output. With manual operation, the limits are effective, and the following quantities are updated: 1) X_k, XW_{k-1} and PW_{k-1} 2) XZ_k, XZ_{k-1} and PZ_{k-1}, if STEU bit 1 = 1 3) Z_k and Z_{k-1}, if STEU bit 5 = 0 The value dD_{k-1} is set to 0. The algorithm is not calculated.
5	NO_Z	= 1: no feed forward control = 0: with feed forward control
6	PGDG	= 1: W_{k-}, Y_{Hk-} input as a floating point number. = 0: input as a binary/decimal fraction.
7	VAR_GP	= 1: the variables X_k, XZ_k and Z_k are entered as floating point numbers. = 0: input of the variables as a binary/decimal fraction.
8..15		No significance

[1]) Only relevant for manual operation (AUTO = 0)

7.2.3 Initialization and call up of the PID controller in the STEP 5 program

- Initialization in the start up OB's 20/21/22

- Selection of the transfer DB x (contains parameters)
- Call of OB 250 (controller initialization routine)

For data transfer, each controller must use its own DB x ($x \leq 254$). The system program generates another DB x + 1, in the data block RAM of the CPU, which the controller uses as a data field during cyclic operation; therefore the appropriate DB numbers must be kept available. These DB x + 1's are the data interface between the controller and the user or peripherals.

Caution! If the DB x + 1 was not kept available during initialization, it will be used, without any indication from the operating system, as a controller data field, provided that it is the **same length** as a controller DB (48 data words) and data words 20 to 48 will be erased. Otherwise, the PC stops.

- Calling up the controller during the cycle

Calling up the controller after the scan time has elapsed

- Select data field DB x + 1
- Load input data X_k, XZ_k, Z_k and YH_k or a subset of them
- Input data are converted to the correct format and transferred into DB x + 1
- Call OB 251 (PID controller)
- Load output data YA_k from DB x + 1
- Conversion of the data and transfer to process I/O's

7.2.4 Format of controller inputs and outputs

The PID controller algorithm uses the floating point format internally to represent numbers and can be supplied with floating point values. It can also be supplied with binary or decimal fractions (see bit 6 and 7 in control word STEU). In this case, the controller automatically converts the words into floating point format whenever it is called.

In the STEP 5 program the conversion of words from the input and output modules requires less runtime if the binary or decimal fraction format is used.

- Inputs

W, YH, X, Z and XZ can be input as floating numbers or binary/ decimal fractions. The data transfer block has different memory locations for each quantity.

- Input as binary/decimal fractions

For details of these numbers see section 7.2.7.

Caution! In order to keep within the nominal input range of the analog input modules, it is important to remember that the bit pattern for a particular input value is different from that when the whole input range is used. This fact must be considered, particularly when setting a setpoint, otherwise it might be possible that a setpoint, input via the PG, is not reached, although the actual value is far higher than the desired value.

If the analog-digital converter used supplies negative numbers as values with a sign, then the two's complement must be formed from them before they are transferred into the controller DB. Binary position 15 must then be set to 1.

If, with the analog-digital converter, the number 0 can occur as a value with sign as follows:

1000000000000000

then the two's complement must not be formed from it. Instead, the number must be transferred to the controller DB as +0:

0000000000000000

- Output

The controller output YA is available in the DB as a left and fixed point number. The format of binary/decimal fraction inputs and outputs must be converted depending on the input and output modules used (analog-digital converter, digital-analog converter) before and after the controller call up in the STEP 5 user program, before they are transferred into or out of the controller DB.

7.2.5 General notes

The parameter TI = TA/TN should never be zero, otherwise when switching over from manual to automatic operation in the steady state ($XW_k = XW_{k-1} = XW_{k-2}$), any particular control deviation might not change the manipulated variable (formula for the calculation of dY_k: see section 7.2.1).

The controller data blocks DB x + 1 cannot be reloaded during cyclic operation.

If a cascaded control with two or more controllers is set up, the following must be observed:

- If the cascade is to be split, then either all controllers must go over to manual operation at the same time, so that no controller can drift owing to its I component, or at least the controller of the outer loop must be on manual, so that the last manipulated variable, which corresponds to the setpoint of the inner loop, is maintained or can be set to a safe value.

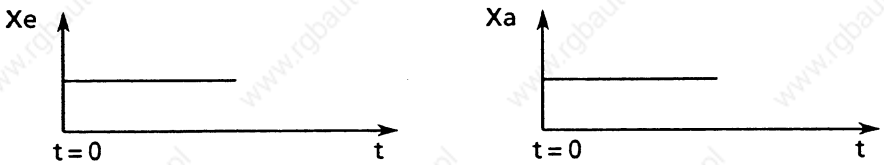
- If the cascade is to be closed, then both loops should operate simultaneously on automatic or at least the inner loop so that the manipulated variable of the outer loop can be taken as the setpoint.

If during the switchover to manual operation, the controlled system is separated from the controller and is adjusted directly at the final control element, the resulting manipulated variable must be supplied to the controller by manual input. This ensures that when switching over from manual to automatic operation, the controller output corresponds to the manipulated variable set in manual operation. With the velocity algorithm this is the manipulated variable deviation.

7.2.6 Controller characteristic quantities

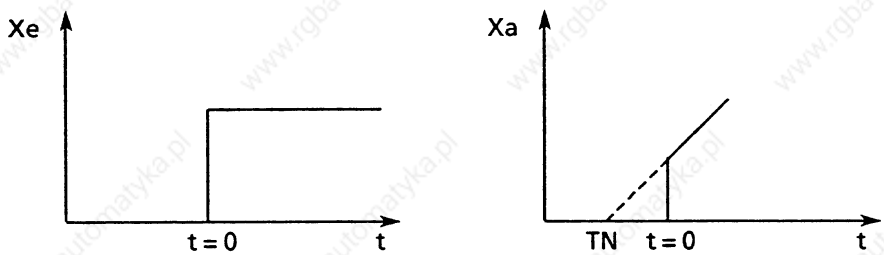
- P controller

The characteristic quantity for a P controller is K. This is the quotient of output and input quantity: $K = X_{out}/X_{in}$.



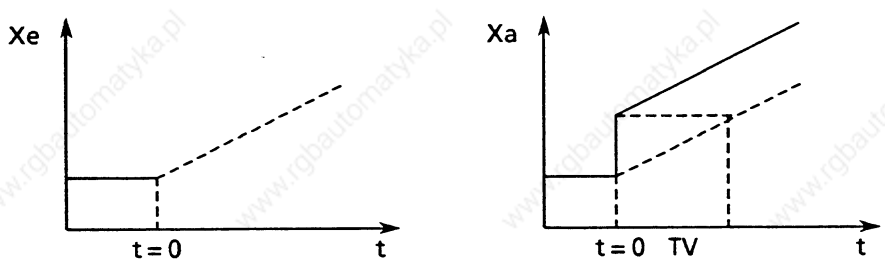
- PI controller

The proportional coefficient K and integral time constant (reset time) TN are the characteristic quantities for a PI controller. Proportional coefficient K is the quotient of the input and output quantities and determines the P response. The reset time TN is the time required to respond and achieve the same change in the manipulated variable by means of the I action, as is brought about by the P component.



- PD controller

Proportional coefficient K (see above) and derivative time constant TV are the characteristic quantities for a PD controller. The feed forward time is the time which a P controller would need given a constant rate of change in order to bring about the same change in the output quantity, as a PD controller immediately brings about as a result of its D component. To determine the feed forward time a linear change of the input value is used instead of a step function.



- PID controller

The characteristic quantities of a PID controller are the proportional coefficient K, the reset time TN and the feed forward time TV. These determine the P, I and D responses.

7.2.7 Binary/decimal fractions

To represent a binary/decimal fraction in the data block, one word is necessary. The relationship between a decimal number, a binary number and the representation in KF format on the programmer is shown in the following example

Fraction		Fixed point number
Decimal	Binary	
−0.999...	1000000000000001	−32767
−0.75	1010000000000000	−24576
−0.5	1100000000000000	−16384
−0.25	1110000000000000	− 8192
0	0000000000000000	0
+0.25	0010000000000000	+ 8192
+0.5	0100000000000000	+16384
+0.75	0110000000000000	+24576
+0.999...	0111111111111111	+32767

In binary representation negative fractions are derived from positive fractions by forming the two's complement.

Binary/decimal fractions (LP) can be converted into the values displayed (KF) on the programmer with the following formula:

$LP \times 32767 = KF$

with $-1 < LP < +1$ and $-32767 \leq KF \leq +32767$

7.2.8 Abbreviations

dY_k	calculated positioning increment
dZ_k	influencing increment
GP	floating point representation
k	kth scan
K	proportional coefficient
LP	left point representation
UL	upper limit (limiter)
R	R parameter
TA	scan time
TD	TV/TA
TI	TA/TN
t	scan point (time) = k.TA
TN	reset time
TV	feed forward time
LL	lower limit (limiter)
W_k	setpoint
X_k	actual value
XW_k	control derivation
Y_k	calculated manipulated variable
YA_k	value of manipulated variable (positioning increment or manipulated variable)
Z_k	influencing quantity

7.3 Shift register (OB 240, 241, 242)

7.3.1 Mode of operation

The following diagram illustrates the principle underlying the software shift register. It consists of rows of 8-bit wide storage locations in the data block RAM of the S5 135 U.

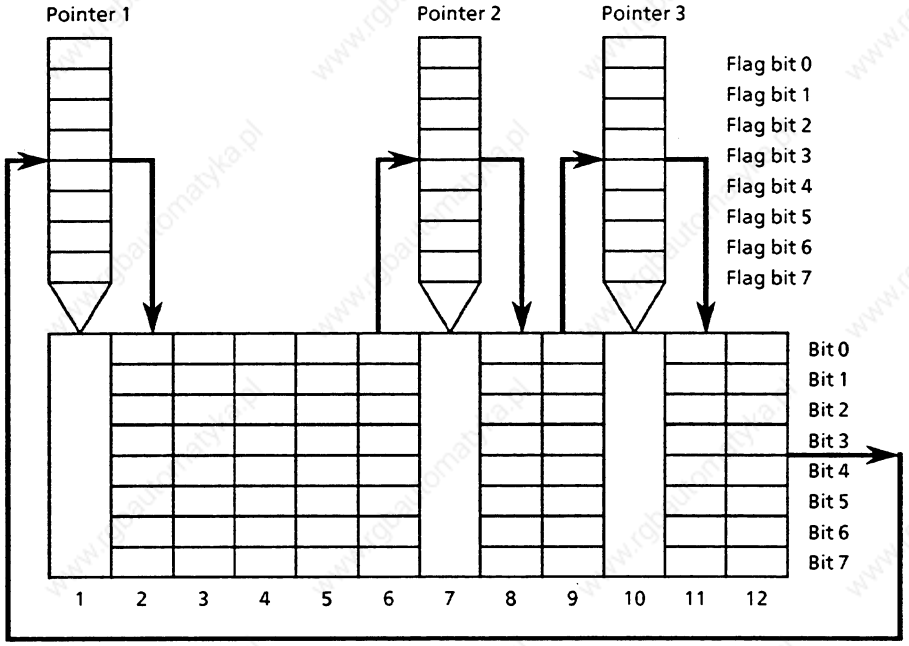


Fig. 19 Schematic diagram of the shift register with 3 pointers and 12 storage locations

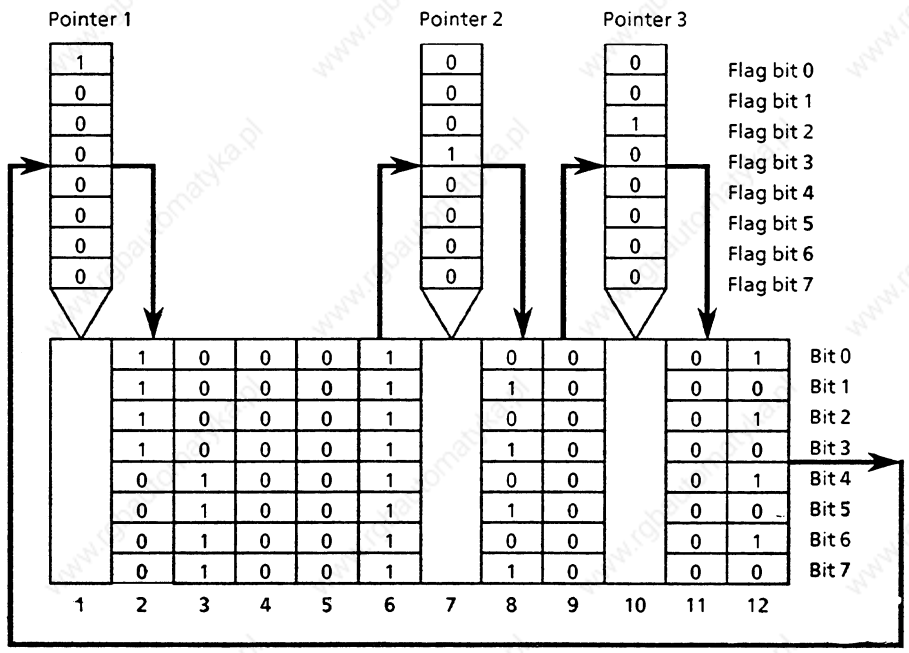


Fig. 20 Schematic diagram of the shift register with 3 pointers and 12 storage locations before the first clock pulse

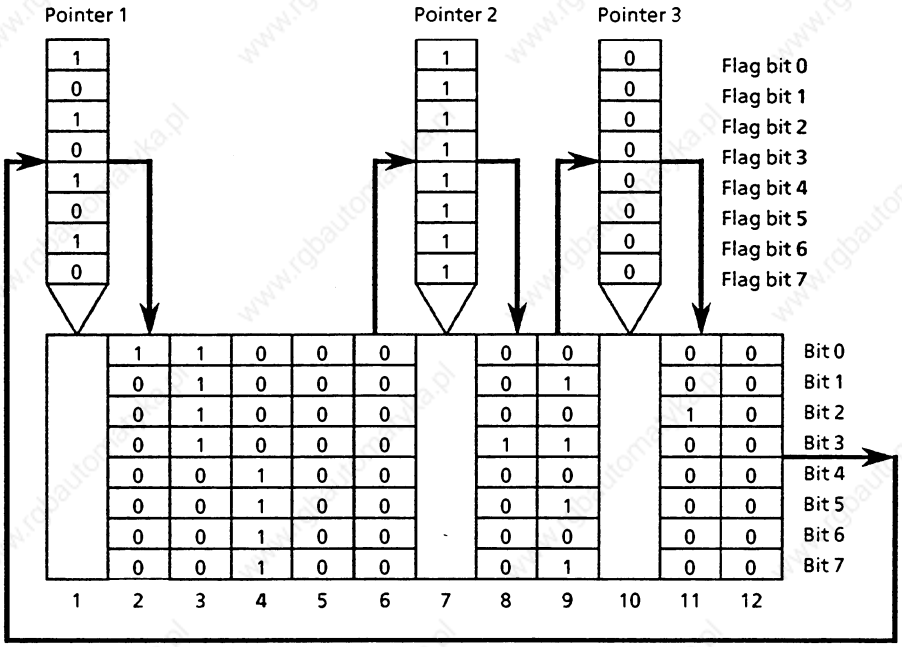


Fig. 21 Schematic diagram of the shift register with 3 pointers and 12 storage locations after the first clock pulse

The number of storage locations between L = 2 and L = 256 can be selected by the user. Individual shift register locations are stored outside the data block RAM in flag bytes, which will be designated as pointers from now on.

The setting of the first pointer (base pointer) is fixed at the first storage cell. All other pointers can be positioned by the user relative to the base pointer. The number of pointers between 1 (base pointer) and 6 can be selected by the user. As in the case of a hardware shift register, the information is shifted through in bytes from storage location 1 as shown by the arrows in the diagrams to storage location L. From there, the information returns to storage location 1. Each shift register function call causes the whole information to be shifted by 1 storage location ≙ 1 clock pulse.

The user can enter information into the shift register, or interpret information from it. This is only possible using the pointers i.e. the flag bytes functioning as pointers.

- Before the SR function call: setting/resetting the flag bits (Fig. 20).

		STEP 5
Example:	Set flag bit 0 of pointer 1	S F 0.0
	Set flag bit 3 of pointer 2	S F 1.3
	Set flag bit 2 of pointer 3	S F 2.2

- Call of the shift register function JU OB 241

  Effect: the information (8 bits) of the storage locations is shifted by 1 location.

	STEP 5
- Interpretation of the information which is now in the pointers	L FB 0

Example: scanning of flag bits 0, 3 and 2 (Fig. 21) at the base pointer.

In the base pointer, therefore, all the information which comes from the entries to all pointers can be interpreted (in the above example, this is only possible after 12 clock pulses).

7.3.2 Programming the shift register in the user program (OB 241 to 248)

Eight shift registers can be called in the S5 135 U. The calls are implemented in the user program using OB 241 to 248. Before a shift register can be called in the cyclic user program, it must have had parameters assigned to it via a data block. The allocation of shift register numbers, organization block numbers and data block numbers is shown in the following table:

shift register number 1	OB 241	DB 241
shift register number 2	OB 242	DB 242
shift register number 3	OB 243	DB 243
shift register number 4	OB 244	DB 244
shift register number 5	OB 245	DB 245
shift register number 6	OB 246	DB 246
shift register number 7	OB 247	DB 247
shift register number 8	OB 248	DB 248

Each shift register is assigned six flag bytes as pointers:

Flag byte 0 ≙ pointer 1 Flag byte 1 ≙ pointer 2 Flag byte 2 ≙ pointer 3 Flag byte 3 ≙ pointer 4 Flag byte 4 ≙ pointer 5 Flag byte 5 ≙ pointer 6	shift register 1
Flag byte 6 ≙ pointer 1 Flag byte 7 ≙ pointer 2 Flag byte 8 ≙ pointer 3 Flag byte 9 ≙ pointer 4 Flag byte 10 ≙ pointer 5 Flag byte 11 ≙ pointer 6	shift register 2
Flag byte 12 ≙ pointer 1 Flag byte 13 ≙ pointer 2 Flag byte 14 ≙ pointer 3 Flag byte 15 ≙ pointer 4 Flag byte 16 ≙ pointer 5 Flag byte 17 ≙ pointer 6	shift register 3
Flag byte 18 = . . .	
Flag byte 42 ≙ pointer 1 Flag byte 43 ≙ pointer 2 Flag byte 44 ≙ pointer 3 Flag byte 45 ≙ pointer 4 Flag byte 46 ≙ pointer 5 Flag byte 47 ≙ pointer 6	shift register 8

Fig. 22 Fixed allocation of flag bytes to shift registers 1 to 8

- Assigning parameters to the shift registers in the start-up organization block (OB 20/21/22)

The following information must be stored in the data block which is allocated to the shift register in question (example for 6 pointers):

Word 0: 0
Word 1: length L
Word 2: distance n_2
Word 3: distance n_3
Word 4: distance n_4
Word 5: distance n_5
Word 6: distance n_6
Word 7: 0

L is the number of shift register memory locations ($2 \leq L \leq 247$).

n_2 = distance of 2nd pointer from base pointer
n_3 = distance of 3rd pointer from base pointer etc.

The first and last words of the parameter assignment data block must be equal to 0. If there is only one pointer, the base pointer, then only the number of shift register memory locations need be specified.

Word 0: 0
Word 1: L
Word 2: 0

For parameter assignment during the start-up of the S5 135 U (OB 20 to 22) in each shift register used,

- the accompanying data block and
- then organization block OB 240 must be called.
  128 words in the DB RAM are reserved for every shift register initialized. With this, the end address of the useable DB RAM is shifted down to lower addresses.

- Calling shift registers in the cyclic user program

Enter data in the individual flag bytes (pointers).

Call the organization block which is allocated to the shift register (OB 241 to 248).

Intepret the data present in the individual flag bytes.

As each shift register call corresponds to one clock pulse the program sequence might have to be preceded by an edge evaluation or similar.

7.3.3 Enabling the shift register memory areas (OB 221)

Organization block OB 240 reserves user memory locations in the DB RAM of the CPU for each initialized shift register (128 words per shift register). The user memory is enabled by calling organization block OB 221, preferably during the start-up.

7.4 Triggering the scan time (OB 222)

By calling up organization block OB 222, the use can prolong the scan time monitoring which is currently running by 100 ms, measured from the time of the OB call: i.e., the "inner clock" which is also running is restarted by the user program. Before the user program is called up by OB 1 or FB 0 (see Fig. 11), the system program triggers the scan time in the cyclic section.

7.5 Expanding the sign from a 16 to 32-bit fixed point number (OB 220)

By calling up organization block OB 220, the sign of a 16-bit fixed point number in accu 1 is expanded to 32 bits. This is, e.g., necessary prior to fixed point floating point conversion (FDG, 32 bits) of a negative 16-bit fixed point number.

7.6 Comparing start-up modes in multiprocessor operation (OB 223)

If organization block OB 223 is called up in multiprocessor operation, the corresponding CPU checks whether the start-up modes of **all** the CPU's involved are the same. If they are not, the CPU in question stops and outputs the error message SFF (see section 5.7).

7.7 Reading the cross-check sum of the system program EPROM (OB 227)

If organization block OB 227 is called up, the cross-check sum of the system program EPROM is transferred into accu 1-L.

If OB 226 is called up, the contents of a memory address from this EPROM, which must have been specified previously in accu 1-L, are stored in bytes in accu 1.

During the cyclic program execution, the EPROM consistency can be checked by means of the fixed point addition of all byte contents (addresses 0 to 7FFF).

7.8 Block transfer of the IPC flags (OB 224)

Normally, the IPC flags specified in DB 1 by the user are transmitted **in bytes** by the system program to the CPU (see section 1.4.1 and Fig. 11). In multiprocessor operation this transfer takes place with each CPU acting independently. This is intended to keep the time that the bus is blocked by a CPU to a minimum, the bus assignment being controlled by COR. Therefore, only bytes can be used as coherent units of information.

By calling OB 224 during the initial start with each start up mode used and with each CPU involved the user can transmit all the IPC flags specified in DB 1 in blocks. Each CPU can only start its IPC transfer when the transfer by another CPU has been completed. The cyclic program execution will be appropriately delayed (cycle time!). By means of this special function, IPC interleaved updating will be separated out for the individual CPU's in order to clarify the assignment of all the IPC flag information. The function is only effective in the COR operating mode "RUN" with a bus enable time of < 8 µs (see COR operating instructions).

7.9 Assigning parameters to the start-up characteristics (OB 225)

If the special function OB 225 is called up, the statuses of the timer and counter locations are retained in the manual and automatic cold restart with memory recall. OB 225 must be called up **before** OB 21 or 22 has finished being processed. It then becomes effective at the next cold restart with memory recall. It should be noted that following each cold restart without memory recall the standard setting "erase timer and counter locations" must be active until OB 225 is called.

8 STEP 5 special commands

From software release SO A03 of the PG 675, the following STEP 5 special functions are available:

generate data block	EDBxxx
set semaphore	SES
enable semaphore	SEF

8.1 Generate data block

The command E DBxxx generates a data block with the number xxx (between 2 and 255) in the data block RAM of the CPU. The number of data words in accu 1-L is expected (maximum length of data block including block header: 4096 words). If the corresponding data block already exists or there is not sufficient space in the DB RAM the CPU stops with the error message SFF.

8.2 Set/enable semaphore

The SESxxx (set semaphore) or SEFxxx (enable semaphore) commands control the data exchange between CPU's or CP's in multiprocessor operation. By setting an SESxxx semaphore, the data area (IPC flags) designated with the number xxx (0 to 31), which must be determined in the user program, will be disabled for other CPU's. With SEFxxx this data area can be read out of or written into again by other CPU's. A semaphore can only be enabled by the CPU which set it. The commands SES/SEF affect the condition codes (see section 6.1) as follows:

CNC1	CNC0	Meaning
0	0	Semaphore has been set by another CPU and cannot be set/enabled
1	0	Semaphore is set/enabled

The indications can be evaluated by the jump functions (see section 6.3).

9 Overview of STEP 5 operations

Basic functions

Operation	Parameters
• Binary logic operations:	
A I	0.0 to 127.7
A Q	0.0 to 127.7
A F	0.0 to 255.7
A D [1]	0.0 to 255.15
A T	0.0 to 127
A C	0.0 to 127
AN I	0.0 to 127.7
AN Q	0.0 to 127.7
AN F	0.0 to 255.7
AN D [1]	0.0 to 255.15
AN T	0.0 to 127
AN C	0.0 to 127
O I	0.0 to 127.7
O Q	0.0 to 127.7
O F	0.0 to 255.7
O D [1]	0.0 to 255.15
O T	0.0 to 127
O C	0.0 to 127
O I	0.0 to 127.7
ON Q	0.0 to 127.7
ON F	0.0 to 255.7
ON D [1]	0.0 to 255.15
ON T	0.0 to 127
ON C	0.0 to 127
)	
A(	
O(	
O	
• Compare functions:	
!=F	
><F	
>F	
>=F	
<F	
<=F	
!=D	
><D	
>D	
>=D	
<D	
<=D	
!=G	
><G	
>G	
>=G	
<G	
<=G	

[1]) Word 1: B2 + bit address; B3 + relative address

Operation	Parameters
• Memory operations:	
S I	0.0 to 127.7
S Q	0.0 to 127.7
S F	0.0 to 255.7
S D [1]	0.0 to 255.5
R I	0.0 to 127.7
R Q	0.0 to 127.7
R F	0.0 to 255.7
R D [1]	0.0 to 255.15
= I	0.0 to 127.7
= Q	0.0 to 127.7
= F	0.0 to 255.7
= D [1]	0.0 to 255.15
• Loading functions:	
L IB	0 to 127
L IW	0 to 126
L ID	0 to 124
L QB	0 to 127
L QW	0 to 126
L QD	0 to 124
L FB	0 to 255
L FW	0 to 254
L FD	0 to 252
L DL	0 to 255
L DR	0 to 255
L DW	0 to 255
L DD	0 to 254
L T	0 to 127
L C	0 to 127
L PB	0 to 127 128 to 255
L PW	0 to 126 128 to 254
L OB	0 to 255
L OW	0 to 254
LD T	0 to 127
LD C	0 to 127
L KB	0 to 255
L KC	2 alphanumeric characters
L KM	bit pattern (16 bit)
L KH	0 to FFFF
L KF	−32 768 to +32 767
L KY	0 to 255 for each byte
L KT	0.0 to 999.3
L KC	0 to 999
L KG	[2])

[2]) $\pm 0.1469368 \times 10^{-38}$ to $\pm 0.1701412 \times 10^{39}$

Operation	Parameter
• Timer and counter operations:	
SP T	0 to 127
SE T	0 to 127
SR T	0 to 127
SS T	0 to 127
SF T	0 to 127
R T	0 to 127
S C	0 to 127
R C	0 to 127
CU C	0 to 127
CD C	0.0 to 127
• Transfer functions:	
T IB	0 to 127
T IW	0 to 126
T ID	0 to 124
T QB	0 to 127
T QW	0 to 126
T QD	0 to 124
T FB	0 to 255
T FW	0 to 254
T FD	0 to 252
T DR	0 to 255
T DL	0 to 255
T DW	0 to 255
T DD	0 to 254
T PB	0 to 127
	128 to 255
T PW	0 to 126
	128 to 254
T OB	0 to 255
T OW	0 to 254
• Block calls:	
JU PB	0 to 255
JU FB	0 to 255
JU SB	0 to 255
JC PB	0 to 255
JC FB	0 to 255
JC SB	0 to 255
C DB	0 to 255
BE	
BEC	
BEU	
JU OB	1 to 255
JU OB [1]	40 to 255
JC OB [1]	40 to 255

[1] special function call
[2] system operation

Operation	Parameters
• Arithmetic operations:	
+F	
−F	
xF	
:F	
+G	
−G	
xG	
:G	
• Other functions:	
NOP 0	
NOP 1	
STP	
BLD	

Supplementary functions

• Logic functions, in words:	
AW	
OW	
XOW	
• Timer and counter functions:	
FR T	0 to 127
FR C	0 to 127
FR =	formal operand
SP =	formal operand
SR =	formal operand
SEC =	formal operand
SSU =	formal operand
SED =	formal operand
RD =	formal operand
Loading and transfer functions:	
L =	formal operand
LC =	formal operand
LW =	formal operand
LD =	formal operand
T =	formal operand
L RS	0 to 255
L RI	0 to 255
T RI [2]	0 to 255
T RS [2]	0 to 255
LIR [2]	0 to 15

Operation	Parameters
• Loading and transfer functions (continued):	
TIR [2]	0 to 15
TNB [2]	0 to 255
TNW [2]	0 to 255
• Binary logic operations	
A =	formal operand
AW =	formal operand
O =	formal operand
ON =	formal operand
• Conversion functions:	
CFW	
CSW	
CSD	
CBW	
BDW	
DED	
BDD	
FDG	
GFD	
• Shift functions:	
SLW	0 to 15
SRW	0 to 15
SLD	0 to 32
SVD	0 to 32
RLD	0 to 32
RRD	0 to 32
SVW	0 to 15
• Jump functions:	
JU =	Symbolic address
JC =	Symbolic address
JZ =	Symbolic address
JN =	Symbolic address
JP =	Symbolic address
JM =	Symbolic address
JO =	Symbolic address
JS [2]	Symbolic address
JR [1]	−32 768 to +32 767

Operation	Parameters
Setting operations:	
S =	formal operand
RB =	formal operand
= =	formal operand
• Other functions	
RA	
IA	
ENT	
D	0 to 255
I	0 to 255
DO =	formal operand
DO DW	0 to 255
DO FW	0 to 255
BI [1]	
DO RS [1]	0 to 255
TAK	
BLD	0 to 255
• Arithmetic operations:	
ADD BF	−127 to +127
ADD KF	−32 768 to +32 767
• Special functions:	
E DB	0 to 255
SES 1)	0 to 31
SEF 1)	0 to 31

[1]) System operation
[2]) Word 1: jump distance (2 bytes

10 Information about disturbances

Information about disturbances which may have occurred is stored in the system data SD3 and SD4.

Error identifier SD3	SD4	Error text	System reaction
0101	abs	Incorrect block identifier or block length in the user module	Stop setting up DB0
0102	abs	Incorrect block identifier or block length in the DB RAM	Stop setting up DB0
0103	abs	Transferred DB 1 incorrect	Stop setting up DB1
0104	–	EPROM empty or not connected	System program overall reset request. No start–up possible without an overall reset.
0105	–	Contents of user RAM not correct (incorrect identifier).	System program overall reset request. No start–up possible without an overall reset.
0106	–	CPU RAM does not contain correct identifier	System program overall reset request. No start–up possible without an overall reset.
0201	–	Address list missing (DB 1) in multiproc. operation	Remain in stop status
0202	–	The selected start–up mode is illegal	Stop
0301	ptr	Address list error: digital inputs are present, the I/=S of which are not plugged in (in single and multiprocessor operation)	Jump back into the stop loop
0302	ptr	Address list error: digital outputs are present, the I/O's of which are not plugged in (in multi–processor operation)	Jump back into the stop loop
0303	ptr	Address list error: digital outputs are present, the I/O's of which are not plugged in (in single pro–cessor operation).	Jump back into the stop loop

abs = absolute address of the incorrect input
ptr = pointer on process image address list
- = entry irrelevant

Shift register initialization

Error identifier SD3	SD4	Error text	System reaction
0701	sac	No space in the DB RAM	Stop
0702	sac	Shift register too large or pointer position outside the shift register	Stop

Shift register call

Error identifier SD3	SD4	Error text	System reaction
0703	87xx	Shift register has not been initialized xx = shift register number	Stop

PID controller initialization

Error identifier SD3	SD4	Error text	System reaction
0704	sac	No space in the DB RAM	Stop
0705	sac	DBx no. = 255	Stop
0706	sac	DB length incorrect	Stop
0707	sac	DB+1 is already present in the user module	Stop

sac = step address counter

PID controller call

Error identifier SD3	SD4	Error text	System reaction
0708	87xx	DBx+1 is already present is the user module, (cycle) additional identifier for PID controller error, xx = DB no.	Stop

Error identifier SD3	SD4	Error text		System reaction
2101	–	Power failure	NAU	Stop
2102	–	I/O not operable	PEU	Stop
2103	–	Battery failure	BAU	Stop
2104	–	Halt signal	HALT	Stop
2105	–	Stop signal	STOPS	Stop
2106	x	Acknowledgment delay x = during updating of process image (see ISTACK): pointer on process image address list element x = with single I/O access: sac	QVZ	Stop
2107	sac	Addressing error	ADF	Stop
2108	sac	Cycle error	ZYKF	Stop
3110	sac	Illegal DB 0 call		Stop
3111	sac	Data block called does not exist		Stop
3112	sac	There is an illegal multi-word command in the BS RAM		Stop
3117	sac	Stop request from the user		Stop

sac = step address counter
- = entry irrelevant

Error identifier SD3	SD4	Error text	System reaction
3118	sac	This STEP 5 command does not exist	Stop
3119	sac	This STEP 5 command does not exist	Stop
3120	sac	This STEP 5 command does not exist	Stop
3121	sac	This STEP 5 command does not exist	Stop

Special function OB 255

Error identifier SD3	SD4	Error text	System reaction
3403	sac	DB not loaded	Stop
3404	sac	Insufficient memory space	Stop
3405	sac	DB already in the DB RAM	Stop

Error identifier SD3	SD4	Error text	System reaction
35xx	sac	xx: number of timer location called is greater than the number of timer locations specified in DB 1 or > 127	Stop
3610	sac	Stack overflow with a block call	Stop
3611	sac	Block called is not loaded	Stop
3616	sac	Time interrupt error	Stop
3622	sac	The special function OB called is not present	Stop

sac = step address counter

Error identifier SD3	SD4	Error text	System reaction
3701	sac	Command is an illegal multi-word command BMW/BDW)	Stop
3702	sac	Block parameter with the command L=/T= is not permissible	Stop
3711	sac	Formal parameter with the command BI= equals zero	Stop
3712	sac	Formal parameter with the command BI= is greater than 126	Stop
3801	sac	Block parameter in the command (A=, AN=, O=, ON=, ==, S=, RB=) is not permissible	Stop
3810	sac	Erase DB 0	Stop
3812	sac	Insufficient memory space with command EDB	Stop
3813	sac	DB already present with command EDB	Stop

sac = step address counter

Appendix A:

Aspects of the handling of STEP 5 commands in which the S5 135 U differs from the S5 150 S:

Loading functions

Not all byte and word loading functions erase the high word in accu 1. Also, only the low word of accu 1 is saved in the low word of accu 2. The most significant 16 bits in accus 1 and 2 are therefore not changed by these commands.

Timer and counter operations

The parameter area for all timer and counter operations is limited to 0 to 127, i.e. a maximum of 128 timers and counters can be programmed. Naturally, this limitation also applies to binary logic operations with timers and counters.

Arithmetic operations

With the floating point operations + G, - G, x G and : G, only a 16-bit mantissa is used, in contrast to the 24-bit mantissa with the S5 150 S. The missing least significant 8 bits are set to zero.

Appendix B

Notes on programming the S processor in the S5 135 U

The S5 135 U can be programmed and test functions can be carried out on the PG 670 (software release A08) and the PG 675 (software release PG 675U S0 A01) with the presetting "S5 150 S". The following limitations apply here:

- STEP 5 command set and parameter limits (see section 9).
- During the execution of byte and word loading commands, the high word in accu 1 is not erased, and only the low word of accu 1 is transmitted to accu 2.
- Program dependent signal status displays and program checks can only be implemented if there is a RAM. VAR status is also possible with EPROM operation.
- Data blocks DB 0 and 1 must **not** be used in the user program neither in single nor multiprocessor operation (reserved for system program purposes).

* OB calls with numbers > 39 are not possible with program inputs/outputs.

* Memory configuration in the PC: the values for the 16-bit user memory and the 8-bit DB RAM are output alternately.

* System parameters in the PC: the allocation of information to the screen form is incorrect.

* Memory address in the PC: if addresses > 0B000H are output, the address allocation at the PG is incorrect.

* Program dependent signal status display/direct signal status display/ program check: the output for timers, counters and data words supplies irrelevant data.

* Control process image (control VAR): this call is illegal.

* Read out interrupt stack: the allocation of control bits to the screen and of identifiers in the indicator byte is incorrect.

* Switch over operating mode (PC start/stop): this call is illegal.

- Test blocks: test blocks cannot be input.

The above limitations marked with an asterisk do not apply if the PG 675 is used with software release PG 675U S0 A02 or PG 675U S1 A01.

Siemens Aktiengesellschaft

Ord. No.: C79000-B8576-C264-03
Order from: Gerätewerk Karlsruhe
Printed in the Federal Republic of Germany
AG 0786 0.4 SK 122 En

SIEMENS

SIMATIC S5
S5135 U Programmable Controller
R Processor

Programming Guide Order No. C79000-B8576-C364-02

Contents

		Page
1	**Explanatory Notes**	4
1.1	Application	4
1.2	Single Processor Operation	4
1.3	Multiprocessor Operation	4
2	**Programming**	5
2.1	Programming Language and Program Structure	5
2.1.1	STEP 5 Programming Language	
2.1.2	Program Structure	5
2.2	Program Organization, Program Storage and Program Execution	7
2.2.1	Program Organization	7
2.2.2	Program Storage	8
2.2.3	Program Execution	9
2.3	Programming in Multiprocessor Operation	10
2.3.1	Flags for Inter-processor Communication (IPC)	10
2.3.2	Assignment of I/O's	11
2.4	Runtime Optimization and Limitations	12
2.5	Program Blocks	14
2.5.1	Programming Program Blocks	14
2.5.2	Calling Program Blocks	14
2.6	Data Blocks	15
2.6.1	Programming Data Blocks	15
2.6.2	Calling Data Blocks	16
2.6.3	Special Data Blocks	17
2.7	Function Blocks	24
2.7.1	The Structure of Function Blocks	24
2.7.2	Calling Function Blocks and Parameter Assignment	25
2.7.3	Programming Function Blocks	27
2.8	Organization Blocks	32
3	**Operation**	35
3.1	Overview of the Operating Statuses	35
3.2	Initialization	36
3.3	Stop Status	37
3.3.1	The "STOP" LED Flashes Quickly (Warning)	37
3.3.2	The "STOP" LED Is Lit Continuously	38
3.3.3	The "STOP" LED Flashes Slowly	38
3.4	Test Operation	39
3.5	Start-up	40
3.5.1	Cold Restart	40
3.5.2	Manual Warm Restart	41
3.5.3	Automatic Warm Restart	41
3.5.4	Start-up in Multiprocessor Operation	41
3.6	Program Execution	42
3.6.1	Cyclic Program Execution	42
3.6.2	Interrupt-driven Program Execution	47
3.6.3	Time-driven Program Processing	48
3.7	Interrupt Handling	49
3.7.1	Interruption at STEP 5 Command Boundaries	49
3.7.2	Interruption with Other Causes of Trouble	50
3.7.3	Control Bits and Interrupt Stack (Output with the PG)	54
4	**STEP 5 Command Set with Programming Examples**	60
4.1	General Rules	60
4.2	Basic Operation Set	66
4.3	Supplementary Operation Set	88
5	**System Program - Special Functions**	100
5.1	Fixed Point Expansion Form 16 Bits to 32 Bits (OB 220)	100
5.2	Resetting the Cycle Time (OB 221)	101
5.3	Retriggering the Cycle Time (OB 222)	101
5.4	Reading a Storage Location of the System Program EPROM (OB 226)	101
5.5	Reading the Cross-checksum of the System Program EPROM (OB 227)	101
5.6	Shift Register (OB 240, 241, 242)	102
5.6.1	Mode of Operation	102
5.6.2	Initialization and Call of the Shift Register	104
5.6.3	Erasing the Shift Register (OB 242)	105
5.7	PID Controller (OB 250 and OB 251)	106
5.7.1	PID Algorithm	108
5.7.2	Data Blocks for the PID Controller	109
5.7.3	Initialization and Call up of the PID Controller in STEP 5 Program	113
5.7.4	Format of Controller Inputs and Outputs	113
5.7.5	General Notes	114
5.7.6	Controller Characteristic Quantities	115
5.7.7	Binary/Decimal Fractions	116
5.8	Data Block Copying Function (OB 254, OB 255)	117
5.9	Block Transfer of the IPC FLags	118
5.10	Comparing the Start-up Mode	118
5.11	Access to Pages (OB 216 to 218)	119
5.11.1	Writing Data on a Page (OB 216)	119
5.11.2	Reading Data from a Page (OB 217)	121
5.11.3	Occupying a Page (OB 218)	121
5.12	System Program Auxiliary Functions (OB 230 to 237)	122
6	**Overview of STEP 5 Operations**	123
7	**Error Information**	126
7.1	Error Information in System Data 3 and 4	126
7.2	Error Information about Accu 1 and Accu 2	129

Abbreviations

ACCU 1(2)-L(H)	Accumulator 1(2) low value (high value)
BARB	Program check
BARBEND	Finish program check
BCD	Binary coded decimal
COR	Coordination module (coordinator)
CP	Communications processor
CPU	Central processing unit
CSF	Control system flowchart
DB	Data block
DX	Extended data block
FB	Function block
FX	Extended function block
IP	Intelligent I/O modules
ISTACK	Interrupt stack
LAD	Ladder diagram
OB	Organization block
PC	Programmable controller
PI	Process image
PII	Process image inputs
PIO	Process image outputs
PG	Programmer
RLO	Result of logic operation
SB	Sequence block
STL	Statement list

Further reading

The following handbooks contain an introduction to programming with STEP 5 and using standard function blocks:

Programming logic controls with STEP 5
Volume 1, Programming basic functions
Siemens AG, ISBN 3-8009-1407-7
Volume 2, Using standard function blocks
Siemens AG, ISBN 3-8009-1373-9
Volume 3, Programming function blocks yourself
Siemens AG, ISBN 3-8009-1366-6

1 Explanatory notes

1.1 Application

The S5 135 U programmable controller (PC) with a programmable memory is a high performance multiprocessor device for process automation (open loop control, signalling, monitoring, closed-loop control, logging). It can be used both to create the simplest logic controls with binary signals and to solve complex automation tasks. Its user programs are created with the programming language STEP 5.

The central controller of the S5 135 U can be equipped by the user with:

- one central processing unit (CPU) for single processor operation or
- one coordinator (COR)
  and up to 4 CPU's for multiprocessor operation:
- and also communications processors (CP's): up to 8 CP's for single processor operation or from 4 to 7 CP's for multiprocessor operation.

In the central controller of the S5 135 U, the remaining unoccupied module locations are available for input and output modules. In order to extend the peripherals, expansion units can be connected to the central controller.

The module location assignment and the possible combinations of central controllers and expansion units can be found in catalog ST 54.

1.2 Single processor operation

There are two types of CPU specially designed for automation tasks where the emphasis is on open-loop or closed-loop control:

- S processor for open-loop control or bit processing,
- R processor for closed-loop control or word processing.

With single processor operation for simple automation tasks, either an R or an S processor must be used depending on the requirement. Data exchange between the I/O modules and CP's takes place via the S5 bus.

1.3 Multiprocessor operation

For complex automation tasks, the central controller can be expanded to a multiprocessing device by using several S processors and/or R processors at the same time.

In a multiprocessor PC each individual CPU processes the user program which has been assigned to it independent of the other CPU's.

If more than one CPU is used, a COR must be included. The COR manages the data traffic on the S5 bus. In addition, for every CPU, the user must program data block 1 (DB 1) (see section 2.3).

For data exchange between the individual CPU's, flags for interprocessor communication (IPC) are available on the COR. The CPU is in multiprocessor operation as soon as a COR is used in the PC, even if there is only one CPU present. Therefore, even in this case DB 1 must be programmed.

2 Programming

2.1 Programming language and program structure

2.1.1 STEP 5 programming language

The use of the STEP 5 programming language makes it possible to program functions ranging from simple binary logic to complex digital processing and basic arithmetic operations.

The program can be written using any of three methods of representation: control system flowchart (CSF), ladder diagram (LAD) and statement list (STL). This means the programming method can be adapted to the particular application. The machine code generated by the programmers (PG's) is identical for all three methods of representation. If certain programming rules are followed, the PG can translate the user program from one method of representation to another.

2.1.2 Program structure

The complete program of a PC consists of the system program and the user program. The system program contains all statements and declarations for internal functions (e.g. saving data in the event of a power failure, prompting operator reactions in particular situations etc.). This program is an integral part of the PC (EPROM) and cannot be changed by the user.

The user program consists of all statements and declarations programmed by the user for signal processing, through which the system (process) to be controlled (open or closed-loop) will be influenced according to the automation task.

The S5 135 U enables the user to carry out structured programming, i.e., the complete program is divided into individual self-contained program sections (blocks). This method has the following advantages for the user:

- simple and clear programming, even of large programs,
- program sections can be standardized,
- simple program organization,
- easy program modification,
- simple program testing,
- simple commissioning.

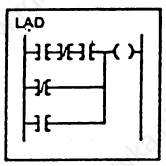
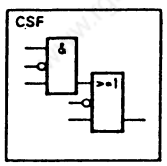
Ladder diagram	Statement list	Control system flowchart
Programming with graphic symbols as in circuit diagram to DIN 19239 (draft)	Programming with mnemonics of the function designation to DIN 19239 (draft)	Programming with graphic symbols to IEC 117-15 DIN 40700 DIN 40719 DIN 19239 (draft)
		

Fig. 1 Methods of representation in the STEP 5 programming language

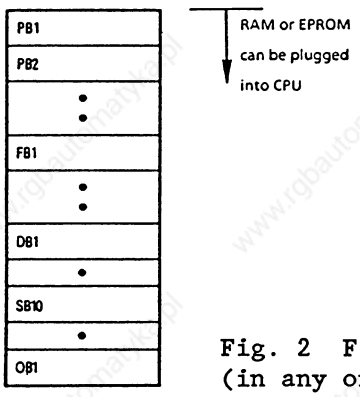


Fig. 2 Filing the blocks in the program memory (in any order)

Several types of software blocks, each with a different task, can be used to construct the user program:

- Organization blocks (OB)

These provide the interface between the system program and user program. There are special organization blocks which can be programmed by the user. These are intended for specific situations and are used to prompt a reaction from the user. There are also organization blocks, which the user cannot program, but can call and which contain system program special functions (see section 2.8).

- Program blocks (PB)

These are used to structure the user program in hardware oriented program sections (see section 2.5).

- Function blocks (FB) and extended function blocks (FX)

These are used to program functions which are frequently repeated or complex functions (e.g. unit control, signalling functions, arithmetic and closed-loop control functions). Exception: FB 0 (see section 2.7).

- Sequence blocks (SB)

These are special types of program blocks for processing sequence cascades.

- Data blocks (DB) and extended data blocks (DX)

These are used to store data and texts. The functions of these blocks are fundamentally different from those of the other blocks.

DB 0, 1, 2 and DX 0 are reserved for special purposes (see section 2.6).

A maximum of 256 program, function and sequence blocks, 253 data blocks, 255 extended data blocks and 39 organization blocks can be programmed. One block may occupy a maximum of 4096 words in the CPU program memory. In the case of inputs/transmission of blocks with the PG, the memory size of the PG used must be taken into account.

All blocks which have been programmed can be stored in any order by the PG in the program memory (Fig. 2), which is implemented on the CPU as a plug-in RAM or EPROM.

2.2 Program organization, program storage and program execution

2.2.1 Program organization

The program organization determines whether and in which sequence the blocks generated by the user will be processed (Fig. 3). Therefore corresponding calls (conditional or unconditional) for the blocks selected are programmed in organization blocks.

Additional program, function and sequence blocks can be called up in any desired combination by organization, program, function and sequence blocks.

The maximum permissible nesting depth is 24 blocks. This value can be seen as the total block nesting depth resulting from all the possible modes of operation (cyclic, interrupt-driven, time-driven and possibly also interrupt handling; see sections 3.6 and 3.7).

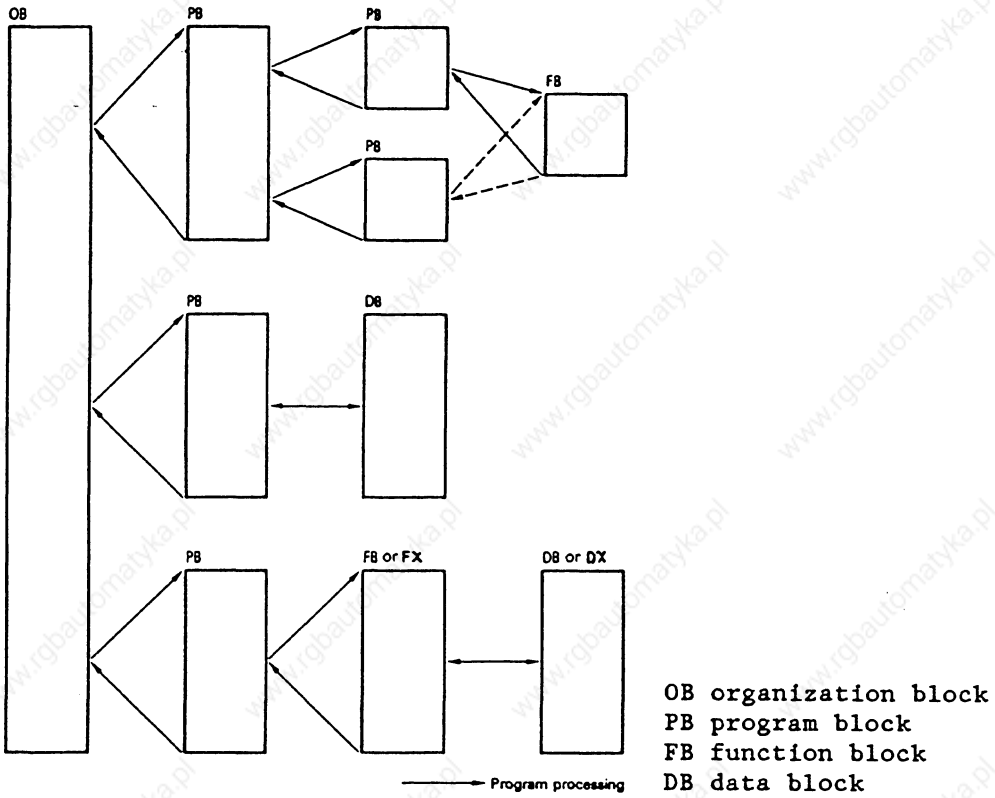


Fig. 3 Program organization in the STEP 5 programming language

2.2.2 Program storage

If a plug-in RAM is available in the CPU, the user program can be transferred directly from the PG to the CPU. Whereby, all programmed blocks are stored in the RAM in any order. The DB and DX data blocks will be stored in the RAM until it is full, after this they will be stored in the data block RAM of the CPU (see Fig 17). The CPU RAM has enough space for 11392 memory words. If shift registers are used, this space is, however, reduced by the number of data words required per shift register addressed; with the end address of the data block RAM shifting to lower addresses (see section 5.6).

If an EPROM is used to store the user program, all programmed blocks will be stored in it. Data blocks which contain variable data - i.e. which are to be changed during the user program - must therefore be copied from the EPROM to the RAM memory area of the CPU during the cold restart (see section 5.8). DB 0, 1, 2 and DX 0 are managed by the system program and may only be used in certain special cases (see section 2.6).

2.2.3 Program execution

The user program can be executed in three different ways (Fig. 4):

- Cyclic program execution (see section 3.6.1)

In order to execute the user program cyclically, either the organization block 1 (OB 1) or the function block 0 (FB 0) can be used:

- OB 1 runs cyclically, calling the blocks programmed in the user program.

- FB 0 is executed like OB 1; in addition, however, it allows supplementary STEP 5 operations to be used. It is therefore especially suitable for the processing of small time-critical programs, which do not need structured programming nor the block calls it involves.

  If OB 1 **and** FB 0 are programmed, only OB 1 will be run.

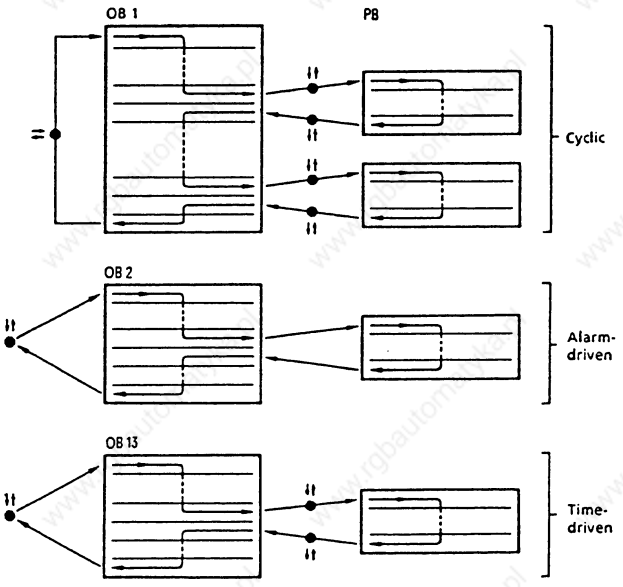
- Interrupt-driven program execution (see section 3.6.2)

With this type of program execution, the interruption of the cyclic program execution is initiated peripherally when there is a change of block. OB 2 is intended for calling interrupt routines.

- Time-driven program execution (see section 3.6.3).

With this type of program execution, certain program sections (called by OB 13) are automatically inserted into the cyclic program execution using a time base.

Time-driven program execution is necessary for the solution of closed-loop control tasks.



Interrupt point, at which interrupt-driven or time-driven program execution can normally be inserted into cyclic, interrupt-driven or time-driven program execution. Time-driven execution can only be interrupted by interrupt-driven execution and not vice versa (see section 3.6).

Fig. 4 Types of program execution

2.3 Programming in multiprocessor operation

The programming of individual S5 135 U CPU's for multiprocessor operation corresponds to the programming for single processor operation described in section 2.2. In addition, the following aspects of multiprocessor operation should be noted:

- The individual CPU's can exchange data with each other via the flags for inter-processor communication (IPC flags).
- The whole S5 135 U program can be distributed on the individual CPU's making each CPU responsible for certain aspects of the program.
- The peripheral inputs and outputs must be allocated to the individual CPU's. DB1 must therefore be programmed for each CPU.
- The assignment of the S5 bus to the individual CPU's is carried out by the COR. The number of CPU's used must be set on the COR (see COR operating instructions).

Program distribution:

The CPU's (up to max. four) which process their particular user program in the multiprocessor PC simultaneously and independently of one another, allow the user to divide the whole S5 135 U program into individual, distinct programs. As a result, multiprocessor operation offers the following advantages:

- Dividing the program among the CPU's, which then operate parallel to each other, increases the processing speed of the whole program.
- Programs with short runtimes for handling processes which depend on fast responses can be put together on their own CPU's. The user program runtime in such a CPU can be further reduced if the user brings in FB0 instead of OB1 and makes use of the opportunity to specify a timer block length (see section 2.4).
- User programs with long runtimes for handling processes which are not time-critical can be programmed on their own "slow" CPU separate from the "fast" CPU's.
- Each CPU can be assigned to a particular part of the plant depending on its function. By using M, R, and S processors in a PC, closed-loop and open-loop control tasks can be solved optimally.

2.3.1 Flags for inter-processor communication (IPC)

IPC flags are flag bytes which are designated by the user on the CPU as output or input. They are used for byte-serial, cyclic exchange of data between the CPU's and reside physically on the COR.

A flag byte defined as an output on a CPU will be transferred, in the cylic operation of the S5 135 U, via the COR, to the CPU's, which have an input flag byte designated with this number.

The following rules result from this flag byte function:

- The flag byte designated on one or more CPU's as an IPC input flag must be defined on another CPU as an IPC output flag.

- If a flag byte is designated as an IPC output flag on one CPU, it cannot be defined on another CPU as an IPC output flag. It can, however, be defined on three more CPU's as an IPC input flag.

- The flag bytes designated as IPC flags on a CPU are only available on this CPU for the exchange of data. All other flags which have not been designated can be used for their normal application.

- Only those flag bytes can be specified which are set on the COR or on the CP(s) (see COR or CP operating instructions).

- Using IPC flags is not absolutely necessary.

IPC flags can also be used for data transmission between CPU's and CP's. This function is possible both in single and multiprocessor operation.

If IPC flags are to be used simultaneously on the COR and CP, the whole area available for IPC flags can be divided among a maximum of 256 IPC flag inputs/outputs into sub-areas of 32 bytes (see COR or CP operating instructions). If IPC flags on the COR or CP do not acknowledge then an interrupt procedure takes place (see section 3.7). The input of address lists for IPC flags in DB1 is described in section 2.6.

2.3.2 Assignment of I/O's

In multiprocessor operation, the user must assign the peripheral input and output modules (only P peripherals with relative byte addresses from 0 to 127) in bytes to the individual CPU's, as with the IPC flags. The DB1 data block is provided for this purpose, and on the PG the user enters in it the distribution of the peripherals and of the IPC flags in the form of address lists (see section 2.6).

The process image is updated during cyclic operation only for the digital peripheral input and output bytes specified in these two address lists (c.f. section 3.6.1).

In single processor operation, the user can also program the address list (DB1) for inputs and outputs, in order to optimize the program runtime. When programming the DB1 both in single processor and multiprocessor operation, the following rules must be observed:

- As soon as an address list is accepted by the CPU with a cold restart, access to the I/O modules via the process image is only permissible for the byte addresses specified in DB1 (L/T I, IB, ID, Q, QB, QD) commands and the results of logic operations with inputs and outputs).

- Direct access to I/O modules with loading and transfer commands (L/T, PB, PW, OB, OW) while by-passing the process image, is possible for all the inputs which acknowledge on the corresponding modules during **loading**, irrespective of the address list.

- However, direct **transfer** is only possible for the outputs specified in DB1, if an address list is present, since with direct transfer the process image will also be written into. A process image is only available for the P peripherals with relative byte addresses from 0 to 127.

During the cold restart, the system program accepts the DB1, input by the user in the form of internal address lists, and checks whether the inputs and outputs or IPC flags specified in DB1 acknowledge on the corresponding modules. If they do not, the CPU goes into the stop status with the DB1 error message (see section 3.7) and the "STOP" LED will flash slowly. The execution of the user program will not be started.

2.4 Runtime optimization and limitations

Runtime optimization of the user program

- Program structure

In single processor operation - as well as in multiprocessor operation - the runtime of the user program can be minimised, if the user only makes use of structured programming when necessary.

As every block change requires additional runtime, structured programming can be avoided for short, time-critical programs, and only FB0 should be programmed. In FB0, the whole STEP 5 instruction set (see section 4) present in the S5 135 U is available.

- I/O assignment

With I/O assignment by means of DB1 (in single processor and multiprocessor operation), it is important that, in the address lists for I/O's and IPC flags, only those addresses are specified which the user program of the CPU in question accesses.

I/O addresses and addresses of IPC flags, which are not necessary for the particular user program, but which were cyclically updated due to the entry in DB1, extend the runtime of the whole program.

- Timer block length

In DB1 the user can specify the number of timer locations used as the timer block length. As a result the execution time spent on updating all the timer locations in the system program outside this timer block length will be saved, thus decreasing the cycle time (see 3.6.1). However, this is only possible if the numbers of the timer locations used by the user are smaller than the timer block length.

If 0 is specified as the timer block length, no timer locations will be processed. If no timer block length is specified, then all timer locations (numbers 0 to 127) are permissible.

The timer block length can be input in single processor and multiprocessor operation. The user must, however, program the complete DB1 address lists, i.e. he must also specify the address lists of the I/O's.

Section 2.6 describes the entry of timer block lengths in DB1. If a timer location is processed which is outside the timer block length, or the number of which is greater than 127, then the CPU recognises a parameter error (see section 3.7).

Limitations

Standard function blocks occupy the numbers 1 to 199. User function blocks can therefore only be generated with the numbers 200 to 255, if standard function blocks are used.

FB0 can only be used to program the cyclic program execution (see section 3.6.1).

If standard function blocks are used, flag bytes 200 to 255 are occupied and are no longer available to the user.

The DB 0, 1, 2 and DX 0 data blocks are reserved for special purposes:

- DB 0 is generated by the system program and contains the address list of all blocks in the loaded user program.

- DB 1 is input by the user as an address list, and is managed by the system program.

- DB 2 is used for assigning parameters to the compact closed-loop control, which can be obtained as a software product. It is written in assembler language and optimized in terms of runtime, it is further supported by the system program of the R processor (see operating instructions for compact closed-loop control in the S5 135 U U/R processor, order no. 6ES5 842-0BB10).

- The DX 0 extended data block is available to the user for presetting particular system program functions (see section 2.6).

For organization, program and sequence blocks, only the STEP 5 basic operation set is valid. Commands from the extended operation set can only be programmed in the FB and FX function blocks.

2.5 Program blocks

2.5.1 Programming program blocks

The following description applies to the programming of organization blocks, program blocks and sequence blocks. These three types of blocks do not differ as far as programming is concerned. They can be programmed in all three methods of representation STL, LAD and CSF of the STEP 5 programming language. Programming is started by entering a block number:

- program blocks 0 to 255
- sequence blocks 0 to 255
- organization blocks 1 to 39 (see section 2.8)

This is followed by the actual logic control program which is completed with the statement BE. Only the STEP 5 basic operation set can be used. The STEP 5 block program must not occupy more than 2000 words in the program memory. The block header, which the PG automatically generates for a block, occupies another 5 words in the program memory.

A block should always contain a complete program. Logic operations which go beyond the block limits are meaningless.

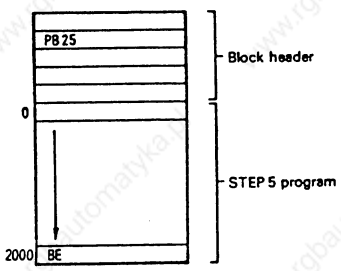


Fig. 5 Structure of an organization, program and sequence block

2.5.2 Calling program blocks

Block calls enable the blocks for processing (Fig. 6). These block calls can be programmed within an organization, program, function or sequence block. They are comparable with jumps to a subprogram and can be implemented both conditionally and unconditionally.

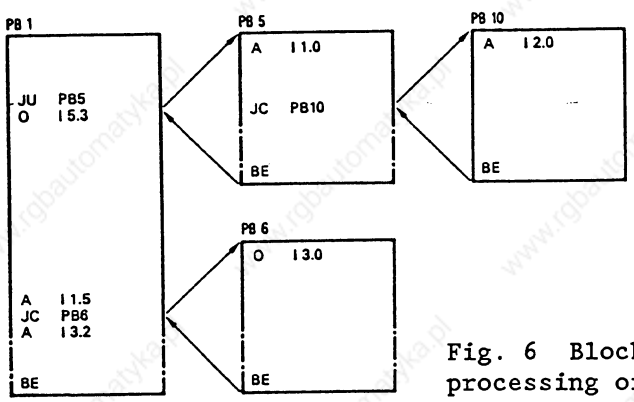


Fig. 6 Blocks calls which enable the processing of a program block

Following the BE statement, a jump is made back to the block, in which the block call has been programmed, and the STEP 5 command which follows the block call will be executed. Both following a block call and following BE, no further logic operations can be carried out on the result of the logic operation (RLO, see section 4.1), since both of these are RLO limiting commands (see section 4.1). The RLO is, however, taken into the new block and can be evaluated.

- Unconditional call: JU xx

The program block addressed is processed independent of the result of the previous logic operation.

- Conditional call: JC xx

The program block addressed is processed dependent on the result of the previous logic operation.

When RLO = 1, the jump statement is executed, when RLO = 0, it is not. In both cases, RLO is set to 1 by the jump statement. This dependence on the RLO and its influence also applies to the conditional block end statement BEC.

2.6 Data blocks

2.6.1 Programming data blocks

Data required within the user program are stored in data blocks. No STEP 5 operations are carried out in data blocks. Data may consist of:

- any desired bit pattern, e.g. for plant status,
- numbers (hexadecimal, fixed point, floating point) for times or results of calculations,
- alphanumeric characters e.g. for message texts.

The generation of a data block is started by specifying a data block number (maximum 255, e.g. DB 25). Each data block is made up of data words 16 bits wide which must be entered in ascending order starting with the data word 0 (see Fig. 7).

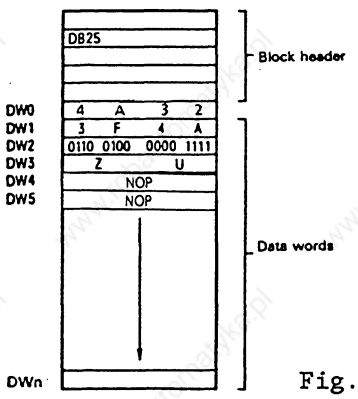


Fig. 7 Structure of a data block

One memory word is reserved per data word in the program memory. A block header which occupies five more words in the program memory, is generated by the programmer for each data block. A data block may occupy a maximum of 4096 words in the CPU program memory. When entering/transferring using the PG, the memory size of the PG should be taken into account.

Caution! With the L/T DW... load/transfer commands, access is only possible up to data word number 255. The DB0, 1 and 2 data blocks are reserved for special functions and cannot be put to other purposes by the user. The DX extended data blocks are generated and used just as DB's. The DX 0 is also reserved (see section 2.6.3).

2.6.2 Calling data blocks

Data blocks can only be called unconditionally. The call remains valid until a new data block is selected. A DB data block can be called within an organization, program, function or sequence block by the command C DB...., an extended data block by the command CX DX.... . The DB 0, 1, 2 and DX 0 data blocks are managed by the system program and must not be called by the user (see section 2.6.3).

Caution! Before a data word is loaded/transferred, a data block must have been selected. The addressed data word must be contained in the selected data block. Otherwise the system program recognises an error during the transfer (TRAE, see section 3.7).

Example

The contents of data word 1 should be transferred from the DB 10 data block to data word 1 of the DB 20 data block (Fig. 8).

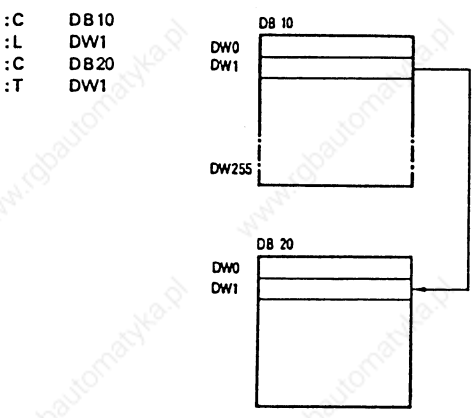
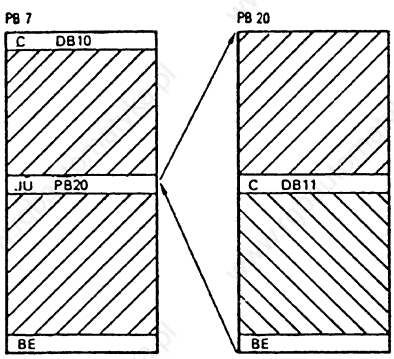


Fig. 8 Addressing a data block



/// area of validity of DB 10
\\\ area of validity of DB 11

Fig. 9 Area of validity of a selected data block

If a further program block is called by a program block in which a data block has already been addressed, and another data block is addressed in this second program block, then this second data block is only valid in the program block which has been called. Following the jump back to the first program block, the old data block is valid again (see Fig. 9).

Example (see Fig. 9)

In the PB 7 program block, the DB 10 data block is selected. In the subsequent operation the data of this data block will be processed.

Following the call, the PB 20 program block is processed. The DB 10 data block, however, remains valid. Only when the DB 11 data block is called will the data area be changed. The DB 11 data block is now valid until the end of the PB 20 program block.

After the block change back to the PB 7 program block, the DB 10 data block will be valid again.

2.6.3 Special data blocks

The DB 0, 1, 2 and DX 0 data blocks are reserved for particular functions. They are managed by the system program and cannot be put to other uses by the user:

- The DB 0 data block contains an address list with the start addresses of all the blocks which reside in the user memory submodule or in the data block RAM of the CPU. This address list is generated by the system program during initialization (see section 3.2). and is automatically updated when blocks are input or modified by means of the PG.

- The DB 1 data block contains a list of the digital inputs and outputs (P peripherals with relative byte addresses from 0 to 127) and of the inputs and outputs of IPC flags which are assigned to the CPU. DB 1 may also contain a timer block length (see section 2.3). DB 1 must be created by the user. During the cold restart, the system program generates internal address lists from it. If the inputs and outputs or IPC flags output in DB 1 are not present on corresponding modules, then the CPU goes into the stop status signalling a DB 1 error message (see section 3.7).

- The DB 2 data block is used for the assignment of parameters to the compact closed-loop control by the user. The compact closed-loop control can be obtained as a software product and is supported by the system program.

- The DX 0 extended data block allows the user to preset particular system program functions.

- Data block DB 1

Entering/modifying DB 1

- Online using the PG with the CPU in the stop status, if the CPU is equipped with a user RAM. The entered or modified DB 1 will only be accepted in a "cold restart" (see section 3.5.1).

- By programming the user EPROM.

Creation of DB 1 with the PG

The data words 0, 1 and 2 have the defaults

KH = 4D41, 534B, 3031.

From data word 3 onwards the individual address lists are specified.

Each address list begins with a keyword. Possible keywords are:

keyword for digital inputs	KH = DI00
keyword for digital outputs	KH = DQ00
keyword for IPC input flags	KH = CI00
keyword for IPC output flags	KH = CQ00

Following the keyword the relative byte addresses of the assigned I/O's or IPC flags which belong to this address list are listed as data words in fixed point format. The order of the entries within an address list is as arbitrary as the order of the address lists themselves.

The timer block length can be specified by

keyword for timer block length	KH = BB00

After this, the timer block length is specified using the fixed point format.

Following the last entry in DB 1,

KH = EE00

must be entered as a keyword.

In multiprocessor operation, DB 1 must be generated for **each** CPU. In single processor operation, it can be programmed to optimize the runtime. When a DB 1 is programmed, the inputs and outputs must be specified, as only these are updated cyclically. IPC flags and the timer block length entry are not absolutely necessary.

Example of DB 1

```
 0 :     KH = 4D41;
 1 :     KH = 534B;
 2 :     KH = 3031;
 3 :     KH = DI00;        keyword for digital inputs
 4 :     KF = +00000;      input byte 0
 5 :     KF = +00001;      input byte 1
 6 :     KF = +00002;      input byte 2
 7 :     KF = +00003;      input byte 3
 8 :     KF = +00007;
 9 :     KF = +00010;
10 :     KH = DQ00;        keyword for digital outputs
11 :     KF = +00000;      output byte 0
12 :     KF = +00002;      output byte 2
13 :     KF = +00004;
14 :     KF = +00012;
15 :     KH = CI00;        keyword for IPC input flags
16 :     KF = +00050;      flag byte 50
17 :     KF = +00051;
18 :     KF = +00060;
19 :     KH = CQ00;        keyword for IPC output flags
20 :     KF = +00070;      flag byte 70
21 :     KF = +00072;
22 :     KF = +00100;
23 :     KH = BB00;        keyword for timer block length
24 :     KF = +00040;
25 :     KH = EE00;        end identifier
```

From the S0A03 or S1A01 software release of the PG 675 onwards DB1 can also be input supported by screen forms (using the softkeys F1 and F3):

PERIPHERAL ASSIGNMENT

DIGITAL INPUTS	0,	1,	2,	3,	7,	10,					
DIGITAL OUTPUTS	0,	2,	4,	12,							
IPC INPUT FLAGS	50,	51,	60,								
IPC OUTPUT FLAGS	70,	72,	100,								
TIMER BLOCK LENGTH	40,										

- Extended data block DX 0

With DX 0, the user can match certain system program capabilities to his requirements, by specifying other defaults instead of the standard defaults (DF). It should be noted that the standard defaults (DF) are only set in the cold restart and that inputting or changing the DX 0 is only effective in a start-up mode.

The structure of DX 0

The data in DX 0 are divided into information blocks of various lengths.

Each block starts with a block identifier, with a specification in words about the block information length and with the block information which follows. Each block is allocated to a particular part of the system program or to a particular operating function. With the aid of the length specification it is possible to recognise the scope of the block information in each case. The block information consists of 1 to n data words, the significance of which is to be specified block by block. After the last block, there is the end identifier EEEE.

Formal structure:

Bit No. 15 8 7 0		
DW 0	'M' 'A'	ASCII 4D41
1	'S' 'K'	chars. 534B
2	'X' 'O'	5830
3	Block ident. 1, block info. length 1	
.	Parameter	Block info.
.	Parameter	
.	Parameter	
.	Block ident. 2, block info. length 2	Block info.
.	Parameter	
.	Block ident. 3, block info. length 3	
.	Parameter	
.	Parameter	Block info.
.	Parameter	
m	End identifier = EEEE	

The system can determine which information blocks are evaluated at a particular time (initialization, stop, initial start).

The blocks set out below are specified for the options which the user of the loop processor can select. Here, the following conventions apply:

- Blocks and parameters which are not required do not need to be specified.

- One block can appear several times in DX 0; in each case, the last parameter assignments are valid.

- The sequence of the block parameters does not need to be kept to. If particular parameters are named more than once the last specification is valid.

- During a cold restart, the parameters have default values so that a particuar function or reaction is produced by the system program. In all three start-up modes, DX 0 is evaluated and the parameters are set accordingly.

The numerical values specified correspond to the hexadecimal format. The system functions which can be influenced are described in section 3.

Block identifier	Parameter	Significance [2]
Start-up processing (see section 3.5):		
02xx [1]	1000	DF Automatic warm restart after "power on"
	1001	Automatic cold start after "power on"
	2000	DF Synchronization of the start-up in multiprocessor operation
	2001	No synchronization of the start-up in multiprocessor operation (time synchronization, see section 3.5.4)
	3000	DF Monitoring addressing errors
	3001	No monitoring of addressing errors
	BB00 00yy	Generating the number of timer locations to be updated (see timer block length in DB 1); Default value: 128 timer locations Permissible: 0 to 128 timer locations
Cyclic processing (see section 3.6.1):		
04xx [1]	1000 yyyy	Length of cycle time in milliseconds Default value: yyyy = 150 ms Permissible: 1 ms to 4000 ms
	4000	DF Updating the process image of the IPC flag without semaphore protection
	4001	Updating the process image of the flag for inter-processor communication with semaphore protection (see section 5.9).

[1]) xx = length of block information (words)
[2]) DF = default during cold restart

Block identifier	Parameter	Significance [2]
Interrupt processing (see section 3.6):		
06xx [1]	100C	DF Processing the time interrupt, loop controller interrupt and HW interrupt at the block limit
	1006	Processing the time interrupt, loop controller interrupt and HW interrupt at the command limit
	1008	Processing the loop controller interrupt and HW interrupt at the command limit and the time interrupt at the block limit
	100A	Processing the HW interrupt at the command limit, the loop controller interrupt and the time interrupt at the block limit
	2000	DF HW interrupt signal, level triggered
	2001	HW interrupt signal, edge triggered
Error handling (see section 3.7):		
10xx	1000	Time interrupt error processing DF System stop, if event occurs and the corresponding OB is not loaded
	1001	No system stop, if event occurs and the corresponding OB is not loaded
	1200	Loop controller error processing DF System stop, if event occurs and the corresponding OB is not loaded
	1201	No system stop, if event occurs and the corresponding OB is not loaded
	1400	Cyclic error processing DF System stop, if event occurs and the corresponding OB is not loaded
	1401	No system stop, if event occurs and the corresponding OB is not loaded

[1] xx = length of block info. (words)
[2] DF = default during cold restart
HW interrupt = interrupt-driven program execution
Time interrupt = time-driven program execution
Loop controller interrupt = processing of compact closed-loop control block (see software manual for S5 135 U)

Block identifier	Parameter	Significance [1]
		Command code error processing
	1800	DF System stop, if event occurs and the corresponding OB is not loaded
	1801	No system stop, if event occurs and the corresponding OB is not loaded
		Runtime error processing
	1A00	DF System stop, if event occurs and the corresponding OB is not loaded
	1A01	No system stop, if event occurs and the corresponding OB is not loaded
		Addressing error processing
	1C00	DF System stop, if event occurs and the corresponding OB is not loaded
	1C01	No system stop, if event occurs and the corresponding OB is not loaded
		Acknowledgement delay (time-out)
	1E00	DF System stop, if event occurs and the corresponding OB is not loaded
	1E01	No system stop, if event occurs and the corresponding OB is not loaded
EEEE		End identifier

[1]) DF = default during cold restart
[2]) Runtime error = calling in a block which is not loaded, a special function group error or a transfer error

2.7 Function blocks

Function blocks (FB's) are just as much a part of the user program as e.g. program blocks. Extended function blocks (FX) have the same structure as FB's and are programmed accordingly. Compared to the organization, program, and sequence blocks, the function blocks have four essential differences:

- Function blocks can have parameters assigned to them, i.e. the actual operands with which a function block is to operate, can be varied by using formal operands.

- In contrast to organization, program, and sequence blocks, the function blocks can be programmed with an extended operation set. These supplementary operations, which are in addition to the basic operations, can only be programmed in function blocks.

- The program of a function block can only be created and documented as a statement list (STL).

- A function block call will be represented graphically as a "black box".

Function blocks represent complex, self-contained functions within the user program. A function block can either be obtained as a software product (standard function blocks on mini diskette, see catalogue ST 57) or programmed by the user himself.

2.7.1 The structure of function blocks

A function block consists of a block header and a block body (Fig. 10).

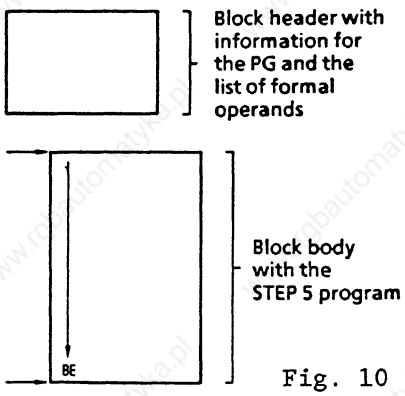


Fig. 10 The structure of a function block

- Block header

The block header contains all the information which the PG requires to be able to represent the function block graphically and to be able to check the operands during parameter assignment to the function block. Before the function block is programmed this block header is input by the user (with the support of the PG).
It is stored in the program memory of the CPU and contains a jump statement which is carried out during the function block call, but

It is stored in the program memory of the CPU and contains a jump statement which is carried out during the function block call, but which is, however, not displayed during the read out (jump over formal operand list).

- Block body

The block body contains the actual program of the function block. The function to be executed is written in the STEP 5 programming language and entered in the block body. When the function block is called only the block body is processed. An extended operation set of greater scope than the basic operations set is available for programming function blocks (see section 4.2).

2.7.2 Calling function blocks and parameter assignment

Repetitive or very complex functions are implemented by function blocks. These exist only once in the program memory and are called in once or several times by a higher-ranking block. At each call, other parameters can be used.

Function blocks are stored in the program memory under a particular designation (FB 0 to FB 255). User function blocks should be addressed in descending order, starting from FB 255, so as not to collide with the standard function blocks which are addressed from FB 1 to 199.

FB 0 should only be used for the programming of cyclic program execution, instead of OB 1 (see section 3.6.1).

The function block call can be programmed within an organization, program, or sequence block or within another function block. The call consists of the call statement and the parameter list.

- Call statement

- Unconditional call (JU FBn for function blocks or BC FXn for extended function blocks):

  The function block addressed is processed independently of the result of the previous logic operation.

- Conditional call (JC FBn for function blocks or BCC FXn for extended function blocks):

  The function block addressed will only be processed if the result of the previous logic operation is RLO = 1. If RLO = 0, the jump statement will not be executed. In **both** cases, RLO will be set to 1 by the conditional jump statement.

Following the unconditional and conditional call, the result of the logic operation can be evaluated but cannot, however, be further operated on. It is taken into the function block addressed with the jump.

- Parameter list

The parameter list is in the block which is calling directly following the call statement (Fig. 11). In the call statement, the input and output variables, as well as data, are defined (see section 2.7.3 "Classes of Block Parameters").

The parameter list can contain a maximum of 40 variables. It allocates the variables (actual operands, see following example) to the formal parameters (formal operands) of the function block.

When the function block program is executed the variables from the parameter list will be used instead of the formal parameters. The PG monitors the sequence of variables in the parameter list.

A jump statement following the FB call is automatically inserted by the PG, but is not displayed during the readout. The FB call, the jump statement, and each parameter occupy one memory word each, the program memory and the FX call occupy two memory words. Exception: each floating point number (KG parameter type) occupies two memory words.

Example (calling a function block and transferring parameters with the STL and LAD/CSF methods of representation in a program block)

- STL method of representation

```
PB25

     :  JU FB 201
NAME : E-ANTR
ZU-E :     DW   1
RME  :     I    3.5
ESB  :     F    2.5
UEZ  :     T    2
TIME :     KT  10.1
ZU-A :     DW   1
BEU  :     Q    2.3
LSL  :     Q    6.0
```

Formal operands — Actual operands

- LAD/CSF type of representation

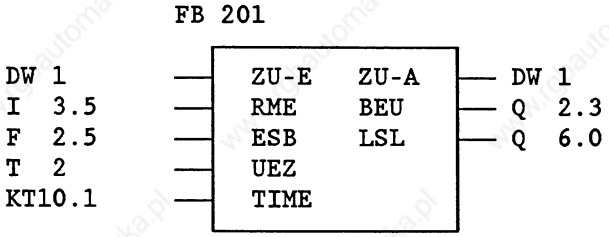


Fig. 11 Function block call

The input/output identifiers of the function block as well as the name of the function block, which appear on the PG during programming, are stored in the function block itself. Before starting to program with the PG, all necessary function blocks must therefore be transferred onto the program diskette or input directly into the program memory of the PC (for more details, see operating instructions of the PG).

2.7.3 Programming function blocks

In keeping with the structure of a function block, the generation is divided into two parts:

Before the block body is input (STEP 5 program) the block header is entered. The block header contains:

- the library number,
- the name of the function block,
- formal operands (the names of the block parameters),
- the class of block parameter,
- the type of block parameter.

• Library number

A number from 0 to 99 999 can be assigned. This number is allocated to the function block, regardless of its symbolic or absolute parameter.

A library number should only be specified once in order to be able to identify a particular function block uniquely. Standard function blocks have a product number.

• Function block name

The name which designates the function block can be up to 8 characters long. It is not identical to the symbolic plant identifier.

• Formal operand (block parameter name)

The formal operand can be up to 4 characters long, and must start with a letter. Up to 40 parameters can be programmed per function block.

Class of block parameter

The class of block parameter may be either I, Q, D, B, T or C.

I = input parameter
Q = output parameter
D = data
B = command
T = timer
C = counter

I, D, B, T or C are parameters which, in the case of graphical representation, appear on the left-hand side of the function symbol. Parameters designated by Q appear on the right-hand side of the function symbol.

Type of block parameter

For the I, Q and D classes of parameter, the type of parameter must also be specified:

BI/BY/W/D	for I and Q parameter classes
KM/KH/KY/KS/KF/KT/KC/KG	for the D parameter class

With I and Q parameters the type of parameter specifies whether bit sizes, byte sizes, word sizes or doubleword sizes are used and which data format is valid for the D parameter (see PG programming instructions).

Class of parameter	Type of parameter	Legal actual operands
I, Q	BI for an operand with bit address	I n.m inputs Q n.m outputs F n.m flags
	BY for an operand with byte address	IB n input bytes QB n output bytes FB n flag bytes DL n data byte left DR n data byte right PB n peripheral bytes OB n peripheral bytes from the extended peripherals
	W for an operand with word address	IW n input words QW n output words FW n flag words DW n data words PW n peripheral words OW n peripheral words from the extended peripherals
	D for an operand with doubleword address	ID n input doublewords QD n output doublewords FD n flag doublewords DD n data doublewords
D	KM for a binary pattern (16 bits) KY for a byte serial number from 0 - 255	constants

Class of parameter	Type of parameter	Legal actual operands
D	KH for a hexadecimal pattern up to 4 digits KS for a character (max. 2 alpha-numeric characters) KT for a time value (BCD coded) with a time base of 1.0 to 999.3 KC for a counter value (BCD coded) of 0 to 999 KF for a fixed-point number from -32768 to +32767 KG for a floating point number	constants
B	No type specification permitted	DB n data blocks; the C DB n command is executed FB n function blocks (only permissible without parameters) are called unconditionally (JU ..n) PB n program blocks are called unconditionally (JU ..n) SB n sequence blocks are called unconditionally (JU ..n)
T	No type specification permitted	T 0 to 127 timer [1]
C	No type specification permitted	C 0 to 127 counter [1]

[1] The timer or counter value should have parameters assigned to it as data or should be programmed as a constant in the function block.

Example (programming a function block)

FB 202

```
NAME : EXAMPLE
DECL : MIKE     I/Q/D/B/T/C : I  BI/BY/W/D :  BI Formal
DECL : BERT     I/Q/D/B/T/C : I  BI/BY/W/D :  BI operand list
DECL : MAUD     I/Q/D/B/T/C : Q  BI/BY/W/D :  BI
```

```
     : A = MIKE                                  STEP 5 program
     : A = BERT
     : = = MAUD
```

Formal operands | Parameter class | Parameter type

Example (function block call in a program block)

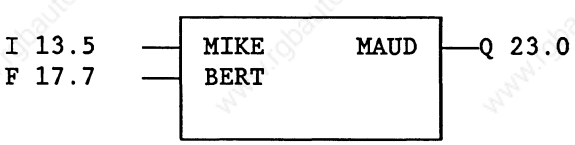
- STL representation

PB25

```
    : JU FB 202
NAME : EXAMPLE
MIKE :   I 13.5
BERT :   F 17.7
MAUD :   Q 23.0
```

Formal operands | Actual operands

- LAD/CSF method of representation



- Program executed

```
     : A  I 13.5
     : A  F 17.7
     : =  Q 23.0
```

Operations (substitution commands) to which parameters are to be assigned, are programmed (formally) in the function block with the formal operands. It is also possible to reference the formal operands several times at different places in the function blocks.

During the function block call, the formal operands are substituted by the actual operands of the parameter list.

Caution! If the order or number of formal operands in the function block header is changed, the substitution commands in the function block program and the parameter list in the block which is calling must be modified accordingly.

Example (standard function block)

Example of a standard function block

```
EXTR:                                FB 6  for 115 A
      floating point root extractor  FB 6  for 135 A
GP                                   FB 19 for 150 A
```

The ROOT:GP function block extracts the root of a floating point number (8 bit exponent and 25 bit mantissa). The result is also a floating point number (8 bit exponent and 24 bit mantissa) whereby the LSB of the mantissa will not be rounded.

If applicable the function block sets the identifier "radicand negative" for further processing.

Numerical range:

radicand	-0.1469368 exp. -38 to +0.1701412 exp. +39
root	+0.3833434 exp. -19 to +0.1304384 exp. +20

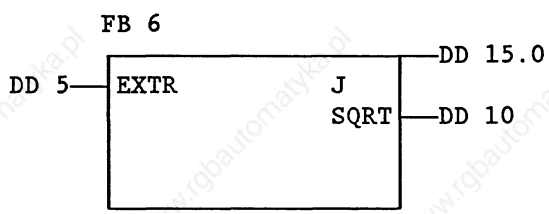
Function: Y = A
Y = SQRT; A = EXTR

Function block call:

- STL representation

```
        : JU  FB 6
NAME    : ROOT: FP
EXTR    : DD 5
J       : D 15.0
SQRT    : DD 10
```

- LAD representation



DD = data doubleword

In the above example, the root of a floating point number, which is available in DD 5 with an 8 bit exponent and a 24 bit mantissa, is extracted. The result, which is again a 32 bit floating point number, is stored in DD 10. The appropriate data block must be selected beforehand. The J parameter (class of parameter: Q, type of parameter: BI) specifies the sign of the radicand: J = 1 with negative radicands. Occupied flag words: FW 238 to 254.

The software manual of the S5 135 U shows the standard function blocks for the S5 135 U, their runtimes, their memory space requirements and the variables occupied by them.

General notes

If standard function blocks are used, the flag bytes 200 to 255 are occupied and are no longer available to the user.

The timer 0, the counter 0 and DB 0, 1, 2 and DX 0 data blocks are also occupied.

Standard function blocks occupy the numbers 1 to 199. User function blocks can therefore only be created with the numbers 200 to 255 if standard function blocks are used.

The function block FB 0 is called in cyclically by the system program instead of the organization block OB 1, if OB 1 is not programmed.

2.8 Organization blocks

The organization blocks constitute the interface between the system program and the user program. The organization blocks OB 1 to OB 39 are a part of the user program, just like program, function or sequence blocks. Organization blocks are called by the system program. The user can program the organization blocks OB 1 to 39 and thus have a direct influence on the system program.

For testing purposes these organization blocks can also be called by the user (JU/JC OBxxx).

By appropriately programming the organization blocks the following modes of operation (see section 3.5 to 3.7) can be set:

- cyclic processing (OB 1 or FB 0)
- interrupt-driven processing (OB 2)
- time-driven processing (OB 13)
- cold restart (OB 20)
- manual warm restart (OB 21)
- automatic warm restart (OB 22)
- interrupt handling (OB 19, 23 to 34)

- Organization blocks in the R processor

Absolute parameter	Designation or processing initiation
OB for cyclic processing (see section 3.6.1):	
OB 1	Cyclic processing
OB for interrupt-driven processing (see section 3.6.2):	
OB 2	Process-interrupt processing
OB for time-driven processing (see section 3.6.3):	
OB 13	Time base with 0.1 s
OB's for the modes of start-up (see section 3.5):	
OB 20	Cold restart
OB 21	Manual warm restart
OB 22	Automatic warm restart

Absolute parameter	Designation or processing initiation	Reaction without OB
OB's for interrupt handling (see section 3.7):		
OB 19	Calling a block which is not loaded	Stop
OB 23	Acknowledgement delay during single access to peripheral modules or other S5 bus addresses (AKD)	None
OB 24	Acknowledgement delay during process image updating and transmission of IPC flags (AKD)	None
OB 25	Addressing error (ADF)	Stop
OB 26	Cycle time exceeded (CYC)	Stop
OB 27	Substitution error	Stop
OB 28	Stop from PG stop switch, and halt from S5 bus [1])	Stop
OB 29	OP code not permitted	Stop
OB 30	Parameter not permitted	Stop
OB 31	Special function group error	Stop
OB 32	Transfer error in the data block	Stop
OB 33	Time interrupt processing requested while the last interrupt processing is still active	Stop
OB 34	Error during loop controller processing	Stop

[1]) OB 28 is called before transition to the stop status. The stop is always carried out regardless of whether or how OB 28 is programmed.

Absolute parameter	Designation or processing initiation
OB for special functions:	
OB 40-255	See the following list

Apart from the organization blocks OB 1 to 39, system program special functions can be called as organization blocks in the S5 135 U with numbers > 39. These organization blocks for special functions cannot be programmed but only called by the user. They do not contain a STEP 5 program. Section 5 describes the special functions individually.

Special function OB's can also be called within the organization blocks OB 1 to 30 (from S0 A03 or S1 A01 software release for the PG 675).

List of the special functions so far implemented in the R processor

OB 216-218	Access to pages
OB 220	Converting accumulator 1 from 16 to 32 bit fixed point number by means of sign extension
OB 221	Setting and triggering new cycle time
OB 222	Subsequent triggering of cycle time
OB 223	Stop with non-uniform start-up mode in multiprocessor operation
OB 224	Block transfer of the IPC flags in multiprocessor operation
OB 226	Byte-serial reading of the contents of the system program memory location
OB 227	Reading the cross-checksum of the system program memory
OB 230-237	System program auxiliary functions
OB 240	Initializing a shift register
OB 241	Calling a shift register
OB 242	Erasing a shift register
OB 250	Initializing a PID loop controller
OB 251	Calling a PID loop controller
OB 254	Copying data block DX
OB 255	Copying data block DB

3 Operation

3.1 Overview of the operating statuses

Both with single processor and multiprocessor operation, there are different operating statuses:

- stop status
- start-up
- program execution

Each operating status can be divided into three types (see Fig. 12).

Stop	Start-up	Program execution
"STOP" LED flashes quickly = warning	Cold restart	Cyclic
"STOP" LED is continuously lit	Manual warm restart	Interrupt-driven
"STOP" LED flashes slowly = error message	Automatic warm restart	Time-driven

Fig. 12 Operating statuses

The operating statuses are displayed by the LED's on the front panel of the CPU:

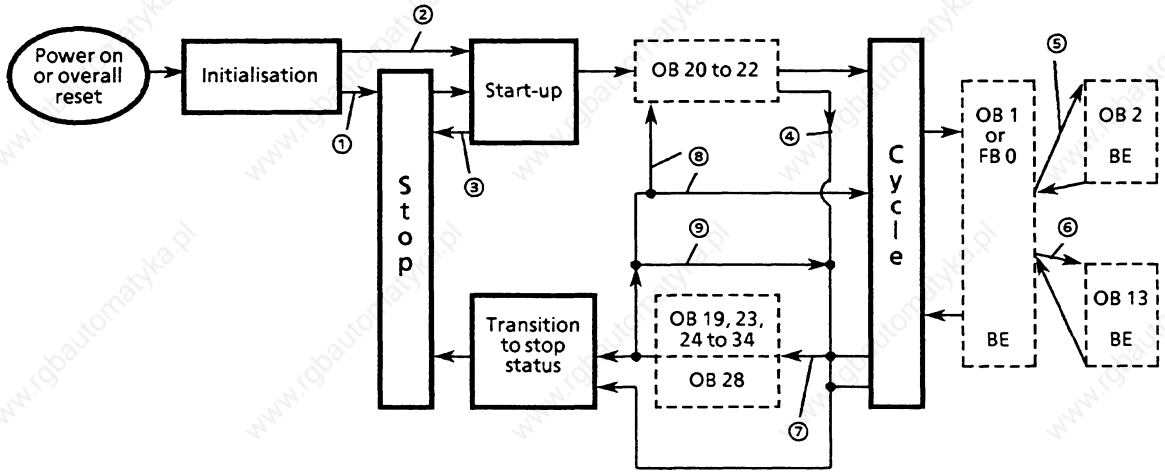
- In the stop status, the red "STOP" LED lights up. Different types of stop are indicated by the LED being lit continuously, or flashing quickly or slowly.

- During **cyclic** program execution, the green "RUN" LED lights up.

- The digital outputs are only enabled at the start of cyclic program execution. In the start-up or stop status, the BASP signal (disable command output) is output by the CPU. This is also indicated by the "BASP" LED, and all digital outputs are disabled (exceptions, see section 3.4).

Fig. 13 shows the structure underlying the system program. The organization blocks constitute the user interface to the system program. They are called by the system program in the various operating statuses and are used for programming user reactions. The system program executes its functions irrespective of whether the organization blocks are programmed. If, for example, the start-up OB is not present, the CPU still starts cyclic operation.

The commissioning of the CPU for single or multiprocessor operation either with a RAM or EPROM module is described in the central controller operating instructions.



———— system program
- - - - - - - - user program

1 Errors have been detected during initialization (see section 3.2) or operating mode switch at "STOP" or status was previously stop or automatically following overall reset.
2 Automatic warm restart following "power on" (see section 3.5.3) with continuation of the user program from the point of interruption.
3 Error during start-up before user start-up OB's called, e.g. DB 1 error (see section 3.7).
4 Causes of trouble during processing of user start-up OB's (see section 3.7).
5 Interrupt-driven program execution (see section 3.6.2).
6 Time-driven program execution (see section 3.6.3).
7 Call corresponding organization block when certain causes of trouble arise (see section 3.7).
8 Continuation of the user program from the point of interruption.
9 Cause of trouble in the error OB.

Fig. 13 The structure underlying the system program

3.2 Initialization

After the power supply has been switched on and while the overall reset is being carried out (see section 3.3.1), irrespective of the previous operating status, the CPU runs through an initialization routine which has the following effect:

Following "power on":

- the contents of the user memory and of the CPU RAM are checked
- the block address list (DBO) is erased and set up again based on the user memory and the DB RAM.

Following the overall reset:

- the RAM's are erased
- otherwise as after "power on".

If errors have been detected during the initialization, the CPU goes into the stop status and the LED flashes quickly:

a) The contents of the RAM's are incomplete, e.g. owing to interruption of the buffering before the power was switched on.
b) The user module is not plugged in or an EPROM is empty.

Quick flashing means a request for overall system reset. The cause of trouble must be dealt with, and following this, a CPU overall reset must be carried out (see section 3.3.1), this must also be carried out after the CPU is plugged in for the first time (see a)).

If the CPU was in cyclic operation before "power off", the system program executes an automatic warm restart following "power on", provided the switch positions remain unchanged on the CPU and COR (see section 3.5.3).

3.3 Stop status

3.3.1 The "STOP" LED flashes quickly (warning)

Significance: overall reset request by the system or the user.

- Overall reset request by the system

If the overall reset request comes from the system, after an error has been detected during initialization, then a CPU overall reset must be carried out by:

- holding the selector switch in the "OVERALL RESET" position; at the same time, switching the operating mode switch from "STOP" to "RUN" then back to "STOP" again.
- or using the "PC overall reset" PG function (see the operating instructions of the PG).

Result: the overall reset is executed (see section 3.2). The "STOP" LED is lit continuously.

- Overall reset request by the user

The overall reset can also be requested by the user:

- Transition to the stop status by switching the operating mode switch from "RUN" to "STOP".
- Hold the selector switch in the "OVERALL RESET" position; at the same time switch the operating mode switch from "STOP" to "RUN" then back to "STOP" again.

Result: the "STOP" LED flashes quickly (warning "overall reset" is requested).

Following this, initiate the overall reset [1]):

- Hold the selector switch in the "OVERALL RESET" position; at the same time switch the operating mode switch from "STOP" to "RUN" then back to "STOP" again.
- Or use the "PC overall reset" PG function (see the operating instructions of the PG). With overall reset using the PG, the manual overall reset request activated by means of the switch can be omitted.

Result: the overall reset is executed (see section 3.2). The "STOP" LED is lit continuously.

Subsequently, only the "cold restart" start-up mode is possible.

3.3.2 The "STOP" LED is lit continuously

Indicates the stop status of the CPU

- when the operating mode switch is switched from "RUN" to "STOP" (not following operator error such as selection of an illegal start-up mode or DB 1 error; in this case the CPU had not yet started the cycle);
- with device errors which are not attributed to one individual CPU (BAU, PEU, NAU, see section 3.7);
- as a result of the "PC stop" PG function on the CPU in multiprocessor operation;
- following overall reset;
- in multiprocessor operation owing to the PG functions "PC stop", BARB, BARBEND on another CPU;
- in multiprocessor operation owing to the halt signal from the COR (when the operating mode switch of the COR is actuated following "STOP" or when another CPU is stopped by trouble in order to identify the CPU's which are not causing the problem; exception, see section 3.4).

3.3.3 The "STOP" LED flashes slowly

Indicates the CPU which is causing trouble in both single and multiprocessor operation.

- with programming errors and device errors (ADF, CYC, SUF, AKD, TRAE..., see section 3.7);
- with operator error (DB 1 error, selection of an illegal start-up mode);
- with the programming of a stop command in the user program;
- with the "PC stop" PG function (only in single processor operation) and with the BARBEND PG function on the CPU;

[1]) If the overall reset requested by the user is not to be executed, a start-up mode must be selected at this point following the overall reset request (see section 3.5).

- Reactions during transition to the stop status

- Output of the BASP signal, which disables all digital outputs and is indicated by the "BASP" LED (exceptions, see section 3.4).
- The user organization block OB 28 is called, and the interrupt stack is set up as a result of certain causes of trouble (see section 3.7).
- In multiprocessor operation, the whole PC is put into the stop status if one CPU stops (exceptions, see section 3.4).

- Leaving the stop status

- Overall reset, then cold restart
- Selection of a start-up mode
- Test operation

3.4 Test operation

With the test operation it is possible to start up individual CPU's in a multiprocessing system (or any desired combination of CPU's) without the CPU's in the stop status blocking the whole PC. The following special features should be noted here:

- The start-up of the individual CPU's is not synchronized. Depending on the length of the organization blocks called at the start-up (OB 20, 21, 22) the CPU's start the cycle at different times.

- In the event of an error, one CPU cannot stop another. If faults occur only the CPU concerned goes into the stop status. Exception: during the overall reset, with the PG functions "PC stop", BARB, BARBEND and with a DB 1 error on one CPU, the whole PC goes into the stop status.

- The BASP signal is not output. In the event of a fault, the digital outputs are not disabled (exceptions see above).

Test operation can be switched off by adjusting the COR, so that, following the system start-up, the risky system statuses, which may occur with this function, can no longer be produced by operating the switch. For safety reasons the "test operation" function is disabled on the COR in the factory.

- Initiating the test operation

The "test operation" function must be enabled at the COR (see COR operating instructions).

- On the COR, the selector must be switched from "STOP" to "TEST"; the "BASP" LED must then go out.
- The start-up mode must be selected on the CPU's which are to go into cyclic operation (see section 3.5).

3.5 Start-up

The system program of the CPU has three different start-up modes:

- Cold restart
- Manual warm restart
- Automatic warm restart (only following power failure).

The permissible start-up mode is displayed by the PG during an error analysis in the stop status (see section 3.7, control bits NEU-ZUL, MWA-ZUL).

For each start-up mode, the system program calls an organization block which the user can program to determine the events during start-up. If this is not required, these organization blocks do not need to be programmed.

3.5.1 Cold restart

To initiate the cold restart:

- hold selector switch in the "RESET" position and at the same time change the operating mode switch from "STOP" to "RUN";
- in multiprocessor operation, following the cold restart of an individual CPU:
  start the COR (exceptions, see section 3.4) or use the PG function "PC start" provided that the switches on all CPU's and on the COR are still in the "RUN" position.

A cold restart will only be executed in multiprocessor operation if a DB 1 is present in every CPU.

The system program then:

- resets all variables (flags, IPC flags, timers, counters, PIO); erases digital and analog I/O's;
- accepts the address lists (DB 1) for inputs and outputs or IPC flags and compares them with the I/O's or IPC flags, which acknowledge on the corresponding modules;
- calls the organization block OB 20.

The data from the initialization are retained (in particular the block address list DB 0). If in multiprocessor operation no DB 1 has been programmed or if digital inputs or outputs or IPC flags specified in DB 1 do not acknowledge on corresponding modules, the CPU goes into the stop status signalling an error (for DB 1 error, see section 3.7). After the error has been corrected, a cold restart must be carried out once again.

In the organization block OB 20, the user can store a program which carries out particular activities once before the start of the cyclic program execution, e.g. sets flags, starts timers, sets outputs and, if necessary prepares the data exchange between the PC and I/O devices. OB 20 must be completed with BE (block end). After OB 20 has been processed, the cyclic execution starts by calling OB 1 or FB 0.

3.5.2 Manual warm restart

To initiate the manual warm restart:

- the selector switch must be in the middle position;
- change the operating mode switch from "STOP" to "RUN";
- in multiprocessor operation, following cold restart of an individual CPU: start COR (exceptions, see section 3.4) or use the PG function "PC start" provided that the switches on all the CPU's and on the COR are still in the "RUN" position.

During manual warm restart the results acquired before the PC stopped and the previous operating statuses are taken into account, i.e. flags and IPC flags are not erased. The time values and counts which were current at the time of the interruption of cyclic program execution are also retained, as are the data from the cold restart (DB 1) and from the initialization (DB 0). OB 21 is called as a user interface; its function corresponding to that of OB 20 in the cold restart.

The halted program execution (not the start-up program) is then continued (under BASP) with the next statement due to be processed, before the halt. The PIO and the digital outputs which are allocated to the CPU will be erased at the end of the halted cycle, to prevent the outputs being influenced by the interruption. Finally a new cycle begins by calling OB 1 or FB 0 after resetting the BASP signal, which has been disabling all the digital outputs.

3.5.3 Automatic warm restart

Initiation of the automatic warm restart:

- going from "power off" to "power on" within the cycle, if
- the switches on all CPU's and on the COR remain unchanged on "RUN"
- no errors have occurred during initialization and the user program is unchanged.

When there has been a power failure, the PC tries automatically to carry out a warm restart when the power returns. In this case, the system program first calls the organization block OB 22, in which the user can program the defaults of particular statuses. Otherwise, the function of the automatic warm restart is identical to the manual warm restart. If the PC is not intended to carry out an automatic warm restart, the STP (stop) statement must be programmed in OB 22.

```
OB 22 : STP (stop)
      : BE (block end)
```

3.5.4 Start-up in multiprocessor operation

In multiprocessor operation, the COR must be started after the individual CPU's (exceptions, see section 3.4):

- change selector switch on the COR from "STOP" to "RUN" or from "STOP" to "TEST"

- or using the PG function "PC start" on the CPU which is causing the stoppage, provided that the switches on all CPU's and on the COR remain unchanged on "RUN".

During a start-up of the PC in multiprocessor operation by starting the COR:

- the operating mode switches on all the CPU's must be on "RUN";
- the type of start-up of each individual CPU will be according to the settings made while in the stop status. It is therefore possible that some CPU's will carry out a manual warm restart, and others a cold restart. If the PC was in cyclic operation before the stop occurred "incorrect" status information can therefore be passed from one CPU to another via the IPC flags, unless this is prevented by programming the start-up organization blocks OB 20 to 22 appropriately. Flags are handled differently in the various start-up modes.

Starting up the PC in multiprocessor operation **solely** by starting the COR is only possible if the PC was only stopped by the COR. In this case, all the CPU's carry out a manual warm restart. This start-up mode also applies to those CPU's which did not cause the PC to stop and which have been stopped by the halt signal from the COR (when the halt signal is reset they go into a manual restart provided that their operating mode switch settings have not been altered in any way).

With the automatic warm restart, the COR is automatically started as well. The start-up mode is the same for all CPU's in the PC.

In multiprocessor operation the start-up of the individual CPU's is chronologically synchronized (not in test operation), i.e. the CPU's remain in a wait loop until they have all finished their start-up procedures, and then all go into cyclic operation together.

When starting using the PG, the switches of all CPU's and COR's must be set to "RUN". The CPU being operated by means of the PG carries out the selected start-up, the others carry out a manual warm restart, as long as the operating mode switch has not been changed (as with the halt signal reset performed by the COR).

3.6 Program execution

3.6.1 Cyclic program execution

Cyclic program execution is the normal type of execution with programmable logic controllers (Fig. 14). The processor begins the program execution at the start of the STEP 5 program, works through the STEP 5 statements one after another until the end of the program, and then begins processing again at the start of the program.

The organization block OB 1 or the function block FB 0 is the interface between the system program and the cyclic execution of the user program. The first STEP 5 statement in OB 1 is also the first statement of the user program, i.e. synonymous with the program start (without OB 20 to 22). If OB 1 **and** FB 0 are programmed, only OB 1 will be processed by the system program.

In OB 1 or FB 0, the program, function and sequence blocks of the cyclic program are called. In these blocks there may be further block calls, i.e. the blocks can be nested up to a depth of 24 blocks. This value is arrived at by taking the sum of the nesting depth resulting from the various types of program execution (cyclic, interrupt-driven, time-driven and if applicable interrupt handling).

- Scan time

The runtime of the user program is the sum of the runtimes of the blocks which have been called. If a block is called n-times, its runtime must be taken into account n-times. The sum of the runtimes of all the parts of the user program executed (cyclic plus time-driven plus interrupt-driven plus interrupt handling if applicable), except for the user start-up program (OB 20 to 22) adds up to the scan time of the user program for <u>one</u> program run - from the OB 1 or FB 0 call to its completion (BE).

The total scan time is the sum of this user program runtime and of the runtime for the cyclic part of the system program (see Fig. 14). This total scan time is monitored by the system program. It is normally set to the maximum permissible value of 150 ms. The user can set the scan time to be monitored in OB 221 (see section 5.3). If this time has elapsed before the system program can be retriggered (Fig. 14), the CPU is stopped with the CYC error message (see section 3.7).

By calling the operating system special function "scan time triggering" (OB 222, see section 5.3), the scan time can also be retriggered by the user program, i.e. whenever OB 222 is called, the "inner clock" will be started again for the standard monitoring time, or the monitoring time specified in OB 221.

- Process image

The data of the digital inputs and outputs (P-peripherals with byte addresses from 0 to 127) are only exchanged once per user program execution cycle between the CPU and I/O modules. The data are temporarily stored in the process image of the inputs (PII) and outputs (PIO) in the system data memory of the CPU. Compared with direct access to the I/O modules this improves the execution time of the corresponding commands (see section 4.2) and avoids the outputs "chattering" because of frequent switching over within a processing cycle.

- IPC flags

The IPC flags are used for the data exchange between the individual CPU's or between CPU and CP's (see section 2.3.1). Just as the process image, they are read-in cyclically by the CPU (IPC flag inputs) or output (IPC flag outputs).

- Indicators

During error-free cyclic operation, the green "RUN" LED lights up. The digital outputs are only enabled during the cycle; the red "BASP" LED then goes out. The red "STOP" LED and the error LED's are also unlit.

- Interrupt points

The cyclic program execution can be interrupted by:

- interrupt-driven program execution (see section 3.6.2)
- time-driven program execution (see section 3.6.3)
- loop controller processing (see description of application of the compact closed-loop controller in the S5 135U/R processor)
- interrupt events (see section 3.7).

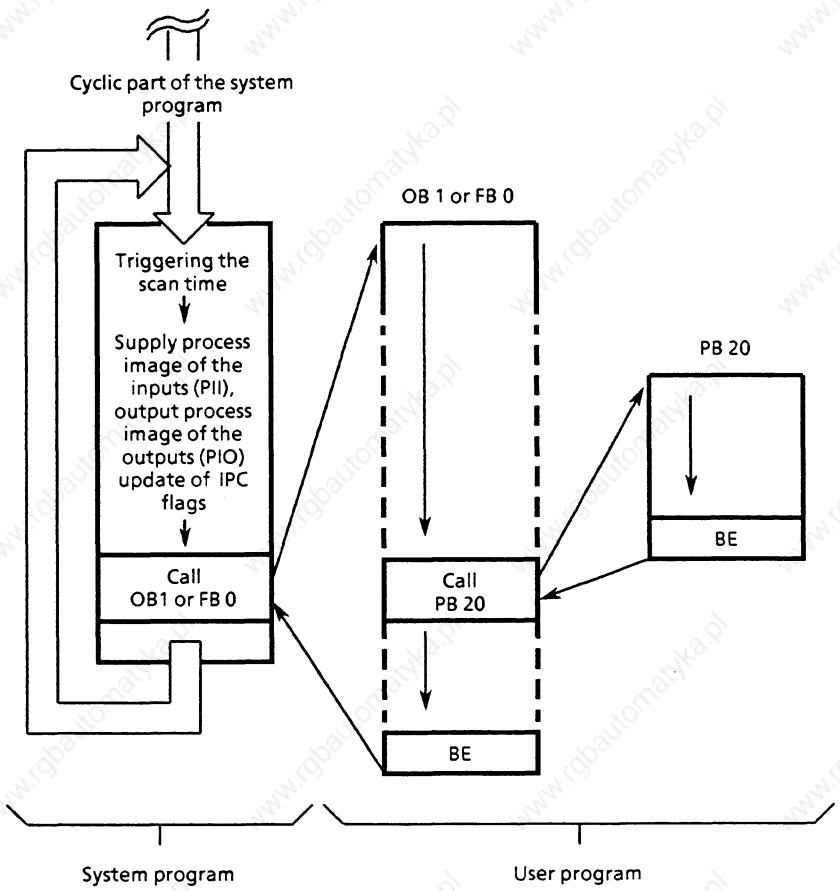


Fig. 14 Cyclic program execution

- General structure of the user program

OB 1 or FB 0 contains the general structure of the user program. The documentation of this block is intended to show the basic program structures (Fig. 15) or emphasize the parts of the system which are connected in terms of the program (Fig. 16).

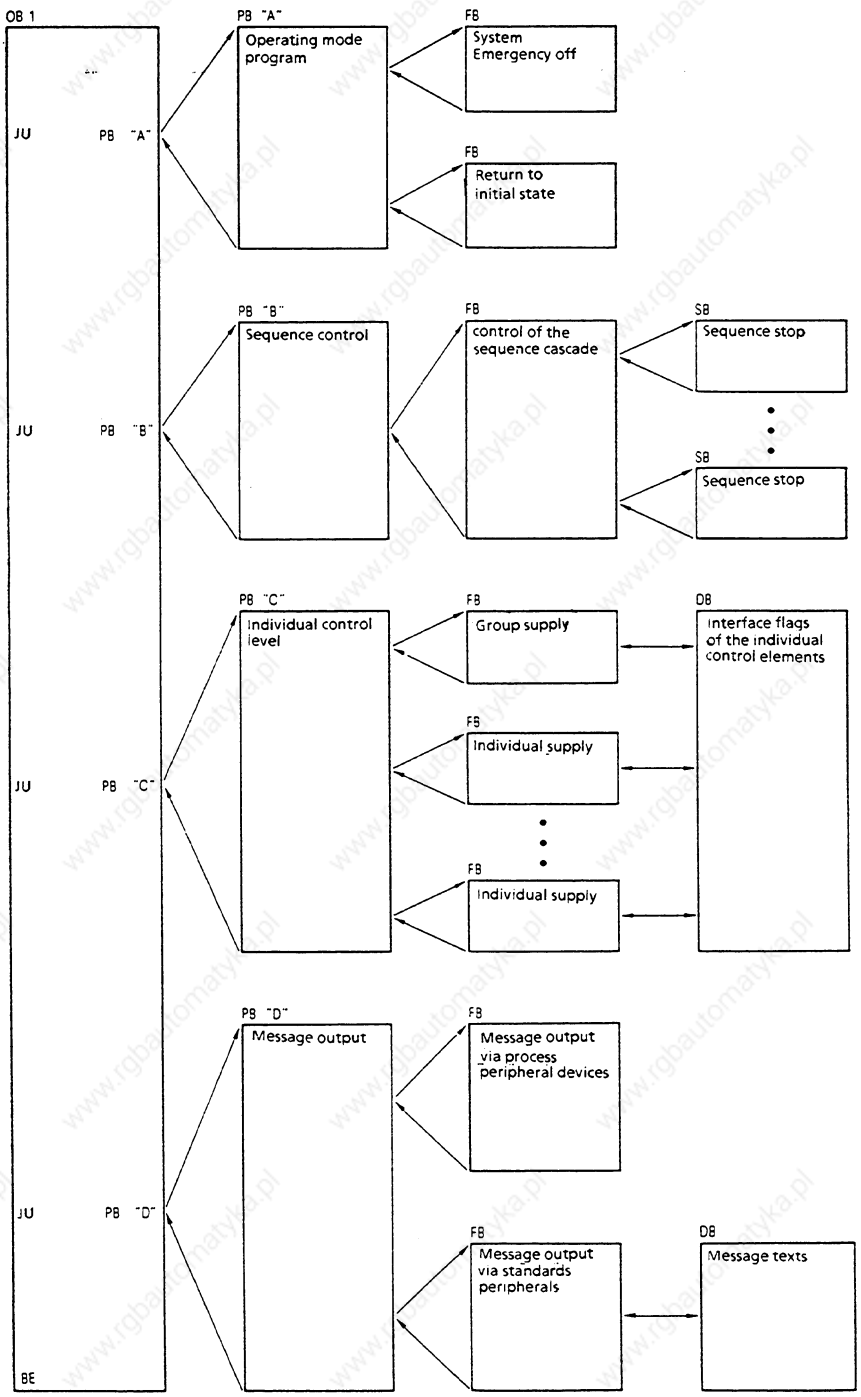


Fig. 15 Basic structure of the user program related to program structure

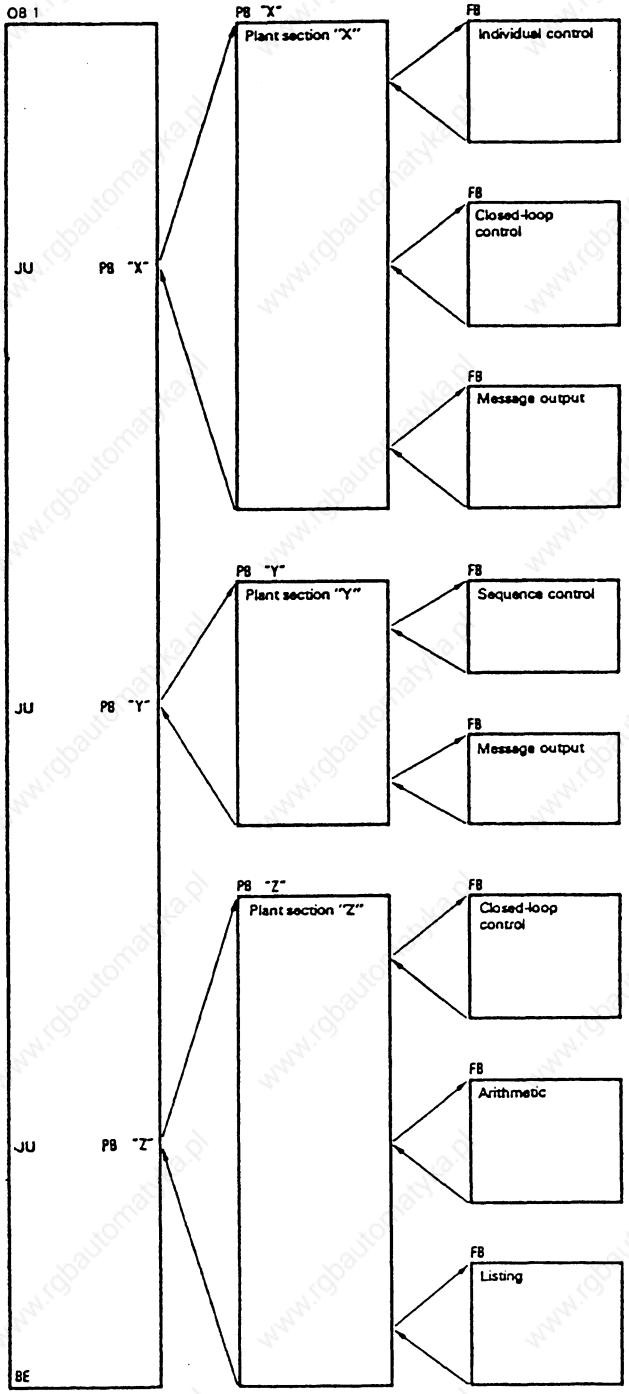


Fig. 16 Basic structure of the user program related to system structure

Note

The arithmetic registers, accumulators 1, 2, 3 and 4 cannot be used as data storage beyond the limits of the cycle (i.e. from the end of one program cycle to the start of the next) as they are required by the system program.

3.6.2 Interrupt-driven program execution

With the S5 135 U, interrupt-driven program execution can be carried out. In this operating mode cyclic program execution is interrupted by an interrupt signal from an I/O module (see operating instructions for the R processor). The system program calls OB 2 as a user interface, in which the user can have a specific program executed. After this program has been executed the processor goes back to the point of interruption and continues the cyclic processing from there.

Further interrupt requests will only be accepted after OB 2 has been processed. The cyclic program execution will be interrupted again at the next block change or STEP 5 command (depending on the presetting) by the interrupt-driven program execution.

Process-interrupt processing means the user can react directly to process signals.

- Points of interruption

Cyclic program execution cannot be interrupted at random by interrupt-driven processing. This is normally only possible at the block boundaries. When a change is made from one block to another (by calling in a new block, or by returning to the higher ranking block following a block end statement) the system program can call in an organization block for the interrupt-driven processing.

On the other hand, by choosing a presetting in DX 0 (see section 2.6), the user has the option of making the cyclic program interruptable at the STEP 5 command boundaries. Loop controller blocks which are written in assembler language, contain pseudo command boundaries.

Interrupt-driven processing can only be interrupted by disturbances (see section 3.7), not by time-driven processing, or a renewed request from the interrupt-driven processing.

- Disabling interrupt-driven processing

An interrupt-driven program is inserted into the cyclic program at a block boundary or a STEP 5 command boundary. At this point, the cyclic program will be interrupted. This interruption can have a negative effect, if a cyclic program section is time-critical, when e.g. a particular reaction time must be achieved.

If a program section must not be interrupted by interrupt-driven processing, the following programming options are possible:

- The program does not contain a block change with the standard preset "interrupts at block boundaries". Therefore it cannot be interrupted.

- The program itself is in an interrupt-driven program. Here, too, it cannot be interrupted at block boundaries or command boundaries.

- Interrupt processing is disabled with the STEP 5 command IA (disable process interrupt). With the RA command (enable process interrupt), the interrupt processing is enabled again. Program sections between IA and RA cannot be interrupted by process interrupts.

3.6.3 Time-driven program processing

The S5 135 U can also carry out time-driven program processing. Time-driven processing is carried out if a signal coming from an "inner clock" causes the processor in the PC to interrupt normal cyclic processing and to execute a specific program.

After this program has been executed, the processor returns to the point of interruption in the cyclic program and continues processing from there.

- Interface between system program and time-driven processing

OB 13 is the interface between the system program and the time-driven processing. It is called every 100 ms by the system program. If no OB 13 is programmed, the cyclic program will not be interrupted.

- Points of interruption

Cyclic program processing can normally be interrupted at the block boundaries or at STEP 5 command boundaries by selecting the corresponding presetting in DX 0 (see section 2.6).

Time-driven program processing can be interrupted by interrupt-driven processing, loop controller processing (see description of application of the compact closed-loop control in the S5 135 U/R processor), or by device faults, but never by renewed time-driven program processing.

If after 100 ms time-driven processing is requested again before the first run has finished, the operating system recognises a "critical" status and calls OB 33. In OB 33, the user can program the required reaction to this status. After OB 33 has been processed, the program is continued at the point of interruption. If no OB 33 is programmed, the CPU stops with the WECKFE error message (see section 3.7).

If requests for interrupt-driven and time-driven processing occur at the same time, the programs for interrupt-driven processing will be processed first. In this case, the time-driven processing has the lower priority.

- Reaction time

The time taken to react to a time interrupt request corresponds to the processing time of a block or of a STEP 5 command depending on the selected presetting. If there are still process interrupts waiting when the cyclic program execution is interrupted, the time-driven program will only be processed when all the outstanding process interrupts have been serviced.

The maximum reaction time between the occurrence and the execution of the time-driven program increases in this case by the processing time required by the process interrupts. The processing time for a time interrupt, including any process interrupts which may be waiting, must not exceed 100 ms (see above).

3.7 Interrupt handling

The system program can recognise when the CPU is operating incorrectly, errors in the system program or the effects of incorrect programming by the user.

3.7.1 Interruption at STEP 5 command boundaries

The program processing is interrupted at STEP 5 command boundaries or not even started. The following situations cause the CPU to stop:

a) Stoppage caused by the PG mode selector on the CPU on "STOP" or halt signal from the COR (switch on COR on "STOP" or another CPU has stopped)
b) Stop command in ths user program
c) Signal from the S5 bus:
   BAU back-up battery failure on the central controller,
   NAU power supply failure on the central controller or
   PEU power supply failure on an expansion unit.
d) Stack overflow at the interupt stack (STUEU) or
   stack overflow at the block stack (STUEB) when the nesting depth is too great.
e) DB 1 address list missing (only in multiprocessor operation),
   incorrect address list (DB 1 error, see section 2.6) during cold restart,
   error in DB 2 loop controller parameter assignment block (DB 2 error) during start-up or
   error in DX 0.
f) Error during initialization (see section 3.2).

With a) the system program calls OB 28 as a user interface, via which the user can specify particular reactions e.g. saving variables. After OB 28 has been processed, the CPU always stops. The transition to the stop status is carried out irrespective of whether and how OB 28 is programmed. Causes of trouble in OB 28 will be recognised and dealt with as in the cyclic program.

The causes of trouble are identified in different ways:

- a) and c) the "STOP" LED is lit continuously;
- b), d) and e), the "STOP" LED flashes slowly;
- f) the "STOP" LED flashes quickly.

The cause of trouble can be analysed with the PG (see sections 3.7.2 and 3.7.3).

If the cause of the trouble occurs after the call of the user program (only a) to d)), an interrupt stack will be set up, in which the point of interruption is specified with the current condition codes and accumulator contents. The interrupt stack can be read out with the PG (see example). In addition to the functions covered by the interrupt stack output, the cause of trouble in e) and f) is defined more exactly in the system data 3 and 4 (see section 7.1).

3.7.2 Interruption with other causes of trouble

Other causes of trouble do not interfere with the operation of the CPU to such an extent. Here, the system program allows the user to determine the subsequent reaction of the CPU itself. Certain organization blocks are called in if the following problems occur,

Problem	Call	Reaction if OB not present
Calling in a block which is not loaded	OB 19	stop
Acknowledgement delay with individual access to I/O modules	OB 23	none
Acknowledgement delay during updating of process image	OB 24	none
Addressing error	OB 25	stop
Scan time exceeded	OB 26	stop
Substitution error	OB 27	stop
Illegal operation code	OB 29	stop
Illegal parameter	OB 30	stop
Special function group error	OB 31	stop
Transfer error in the data block	OB 32	stop
Error with time interrupt processing = interrupt error	OB 33	stop
Error with loop controller processing	OB 34	stop

By calling in an error organization block, the user is provided with additional error information for evaluation, via accumulators 1 and 2 (see section 7.2).

If one of these errors occurs, the user can allow the CPU to continue, stop the CPU (by programming a stop command in the corresponding organization block), or run a special program. If the organization block called is not programmed, the program execution will be continued (if OB 23 and OB 24 are not present), or the CPU will stop (if one of the other error organization blocks is not present).

- Calling a block which is not loaded

The system program recognises if a block is called in the user program which is not present or not valid. This applies to all types of blocks, both for the conditional and unconditional call.

When the call of a block which is not loaded is recognised, the system program calls in organization block OB 19. The further reaction of the CPU can be determined in this block. If OB 19 only contains the BE (block end) command, the call of a block which is not loaded will be handled in the same way as a no-operation (NOP). The execution of the interrupted STEP 5 program will be continued with the next command. If OB 19 is not programmed, the CPU stops when a block, which is not loaded, is called.

- Acknowledgement delay (AKD)

An acknowledgement (or time-out), occurs if a module does not signal back with the RDY signal (ready) within a certain time after it has been addressed. The cause of the acknowledgement delay may be a fault on the module or the module may have been removed during operation.

The following acknowledgement delay errors interrupt the execution of the user program and call a corresponding organization block:

- Acknowledgement delay when a CP, IP, COR or an I/O module (e.g. with load commands and transfer commands L/T P... or O...) has been accessed individually: the system program calls organization block OB 23.

- Acknowledgement delay during updating of the process image for inputs and outputs and transfer of the IPC flags: the system program calls organization block OB 24.

  If the organization blocks called have not been programmed, the execution of the user program will be continued. An acknowledgement extends the run time of the STEP 5 command which caused it. If, as a result of the acknowledgement delay the CPU is to be stopped, the STP stop command must be programmed in OB 23 or 24.

- Addressing errors (ADE)

An addressing error occurs if an input or output in the process image is addressed by a STEP 5 operation (see section 4.2) to which no I/O module was allocated at the time of the last cold restart (module was faulty, not plugged in or not specified in DB 1 of the CPU.

The system program now interrupts the execution of the user program and calls organization block OB 25. Following the execution of the program in OB 25, the next command of the interrupted program will be executed.

If OB 25 is not programmed, the CPU goes into the stop status whenever the addressing error occurs.

- Scan time exceeded (CYC)

The scan time can be exceeded e.g. as a result of faulty programming, if the CPU goes into a program loop at a particular process status, or if the clock generator fails or the CPU becomes overloaded (see section 3.6.1).

If the scan time is exceeded the system program interrupts the execution of the user program and calls organization block OB 26. The scan time is restarted.

If OB 26 has been programmed, the CPU stops whenever the scan time is exceeded. Irrespective of the programming of OB 26, the BASP signal, which disables the signal outputs, will be output if the scan time is exceeded. With the return to cyclic program execution the BASP signal is cancelled.

- Command code error

A command code error occurs if the CPU cannot interpret or carry out a command. The following different types of command code error are possible:

- Substitution error

  The CPU carries out a substitution during the execution of the user program within a function block, if an operation is carried out with a formal operand in a function block. The formal operand is replaced by the actual operand contained in the function block call (see section 2.7).

  The CPU recognises an illegal substitution. The system program interrupts the execution of the user program and calls OB 27.

- Illegal operation code

  An illegal operation code occurs if a command has been programmed which is not within the STEP 5 command range of the CPU (e.g. RU and SU commands can be programmed with the PG, but cannot be interpreted by the R and S processors in the S5 135 U).

  When an illegal operation code is recognised the execution of the user program will be interrupted at this point and OB 29 will be called.

- Illegal parameter

  An illegal parameter occurs if a command line has been programmed with a parameter which is illegal for the CPU concerned (e.g. time and count operations with a parameter number > 127).

  When an illegal parameter is recognised by the CPU, the system program interrupts the execution of the user program and calls OB 30.

The command which causes the corresponding command code error will not be executed, but the corresponding organization block will be called. When the program in the organization block has been executed, the interrupted user program continues with the next command.

If the corresponding organization block has not been programmed, the CPU stops.

- Special function group error

The special function group error is recognised by the system program if an error occurs while a special function organization block is being processed or if the special function is not present. In this case, organization block OB 31 will be called. The special function will not be processed further. After OB 31 has been processed, the program will continue with the command which follows the special function organization block call. If OB 31 is not present, the CPU stops.

- Tranfer error

If, during the transfer of data to data blocks (DB, DX) using the specified parameter for the transfer command, the data block length is exceeded, this will be recognised as a transfer error in order to prevent data in the memory from being overwritten by mistake. A transfer error will also be recognised if a transfer takes place to a dataword, although no data block has yet been opened (with C DBn or CX DXn).

If a transfer error is recognised, the system program calls organization block OB 32. The command which has caused the transfer error will no longer be executed. If OB 32 has not been programmed the CPU stops.

- Time-interrupt error

If a further time-interrupt request occurs while a time-interrupt is being processed (see section 3.6.3), the system program calls organization block OB 33. If OB 33 has not been programmed, the CPU stops.

- Loop controller error

An error which occurs while the loop controller function blocks of the compact closed-loop control (supported by the system program) are being processed, will be recognised as a loop controller error. If a loop controller error occurs organization block OB 34 is processed. If OB 34 has not been programmed the CPU stops. After OB 35 has been processed, the loop controller functions will be processed further.

If the execution of the program is to be continued without any further reaction when one of these errors occurs, then BE (block end) must simply be programmed in the corresponding error organization block. The program execution (cyclic, interrupt-driven, time-driven or interrupt-handling) is continued with the next STEP 5 statement after the interruption.

If a new error occurs in the error organization block it will be handled as in the cyclic program, i.e. the corresponding error organization block will be called. As a result, nested error handling is possible up to a nesting depth of seven errors. At this point the CPU stops immediately without a renewed OB call; the same happens if an error of the same type (in the same error organization block) occurs a second time.

In the start up program (organization blocks OB 20 to OB 22) causes of interruptions are handled as in the cycle. In the case of 3.7.1a, no organization block OB 28 is called in. If the start up program execution is interrupted and the CPU stops, only a cold restart is possible afterwards. Once interrupted a start up cannot be continued. In the start-up phase there is no scan time monitoring.

If program execution is to be terminated when one of these causes of interruption occurs, this is achieved by simply programming a stop command in the error organization block. The CPU then stops immediately (see 3.7.1 b).

3.7.3 Control bits and interrupt stack

By means of PG functions 'PC INFO' followed by 'output ISTACK' you can analyze the operating status, the characteristics of the processor and the user program as well as possible causes of errors and interruptions.

IMPORTANT!
Output of control bits is possible in <u>any</u> operating state, output of the ISTACK <u>only</u> in stop.

- The **control bits** indicate the current or previous operating status as well as the cause of the error.
  If several errors have occurred <u>all errors that have occurred</u> will be displayed in the control bits.

- The breakpoint (addresses) with the condition code words at that point and the contents of the accumulators as well as the cause of the error are entered in the **ISTACK**.
  If several errors have occurred a <u>multi-layer</u> interrupt stack is created:

  depth 01 = last cause of error,
  depth 02 = next to last cause of error etc..

  In the case of an ISTACK overflow an immediate stop will be executed. A cold restart is required afterwards.

The significance of the abbreviations in the control bits and the interrupt stack are explained in the following pages.

Note:
The control bits listed below can be displayed via the S5-DOS-PG software on the PG 685.

> *What to do if the text on the screen of your programmer differs from the one listed below?*
>
> *In this case follow the* **positions** *of the abbreviations displayed on the screen.*

Examples:

1. In the control bits, position 1 in line 6 has been marked. On your screen the corresponding abbreviation is "CHS-FE". In these programming instructions, however, the abbreviation "DXO-FE" has been entered for the same position (see below). The description of "DXO-FE" as stated in this manual is then applicable!

2. In line 8, the penultimate position has been marked though the corresponding abbreviation is missing. In this case, the description of "REG-FE" as stated in this manual is applicable (see below)!

C O N T R O L B I T S

>>STP<<	STP-6	FE-STP	BARBEND	PG-STP	STP-SCH	STP-BEF	MP-STP
>>ANL<<	ANL-6	NEUST	M W A	A W A	ANL-2	NEU-ZUL	MWA-ZUL
		X				X	X
>>RUN<<	RUN-6	EINPROZ	BARB	OB1GEL	FB0GEL	OBPROZA	OBWECKA
X		X			X		
32KWRAM	16KWRAM	8KWRAM	EPROM	KM-AUS	KM-EIN	DIG-EIN	DIG-AUS
		X				X	X
URGELOE	URL-IA	STP-VER	ANL-ABB	UA-PG	UA-SYS	UA-PRFE	UA-SCH
DXO-FE	FE-22	MOD-FE	RAM-FE	DB0-FE	DB1-FE	DB2-FE	KOR-FE
N A U	P E U	B A U	STUE-FE	Z Y K	Q V Z	A D F	WECK-FE
B C F	FE-6	FE-5	FE-4	FE-3	L Z F	REG-FE	DOPP-FE

The statuses of the control bits are displayed on the first page of the screen when the ISTACK is output on the PG.

The following control bits indicate the current or previous operating status of the processor and supply information about certain characteristics of the processor and the STEP5 user program.

Output of the control bits is possible in all operating conditions.

This allows you to e.g. verify that the organization block OB 2 has been loaded and whether interrupt driven program execution is possible or not.

STP processor is in the operating state STOP; the following control bits indicate why the processor has gone into stop:

STP-6	Not used
FE-STP	Error-stop: stop state after NAU (power failure), PEU (I/O not ready), BAU (battery not ready), STUEB (BSTACK overflow), STUEU (ISTACK overflow), DOPP (double error) or processor fault
BARBEND	Finish process check: stop condition after online function "process control end" (cold restart required)
PG-STP	PG stop: stop status due to command from PG in single-processor operation or multiprocessor test operation
STP-SCH	Stop switch: stop status due to stop switch in the STOP position
STP-BEF	Stop command: a) Stop status after the processing of STEP5 operation 'STP' b) Stop status after stop command by the system program if error organization block has not been programmed.
MP-STP	Multiprocessor stop: a) Selector switch on KOR in the STOP position or b) another processor has stopped during multiprocessor operation c) Stop command from the PG in multiprocessor operation

ANL Processor is in operating state START-UP:

ANL-6	Not used
NEUST	Cold restart is requested or active or was executed as the last start-up.
M W A	Manual warm restart is requested or active or was executed as the last start-up.
A W A	Automatic warm restart after power failure is requested or active or was executed as the last start-up.
ANL-2	Not used
NEU-ZUL	Cold restart permissible as the next start-up mode (if not marked, overall reset is necessary)

MWA-ZUL Manual warm restart permissible as the next start-up mode

RUN Processor is in the operating status RUN (cyclic program execution is active):

RUN-6 Not used

EINPROZ Single processor operation

BARB Online function "process control" is active

OB1GEL Organization block OB 1 has been loaded into the user memory.
Cyclic program execution is determined by OB 1.

FBOGEL Function block FB 0 has been loaded into the user memory.
Cyclic program execution is determined by FB 0 if no OB 1 has been loaded. If FB 0 and OB 1 have been loaded, then OB 1 is valid for cyclic program execution.

OBPROZA Process interrupt organization block OB 2 has been loaded, i.e. process interrupt-driven program execution is possible

OBWECKA Time interrupt organization block has been loaded, i.e. time-driven program execution is possible

32KWRAM User memory submodule is a RAM with 32 x 2^{10} words.

16KWRAM User memory submodule is a RAM with 16 x 2^{10} words.

8KWRAM User memory submodule is a RAM with 8 x 2^{10} words.

EPROM User memory submodule is an EPROM.

KM-AUS Address list for interprocessor communication flag outputs is in DB 1.

KM-EIN Address list for interprocessor communication flag inputs is in DB 1.

DIG-EIN Address list for digital inputs

DIG-AUS Address list for digital outputs

URGELOE Overall reset of processor was carried out (cold restart required)

URL-IA Overall reset of the processor being carried out

STP-VER Processor has caused stop status in the PC

ANL-ABB Abort during the start-up (cold restart required)

UA-PG PG has requested overall reset

UA-SYS System program has requested overall reset (no start-up possible); overall reset must be carried out

UA-PRFE Not used

UA-SCH Overall reset requested by operator (switch); carry out overall reset or select a start-up mode if requested overall reset is not to be carried out

The following control bits indicate errors which may occur in the operating states START-UP (e.g. in the case of the first cold restart) and RUN (e.g. in the case of time-driven program execution).

If several errors have occurred, then all the errors which so far have caused an interruption (and which are still to be processed!) will be indicated in the last three lines of the control bits. Note that all the errors which have occured and are still to be processed are also entered in the UAMK (interrupt condition code word, collected, 16 bits) which is contained in system data RS 2.

Errors during the START-UP:

DXO-FE Parameter assignment error in DX 0

FE-22 Not used

MOD-FE User module contains errors (check EPROM or perform overall reset for RAM)

RAM-FE Operating system RAM or the DB-RAM contains errors (overall reset requested)

DBO-FE Structure of the block address list in DB 0 is incorrect

DB1-FE Structure of the address list in DB 1 for updating of the process images is incorrect;

a) DB 1 not programmed with the coordinator plugged-in or in multiprocessor operation;

b) No acknowledgement from the byte addresses for inputs and outputs or interprocessor communication flags specified in DB 1 during a cold restart on the corresponding modules.

DB2-FE Error during the evaluation of the parameter assignment data block DB 2 of controller structure R64

Errors during START-UP or RUN:

KOR-FE Error during data exchange with the coordinator

NAU Power failure in the central controller

PEU I/O not ready = power failure at an expansion unit

BAU Battery defective = failure of the back-up battery (central controller)

STUE-FE Interrupt or blockstack overflow (nesting depth too great; cold restart required)

ZYK Cycle time exceeded

QVZ Acknowledgement delay during data exchange with I/O's

ADF Addressing error at inputs or outputs

(error caused by access to process image with I/O modules addressed that were not plugged-in or defective or not specified in DB 1 during the last cold restart)

WECK-FE Collision of two time interrupts:

Prior to or during the processing of a time interrupt the latter has been called a second time.

BCF Command code error:

a) Substitution error: STEP5 command processed can not be substituted

b) Operation code error: STEP5 command processed is wrong

c) Parameter error: parameter of the STEP5 command processed is wrong

FE-6 Not used

FE-5 Not used

FE-4 Not used

FE-3 Not used

LZF Execution time error:

a) Block called has not been loaded

b) Transfer error with data blocks

c) Other execution time errors
(previously: special-function group errors)

REG-FE Error during the processing of controller structure R64 in the cycle

DOPP-FE Double error:

a) A program level which is still active (ADF, BCF, LZF, QVZ, REG, ZYK) has been activated a second time (cold restart required)

b) System error (overall reset required)

After the control bits have been output on the PG screen by pressing the enter key the ISTACK will appear on the following page of the screen. Here, the system program will enter all information required for a cold restart or restart during the transition to the stop state.

Note:
The ISTACK mask shown below can be displayed on the PG 685 via the S5-DOS-PG software.

> *What to do if the text on the screen of your programmer differs from the one listed below?*
>
> *In this case follow the* **positions** *of the abbreviations displayed on the screen.*

Examples:

1. In the ISTACK, position 3 in line 2 has been marked as a cause of interruption. On your screen the corresponding abbreviation is "TRAF". In these programming instructions, however, the abbreviation "LZF" has been entered for the same position (see below). The description of "LZF" as stated in this manual is applicable!

2. In line 2, the last position has been marked as a cause of interruption though the corresponding abbreviation is missing. In this case, the description of "DOPP" as stated in this manual is applicable (see below)!

ISTACK

```
DEPTH:    02

BEF-REG:  C70A      SAC:    00F3      DBADDR:  0000    BA-ADDR:    0000
BST-STP:  0000      FB-NO.:  226      DB-NO.:             -NO.:
                    REL-SAC:0006      DBL-REG: 0000
LEVEL:    0004      UAMK:   0100      UALW:    0000

ACCU1: 0000 C464    ACCU2: 0000 00FF  ACCU3: 0000 0000  ACCU4: 0000 0000

BRACKETS: KE1 111   KE2 100  KE3 111

                                                        ____
RESULT BITS:        DSP1 DSP0 OVFL OVFLS  OR  STATUS RLO ERAB
                          X                      X     X
CAUSE OF
INTERR.:            NAU  PEU  BAU  MPSTP ZYK   QVZ   ADF STP
                                                     X
                    BCF  S-6  LZF  REG   STUEB STUEU WECK DOPP
```

The following ISTACK identifiers supply information on the location of the errors in the user program. The erroneous instruction that has caused the processor to go into the stop state can thus be found.

DEPTH Layer of the ISTACK for error nesting

DEPTH 01 = cause of last interruption
DEPTH 02 = cause of second last interruption
.....

BEF-REG Instruction register:
Contains the machine code (first word) of the command to be executed next in an interrupted program level

BST-STP Block stack pointer

LEVEL Z States the level of program that has been interrupted

Z: 0002 = cold restart or manual warm restart
0004 = cycle
0006 = time interrupt
0008 = controller
000A = process interrupt
000C = abort (stop switch, PG stop or KOR stop)
0010 = time interrupt
0012 = controller error
0014 = cycle time error
0016 = automatic warm restart
0018 = command code error
001A = execution time error
001C = addressing error
001E = acknowledgement delay

SAC STEP address counter:
contains the absolute address of the next command to be executed in an interrupted program level in the program memory.
(absolute address - 1 or - 2 = 1-word or 2-word command which has caused the interruption!)

If the error is not at the STEP5 level, e.g. during controller processing, the SAC will be '0', the contents of the BEF-REG are irrelevant.

REL-SAC Relative step address counter:
Contains the relative address (relative to the block start address) of the command to be executed next in the block processed last

(Display of relative addresses is possible in the operating mode "input inhibited" (key operated switch) or when the block is output on the printer)

UAMK Interrupt condition codeword (collected):
All the causes of interruption which have occured and are still to be processed are indicated in the UAMK.

UALW Interrupt condition inhibit word

DB-ADDR Absolute start address of the block in the program memory (DW 0) called last

(DB-ADR = 0000, if no DB has been called)

DB-NO Number of the data block called last

DBL-REG Length of the data block called last

BA-ADDR Absolute address in the program memory of the command to be processed next in the block called last

ACCU1...4 Contents of the arithmetic register at the point of interruption

To indicate interruptions, in the case of certain errors the system program will deposit error identifiers in accumulators 1 and 2. These numbers supply a more detailed explanation of the causes of the interruption.

BRACKETS Number of bracketing levels: 'KEx abc'
x = 1 up to 7 levels

a = OR (see bit condition codes)
b = RLO (result of logic operation, see bit condition codes)
c = 1: A(
c = 0: O(

RESULT BITS: see section 4.1

The following abbreviations are the most important causes of errors and interruptions. Only those causes of interruption are marked which have occured in the currently displayed program level (see LEVEL!).

The information about the causes of interruptions is taken from the interrupt condition code word (UAMK, 16 bits). Some of this information is identical with that of the control bits.

NAU Power failure in the central controller

PEU I/O's not ready = power failure in expansion unit

BAU Battery not ready = failure of back-up battery (CC)

MP-STP Multiprocessor stop:
a) Selector switch at KOR in STOP position or
b) another processor has stopped during multiprocessor operation
c) Stop command from PG during multiprocessor operation

ZYK Cycle time exceeded

QVZ Acknowledgement delay during data exchange with I/O's

ADF Addressing error at inputs or outputs

STP Stop state due to stop switch in STOP position
Stop state due to instruction from the PG in single processor operation or multiprocessor test operation
Stop state after processing STEP5 operation 'STP'
Stop state after stop command by system program if error organization block has not been programmed.

BCF Command code error:
Errors recognized during command decoding

a) Substitution errors: STEP5 command processed can not be substituted

b) Operation code error: STEP5 command processed is wrong

c) Parameter error: Parameter of STEP5 command processed is wrong

S-6 Not used

LZF Execution time error:
Errors recognized during command execution

a) Block called has not been loaded

b) Transfer error with data blocks

c) Other execution time errors
(previously: special-function group errors)

REG Error during the processing of controller structure R64 in the cycle

STUEB Block stack overflow (nesting depth too great; cold restart required)

STUEU Interrupt stack overflow (nesting depth too great; cold restart required)

WECK Collision of two time interrupts:
Prior to or during the processing of a time interrupt the latter has been called a second time.

DOPP Double error:
a) A program level which is still active (ADF, BCF, LZF, QVZ, REG, ZYK) has been activated a second time (cold restart required)

b) System error (overall reset required)

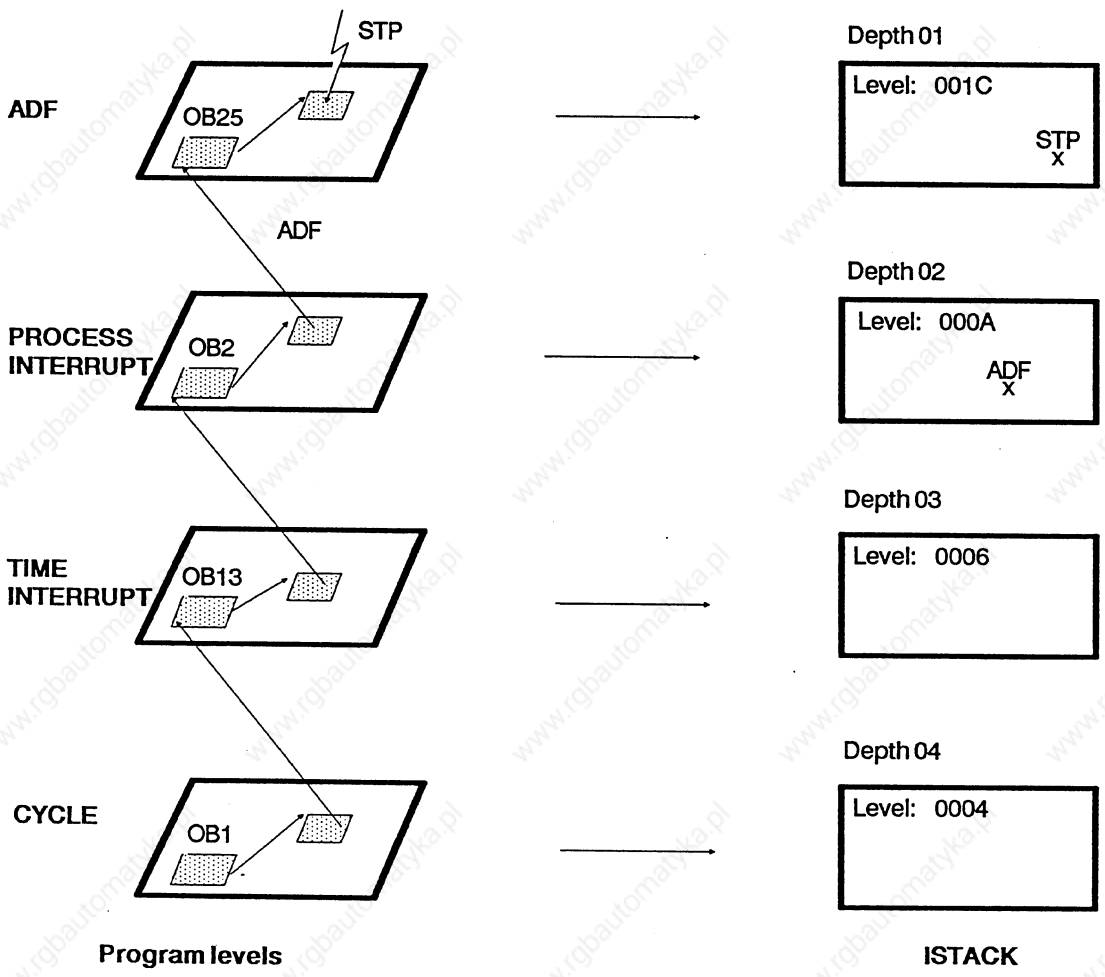
How to evaluate the ISTACK

Example 1:

The structure of the ISTACK in connection with possible interruptions is shown in the following figure.

1. The program level "CYCLE" (**OB 1**) is interrupted by a time interrupt.
2. Then, the program level "TIME INTERRUPT" is activated and **OB 13** processed.
3. A process interrupt causes the processor to exit the "TIME INTERRUPT" level. The "PROCESS INTERRUPT" level is activated and **OB 2** processed.
4. A wrong addressing instruction results in the "ADF" level being activated and **OB 25** processed. The user has programmed a stop command (STP) in the error routine: program processing is aborted by the processor.

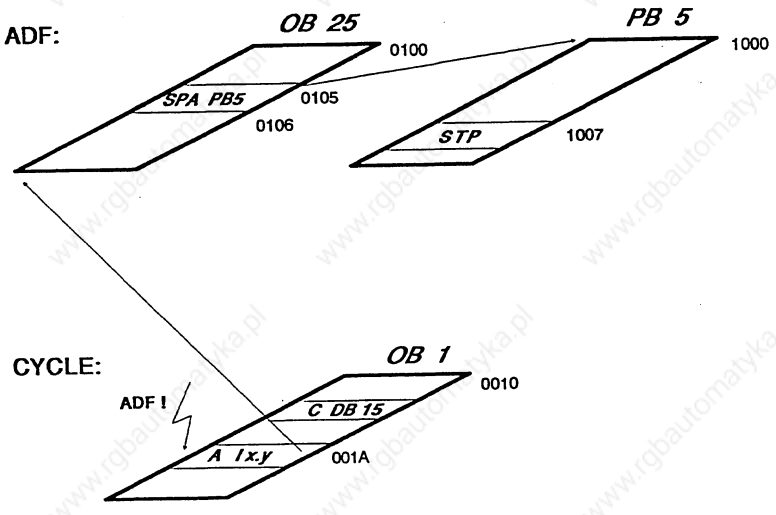
Prior to the final transition into the stop state, four different program levels have been interrupted. If you now output the ISTACK on your PG, a four-layer ISTACK will be displayed: on top, the ISTACK with depth 01 containing an identification of the program level (= ADF) which has been interrupted last. Now, you can move "down" to the ISTACK with depth 04 representing the "CYCLE" program level which has been interrupted first.



Example 2:

In the following example, the processor detects an addressing error in OB 1 when executing the instruction 'A I x.y'. This leads to the processing of OB 25. The processor goes into the stop state due to an STP command in PB 5.

A two-layer ISTACK is created since two program levels have been interrupted:



ISTACK

Depth: 01

BEF-REG: [STP]	SAC: 1007	DB-ADDR.:	BA-ADDR.:	0106
BST-STP: 3	PB-NO.: 5	DB-NO.: 16	OB-NO.:	25
	REL-SAC: 0007	DBL-REG:		
LEVEL: 001C	UAMK: 0300	UALW: 0000		

ACCU1:

RESULT BITS: .....

CAUSE OF INTERRUPTION: STP x

ISTACK

Depth 02

BEF-REG: [A I x.y]	SAC: 001A	DB-ADDR:	BA-ADDR:	0000
BST-STP: 1	OB-NO.: 1	DB-NO.: 16	-NO.:	
	REL-SAC: 000A	DBL-REG:		
LEVEL: 0004	UAMK: 0200	UALW: 0000		

ACCU1:

RESULT BITS: .....

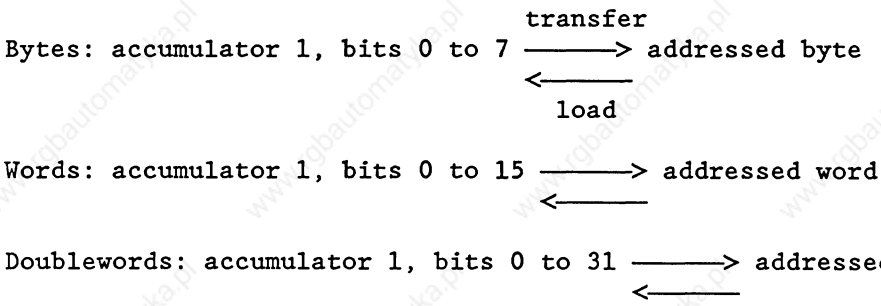
CAUSE OF INTERRUPTION: ADF x

4 STEP 5 command set with programming examples

4.1 General rules

The majority of STEP 5 operations use two registers (32 bits) as source for the operands and as destination for the results. These are accumulator 1 and accumulator 2.

Depending on the method of addressing (in bytes, words or double-words), load and transfer commands use the contents of accumulator 1 as follows:



Accumulator 1 is always the destination of the load operation and source of the transfer operation.

With byte or word load operations, the more significant bit positions which are not used are filled with zeros. Before this, the contents of accumulator 1 will be transferred to accumulator 2.

With loading/transfers in words from/to memory areas organized in bytes (PIO, PII, P/O peripherals, flags, S5 bus) byte n and byte n+1 are loaded/transferred).

With transfer instructions accumulator 1 and accumulator 2 remain unchanged. The auxiliary registers (accus 3 and 4) remain unchanged during all load and transfer instructions.

The execution of STEP 5 commands can be compared with that of the S5 150 S. Any differences will be pointed out.

- Numeric notation

Numbers in various notations can be used as operands for the STEP 5 commands, which logically operate on, change or compare the contents of accumulators 1 and 2. Depending on the operation to be executed, the contents of accumulators 1 or 2 will be interpreted as one of the following notations:

a) **Fixed point number**: is interpreted as a 16 bit binary number in two's complement notation (format change from 16 to 32 bit number, see section 5.1).

   Input with the PG: L KF Z, whereby $-32768 \leq Z \leq +32767$.

b) **BCD number**: with sign and 3 Figures; assignment in accu 1:

Bits	15 to 12	11 to 8	7 to 4	3 to 0
	sign	hundreds	tens	ones

The individual Figures are positive 4-bit binary numbers in two's complement notation.

Sign: 0000 if the number is positive
1111 if the number is negative

c) **Floating point number**: is interpreted as a 32-bit binary number with an 8-bit exponent and a 24-bit mantissa. With the +G, -G, xG and :G floating point operations, only a 16-bit mantissa is recognised in the S5 135 U; the 8 least significant bits are set to zero.

Example: (input of Z floating point numbers with the PG)

Z = 12.34567

L KG + 1234567 + 02
mantissa exponent (base 10)

$Z = +0.1234567 \times 10^{+2} = 12.34567$

Z = -0.005

L KG - 500000000 - 02
mantissa exponent (base 10)

$Z = -0.5 \times 10^{-2} = -0.005$

Note: the internal notation in the CPU need not correspond to the format in which the numbers are input during the creation of a program using the PG. (See operating instructions of the PG). The PG generates the notations shown above.

• Result bits

There are commands for processing information consisting of individual bits and commands for processing information consisting of words (8, 16 or 32 bits).

In both groups there are commands which set condition codes (flags) and commands which interpret these codes (see appendix: operation list, influencing condition codes). There are bit condition codes (bits 0 to 3) and word condition codes (bits 4 to 7) which correspond to the command groups. The condition code byte can be displayed by the PG and appears as follows:

	Word condition codes				Bit condition codes			
	CNC1	CNC0	OV	OS	OR	STA	RLO	$\overline{\text{ERAB}}$
Bit	7	6	5	4	3	2	1	0

Explanation of the bit condition codes:

ERAB First scanning cycle; a logic operation starts. At the end of a logic operation chain (memory operations) ERAB is set to 0. Commands which set ERAB to 0 (e.g. result assignment = Ax.x), limit the RLO (see appendix), i.e. the result of logic operation remains constant. It can, however, be interpreted (e.g. by RLO dependent commands), but not be further operated on. Only after the first logic operation statement (= first scanning cycle) will the result of logic operation be re-established.

RLO Result of logic operation; the result of bit-wide logic operations. Truth statement in the case of compare commands (see appendix: operation list, binary logic operations or compare operations).

STA Status; with bit commands specifies the logical status of the bit which has just been scanned or set. The status is updated with binary logic operations (except for A(, O(,), O) and with memory operations.

OR Or; informs the CPU that the following AND logic operations must be handled before an OR logic operation (AND before OR).

Explanation of the word condition codes:

OV Over; specifies whether during the arithmetic operation just completed, the permissible numerical range was exceeded.

OS Over latching; the over bit is latched. This is used to indicate whether at some stage during several arithmetic operations an error has occurred caused by overflow.

CNC1 and CNC0 are coded result bits, which are interpreted according to the following table.

Word result bits		Result of fixed point calculation	Digital logic operations	Comparison of contents of accu 1 with accu 2	Shifting: last bit shifted
CNC1	CNC0				
0	0	result = 0	= 0	accu 2 = accu 1	0
0	1	result < 0	-	accu 2 < accu 1	-
1	0	result > 0	≠ 0	accu 2 > accu 1	1

Jump operations are available for the immediate interpretation of the displays (see section 4.3).

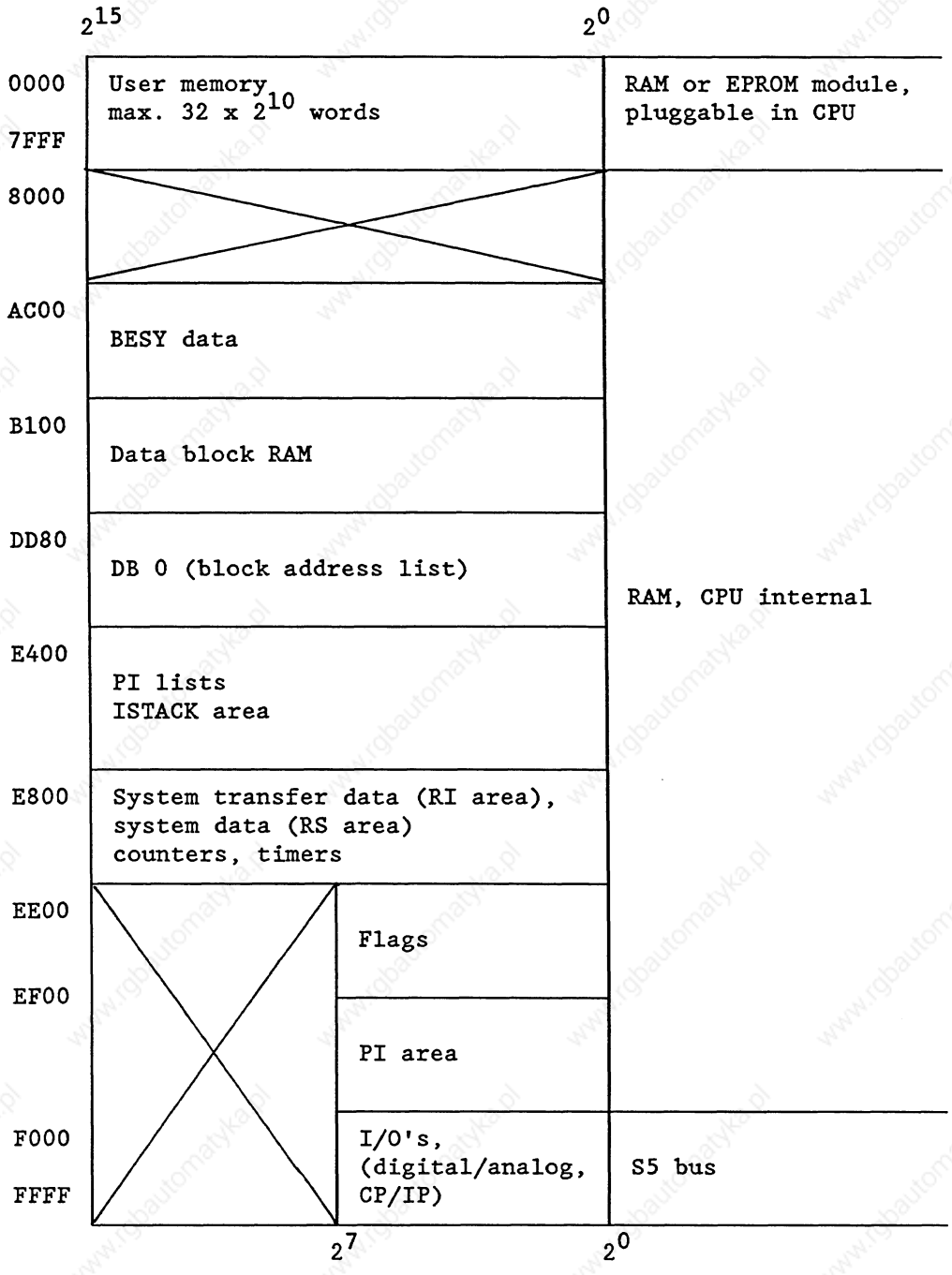


Fig. 17 Address area allocation

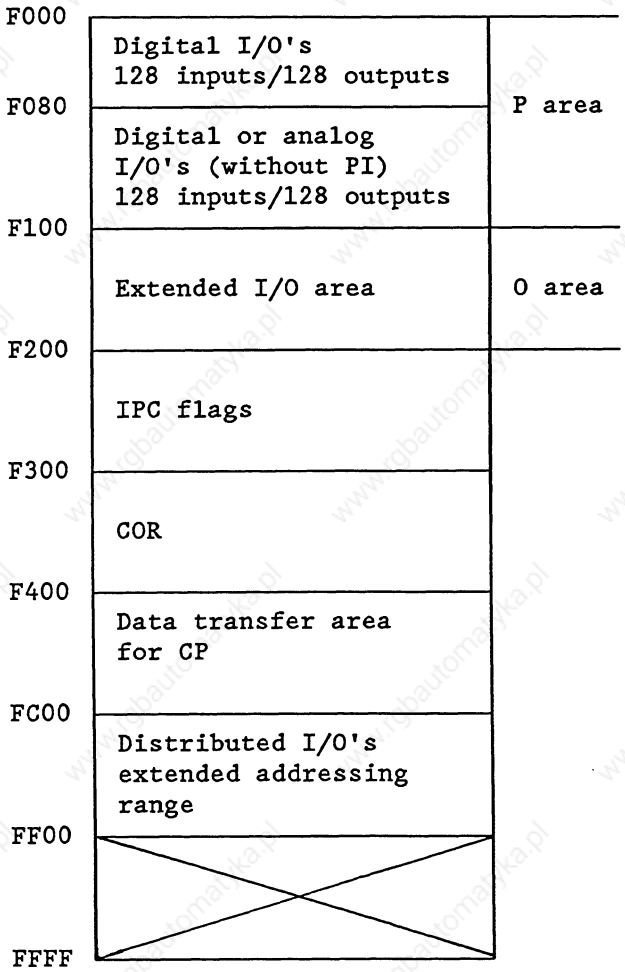
- Address areas for I/O/programming

Area (absolute address)		Is addressed with	Parameters
EF00 EF7F	PII (process image inputs)	L IB/T IB L IW/T IW L ID/T ID A I/AN I/O I/ON I S I/R I/= I	0 to 127 0 to 126 0 to 124 0.0 to 127.7
EF80 EFFF	PIO (process image outputs)	L QB/T QB L QW/T QW L QD/T QD A Q/AN Q/O Q/ON Q S Q/R Q/= Q	0 to 127 0 to 126 0 to 124 0.0 to 127.7
F000 F07F	Digital I/O's	L PB/T PB L PW/T PW	0 to 127 0 to 126
P-I/O's with process image			
F080 EF7F	Digital or analog I/O's	T PB/T PB T PW/T PW	128 to 255 128 to 254
P-I/O's without process image			
F100 F1FF	Extended I/O area	L OB/T OB L OW/T OW	0 to 255 0 to 254
Q-I/O's			

Fig. 17 Address area allocation in the S5 135 U

B8576364/1

- I/O areas on the S5 bus



4.2 Basic operation set

- Binary logic operations

Operation	Parameter	Function
) A (O (O		Close brackets ANDing expressions in brackets ORing expressions in brackets ORing AND functions
A O		AND function OR function
I	0.0 to 127.7	with scanning of an input for signal status "1"
Q	0.0 to 127.7	with scanning of an output for signal status "1"
F	0.0 to 255.7	with scanning of a flag for signal status "1"
D	0.0 to 255.15	with scanning of a data word for signal status "1"
N I	0.0 to 127.7	with scanning of an input for signal status "0"
N Q	0.0 to 127.7	with scanning of an output for signal status "0"
N F	0.0 to 255.7	with scanning of a flag for signal status "0"
N D	0.0 to 255.15	with scanning of a data word for signal status "0"
T	0 to 127	with scanning of a timer for signal status "1"
N T	0 to 127	with scanning of a timer for signal status "0"
C	0 to 127	with scanning of a counter for signal status "1"
N C	0 to 127	with scanning of a counter for signal status "0"

Binary logic operations generate the result of logic operation (RLO) as their result.

At the start of a logic operation sequence the results from the first logic operation (first scan) are only dependent on the status of the scanned signal and whether or not it is negated (N = negation); they are not, however, dependent on the type of logic operation (O = OR, A = AND).

During a logic operation sequence, the RLO is formed from the type of logic operation, the previous RLO and the status of the scanned signal. A logic operation sequence is completed by an RLO limiting ($\overline{\text{ERAB}}$ = 0) command (e.g. memory operations).

The RLO remains unchanged until the next "first scan". It can be interpreted, but cannot be further operated on.

Program	Status	RLO	ERAB	
:				
= Q 0.0	0	0	0	← RLO limited
A I1.0	1	1	1	← first scan
A I1.1	1	1	1	
A I1.2	0	0	1	
= Q 0.1	0	0	0	← RLO limited, end of logic operation sequence

- Memory operations

Operation	Parameters	Function
S		Set
R		Reset
=		Assign
I	0.0 to 127.7	an input in the PII
Q	0.0 to 127.7	an output in the PIO
F	0.0 to 255.7	a flag
D	0.0 to 255.15	a data word bit

- Loading, transfer and compare functions

Operation	Parameters	Function
L		Load
T		Transfer
I B	0 to 127	an input byte from/to the PII
I W	0 to 126	an input word from/to the PII
I D	0 to 124	an input doubleword from/to the PII
Q B	0 to 127	an output byte from/to the PIO
Q W	0 to 126	an output word from/to the PIO
Q D	0 to 124	an output doubleword from/to the PIO
F B	0 to 255.	a flag byte
F W	0 to 254	a flag word
F D	0 to 252	a flag doubleword
D R	0 to 255	data (right byte) from DB, DX
D L	0 to 255	data (left byte) from DB, DX
D W	0 to 255	a data word
D D	0 to 254	a data doubleword
P B	0 to 127	a peripheral byte of the digital inputs or outputs (P area)
P B	128 to 255	a peripheral byte of the analog or digital inputs or outputs (P area)
O B	0 to 255	a byte of the extended I/O area (O area)

• Loading, transfer and compare functions (continued)

Operation	Parameters	Function
L T		Load Transfer
P W	0 to 126	a peripheral word of the digital inputs or outputs (P area)
P W	128 to 254	a peripheral word of the analog or digital inputs or outputs (P area)
O W	0 to 254	a word of the extended peripherals (O area)
L		Load
K M	16 bit pattern	a constant as bit pattern
K H	0 to FFFF	a constant in the hexadecimal code
K F	-32 768 to +32 767	a constant as a fixed point number
K Y	0 to 255 for each byte	a constant, 2 bytes
K B	0 to 255	a constant, 1 byte
K S	2 alphanumeric characters	a constant, 2 ASCII characters
K T	0.0 to 999.3	a time value (constant)
K C	0 to 999	a count value (constant)
K G	[1]	a constant as a floating point number (32 bit)
T	0 to 127	a time value
C	0 to 127	a count value
LD T	0 to 127	BCD loading of a time value
LD C	0 to 127	BCD loading of a count value
! =		compare for equal to
> <		compare for not equal to
>		compare for greater than
> =		compare for greater than or equal to
<		compare for less than
< =		compare for less than or equal to
F		two fixed point numbers (16 bit)
D		two fixed point numbers (32 bit)
G		two floating point numbers (32 bit)

The loading and transfer operations do not influence the condition codes. The compare commands generate the RLO and the CNC1 and CNC0 word condition codes as the result. The contents of accumulators 1 and 2 are always compared (see program examples and operations list).

For loading and transfer operations the instructions in section 4.1 should be noted. The I/O's can be addressed directly by loading and transfer operations - with L/T PB, PW, OB, OW or by means of

[1] $\pm 0.1469368 \times 10^{-38}$ to $\pm 0.1701412 \times 10^{39}$

Process image - with L/T IB, IW, ID, QB, QW, QD and with logic operations. With T PB 0 to 127 and T PW 0 to 126 the PIO will be maintained at the same time. (PII/PIO = process image of the inputs/outputs for 128 input/output bytes of the P I/O's with byte addresses from 0 to 127).

The process image represents a memory area, the contents of which are only output to the peripherals (PIO) or read in by the peripherals (PII) once per user program cycle (see Fig. 14). This avoids frequent changing of the logic status of a bit within a program cycle, which leads to "chattering" of the corresponding peripheral output.

The O area can only be addressed via the 300 and 301 interface modules, so that I/O modules with addresses in the O area can only be plugged into expansion units. For the whole O area and P area with relative byte addresses from 128 to 255, there is no process image.

With word loading and word transfer operations to address areas organized in bytes (PII, PIO, flags, S5 bus), byte n and byte n + 1 will be loaded/transferred; with doubleword operations byte n to byte n + 3 will be loaded/transferred.

Example

L IW 5 bytes 5 and 6 of the PII will be loaded into accu 1.

L FD 10 flag bytes 10 to 13 will be loaded.

- Timer and counter operations

In order to load a timer using a start command, or a counter using a set command, the value must be loaded into accumulator 1 beforehand.

The following loading operations are recommended:

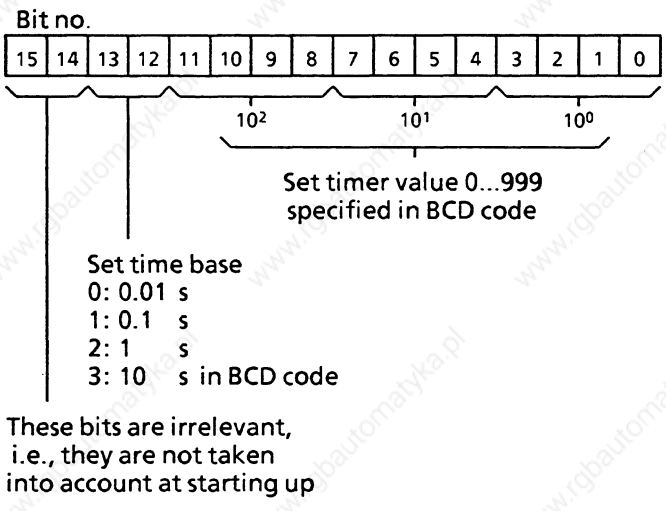
for timers: L KT, L IW, L QW, L FW, L DW
for counters: L KC, L IW, L QW, L FW, L DW.

Operation	Parameters	Function
S P T	0 to 127	starting a timer as a pulse
S E T	0 to 127	starting a timer as an extended pulse
S R T	0 to 127	starting a timer as an "ON" delay
S S T	0 to 127	starting a timer as a latching "ON" delay
S F T	0 to 127	starting a timer as an "OFF" delay
R T	0 to 127	resetting a timer
S C	0 to 127	setting a counter
R C	0 to 127	resetting a counter
C U C	0 to 127	incrementing a counter
C D C	0 to 127	decrementing a counter

When the SP, SR, SE, SS, SF and S timer or counter operations are carried out, the value in accumulator 1 will be fetched into the timer or counter location (corresponds to the transfer command) and the corresponding operation will be started.

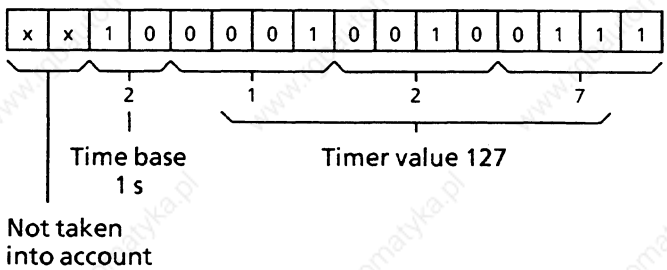
If the time value or count value is loaded using IW, QW, FW or DW, the corresponding word must have the following structure:

For the time value

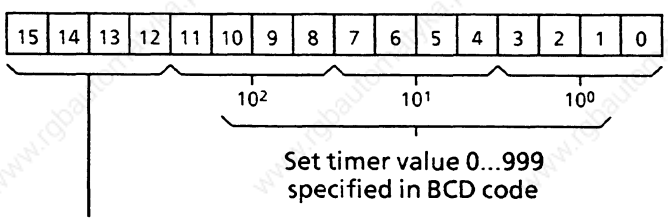


Example: setting a time of 127 s

Bit assignment:

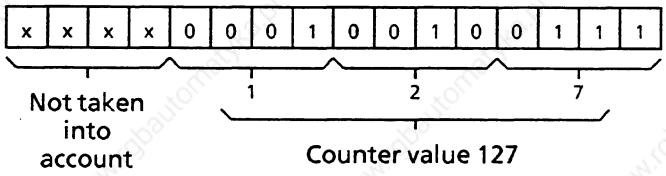


For the count value



Example: setting a count value of 127.

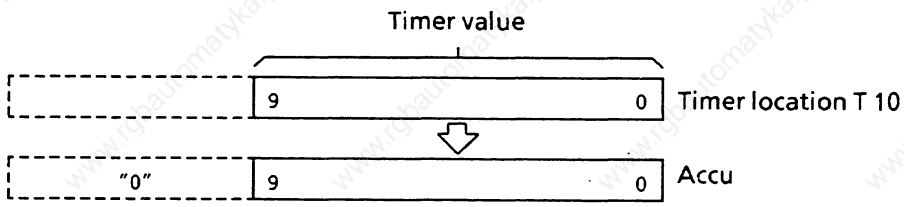
Bit assignment:



The time or count value is stored in the timer or counter location and is binary coded. In order to scan the timer or the counter, the value in the timer or counter location can be loaded into accumulator 1 directly or in BCD.

Example:

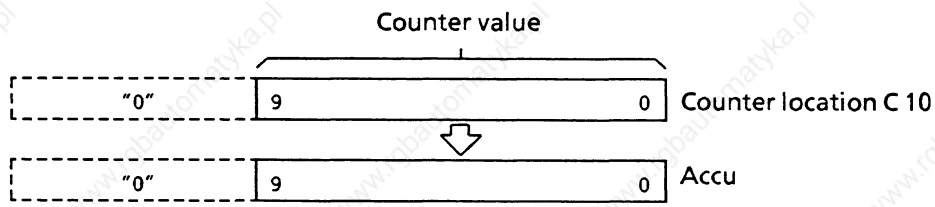
Direct loading of time values



L T 10 Directly loading the binary value of the T 10 timer into the accumulator

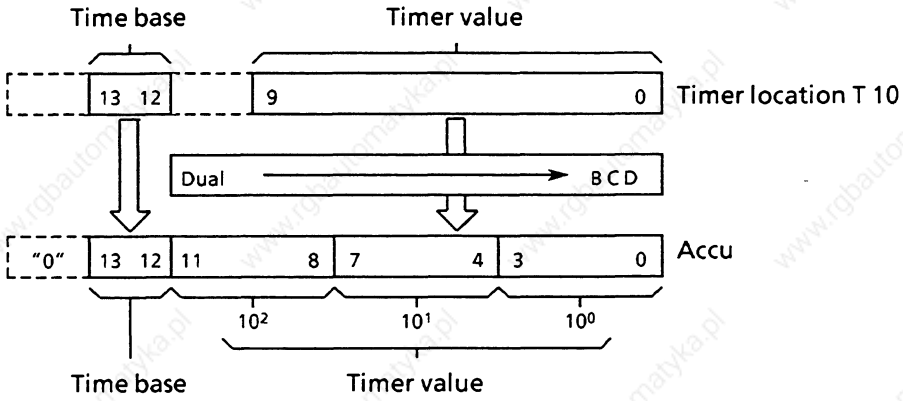
The time base is not loaded at the same time.

Direct loading of count values:



L C 10 Loading the count value of the C 10 counter directly into the accumulator

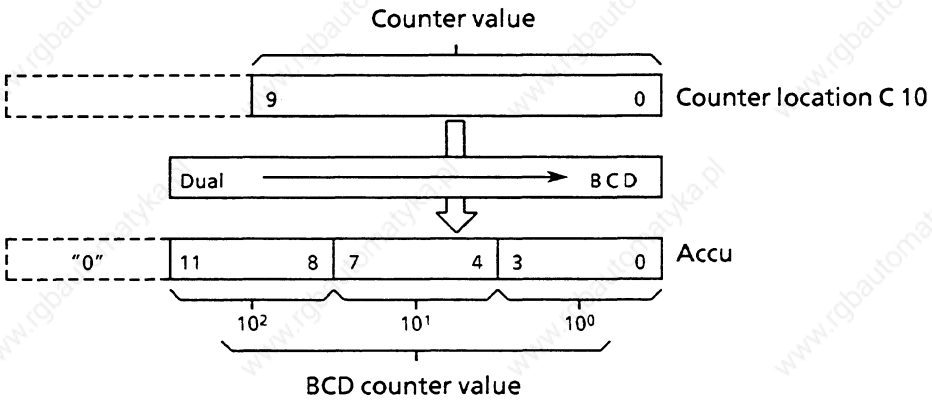
Coded loading of time values:



LD T 10 Coded loading of the time value and the time base of the T 10 timer into the accumulator

The time base is also loaded.

Coded loading of count values:



LD C 10 Coded loading of the count value of the C 10 counter into the accumulator

With coded loading, status bits 14 and 15 of the timer locations or 12 to 15 of the counter locations are not loaded. In their place, there is a 0 in accumulator 1. The value now in the accumulator can be processed further.

- Arithmetic operations

Operation	Parameters	Function
+ F		addition of 2 fixed point numbers (16 bits)
- F		subtraction of 2 fixed point numbers (16 bits)
x F		multiplication of 2 fixed point numbers (16 bits)
: F		division of 2 fixed point numbers
+ G		addition of 2 floating point numbers
- G		substraction of 2 floating point numbers
x G		multiplication of 2 floating point numbers
: G		division of 2 floating point numbers

The arithmetic operations refer to the contents of accumulators 1 and 2 (see operations list). The result is then available in accumulator 1. The arithmetic registers are changed by an arithmetic operation as follows:

<accu 1>: = result
<accu 2>: = <accu 3>
<accu 3>: = <accu 4>
<accu 4>: = <accu 4>

The previous contents of accumulator 2 are lost.

- Block calls

Operation	Parameters	Function
J U		jump absolute
J C		jump conditional (dependent on the RLO)
O B	1 to 39	to an organization block
O B	40 to 255	to a system program special function
P B	0 to 255	to a program block
F B	0 to 255	to a function block
S B	0 to 255	to a sequence block
B C F X	0 to 255	unconditional jump to an extended function block
B C C F X	0 to 255	conditional jump to an extended function block
C D B	3 to 255	DB data block call
C X D X	1 to 255	DX extended data block call
B E		block end
B E C		conditional block end (dependent on the RLO)
B E U		unconditional block end

- No operation

Operation	Parameters	Function
N O P	0	no operation
N O P	1	no operation
B L D	0 to 255	display construction statment for the PG (is treated as a no operation by the CPU)

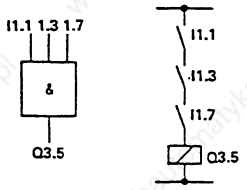
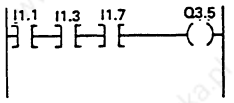
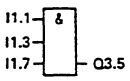
- Stop statement

Operation	Parameters	Function
S T P		CPU goes into stop status

Programming examples for logic, memory, timer, counter and compare functions

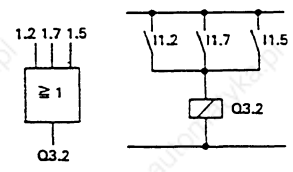
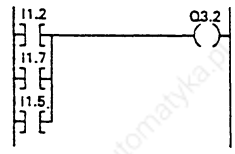
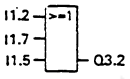
- Logic functions

AND operation

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.1 A I 1.3 A I 1.7 = Q3.5		

A "1" signal appears at output Q 3.5 when all the inputs have "1" signals simultaneously
A "0" signal appears at output Q 3.5 if at least one of the inputs has a "0" signal.
There are no restrictions imposed on the number of scans or on the programming sequence.

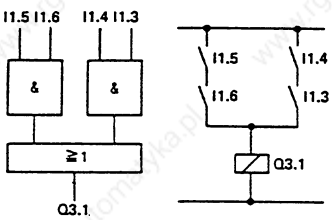
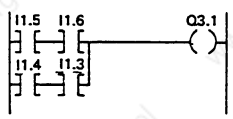
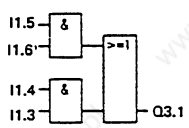
OR operation

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	O I 1.2 O I 1.7 O I 1.5 = Q3.2		

A "1" signal appears at output Q 3.2 if at least one of the inputs has a "1" signal.
A "0" signal appears at output Q 3.2 if all of the inputs have a "0" signal.
There are no restrictions imposed on the number of scans or on the programming sequence.

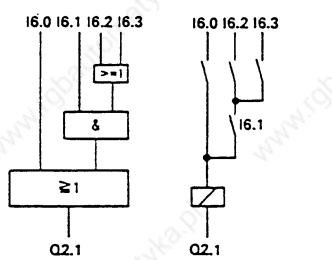
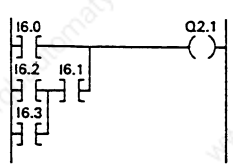
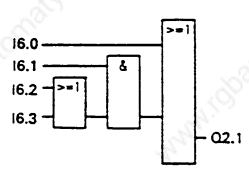
- Logic functions (continued)

AND before OR operation

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.5 A I 1.6 O A I 1.4 A I 1.3 = Q3.1		

A "1" signal appears at output Q 3.1 when the output of at least one of the AND gates is "1".
A "0" signal appears at output Q 3.1 when neither of the AND gates has a "1" at its output.

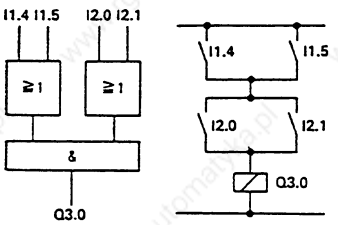
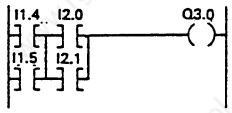
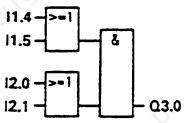
OR before AND operation

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	O I 6.0 O A I 6.1 A(O I 6.2 O I 6.3) = Q2.1		

A "1" signal appears at output Q 2.1 when input I 6.0 or input I 6.1 and one of the inputs I 6.2 or I 6.3 have a "1" signal.
A "0" signal appears at output Q 2.1 when input I 6.0 has a "0" signal and the AND gate has a "0" at its output

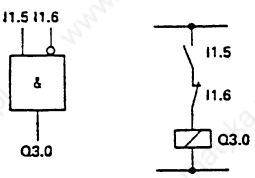
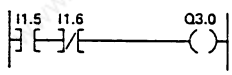
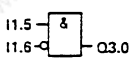
- Logic functions (continued)

OR before AND operation

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A(O I 1.4 O I 1.5) A(O I 2.0 O I 2.1) = Q3.0		

A "1" signal appears at output Q 3.0 when both OR gates have "1" signal at their outputs.
A "0" signal appears at output Q 3.0 when at least one of the OR gates has a "0" signal at its output.

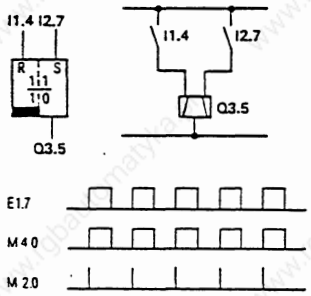
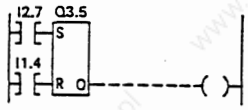
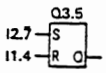
Scanning for "0" signal status

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.5 AN I 1.6 = Q3.0		

A "1" signal appears at output Q 3.0 only when input I 1.5 has a "1" signal (normally open contact actuated) and input I 1.6 has a "0" signal (normally closed contact actuated).

- Memory functions

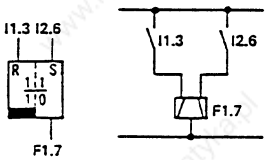
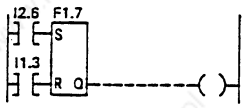
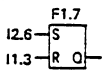
RS flip-flops for latching signal output

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 2.7 S Q3.5 A I 1.4 R Q3.5		

A "1" signal at input I 2.7 sets the flip-flop, (signal "1" at output Q 3.5).
If the signal at input I 2.7 changes to "0", the flip-flop status remains unchanged, i.e. the signal is latched.
A "1" signal at input I 1.4 resets the flip-flop, (signal "0" at output Q 3.5).
If the signal at input I 1.4 changes to "0", the flip-flop status remains unchanged.
If the set signal (input I 2.7) and the reset signal (input I 1.4) appear simultaneously, the scan operation programmed last (in this case A I 1.4) is effective during the processing of the remaining program (reset has priority).

• Memory functions (continued)

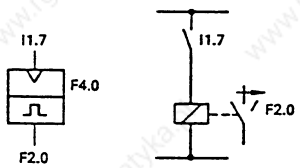
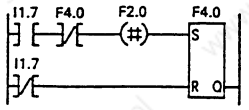
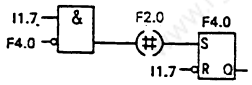
RS flip-flop with flags

Original	STEP 5 operation		
	Statement list	Ladder diagram	Control system flowchart
	A I 2.6 S F 1.7 A I 1.3 R F 1.7		

A "1" signal at input I 2.6 sets the flip-flop.
If the signal at input I 2.6 changes to "0", the flip-flop status remains unchanged, i.e. the signal is latched.
A "1" signal at input I 1.3 resets the flip-flop.
If the signal at input I 1.3 changes to "0", the flip-flop status remains unchanged.
If the set signal (input I 2.6) and the reset signal (input I 1.3) appear simultaneously, the scan operation programmed last (in this case A I 1.3) is effective during the processing of the remaining program (reset has priority).

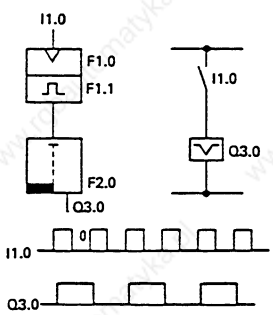
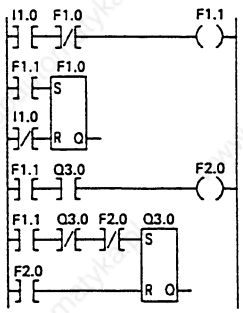
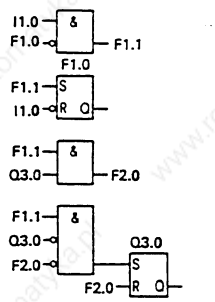
- Memory functions (continued)

Implementation of an impulse relay

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.7 AN F 4.0 = F 2.0 A F 2.0 S F 4.0 AN I 1.7 R F 4.0		

The AND logic condition (A I 1.7 and AN F 4.0) is fulfilled at each positive-going edge of the signal at input I 1.7 and flags F 4.0 ("pulse edge flag") and F 2.0 (pulse flag) are set if the RLO = "1".
The AND logic condition A I 1.7 and AN F 4.0 is no longer fulfilled during the next processing cycle since flag F 4.0 has been set.
Flag F 2.0 is reset, i.e. it is "1" during a single program run.

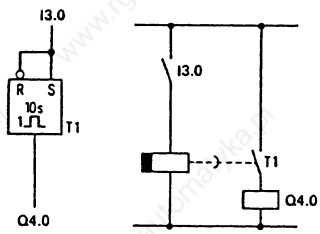
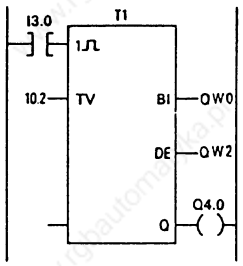
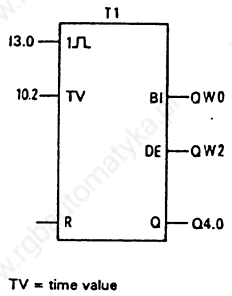
Binary scaler

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 1.0 AN F 1.0 = F 1.1 A F 1.1 S F 1.0 AN I 1.0 R F 1.0 A F 1.1 A Q 3.0 = F 2.0 A F 1.1 AN Q 3.0 AN F 2.0 S Q 3.0 A F 2.0 R Q 3.0		

Output Q 3.0 of the binary scaler changes its state at each positive-going edge of the signal at input I 1.0, i.e. when input I 1.0 changes from "0" to "1". Consequently, half the input frequency appears at the binary scaler output.

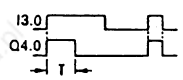
- Timer functions

Pulse

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 3.0 L KT 10.2 SI T 1 A T 1 = Q 4.0		

The timer is started during the first processing cycle if the result of the logic operation is "1". The timer remains unaffected during subsequent processing if this results in a "1" signal.
The timer is set to "0" (deleted) if the result of the logic operation is "0".
The A T and O T scans result in a "1" signal as long as the timer is running.

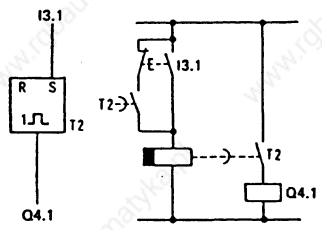
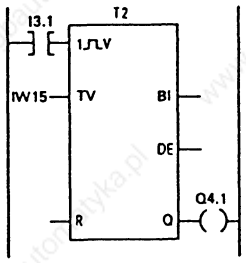
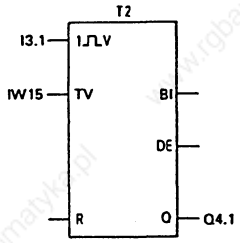
KT 10.2:
The timer is loaded with the specified value (10).
The number to the right of the decimal point indicates the time base:
0 = 0.01 s 2 = 1 s
1 = 0.1 s 3 = 10 s



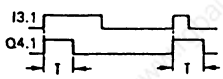
DI and DE are digital outputs of the timer location. The time value is binary at output BI and BCD with time base at output DE.

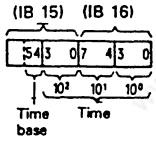
- Timer functions (continued)

Extended pulse

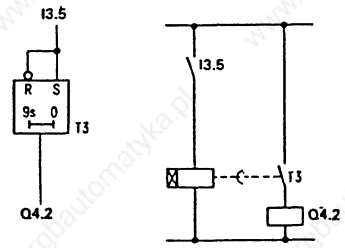
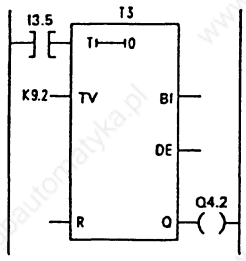
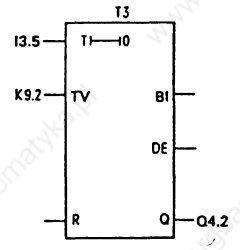
Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 3.1 L IW15 SE T 2 A T 2 = Q 4.1		

The timer is started during the first processing cycle if the result of the logic operation is "1".
The timer remains unaffected if the result of the logic operation is "0".
The A T or O T scans result in a "1" signal as long as the timer is running.
IW 15:
Setting the time value with the BCD value of the operands I, Q, F or D
(input word value 15 in the example).

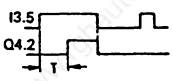




"On" delay

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 3.5 L KT9.2 SR T 3 A T 3 = Q 4.2		

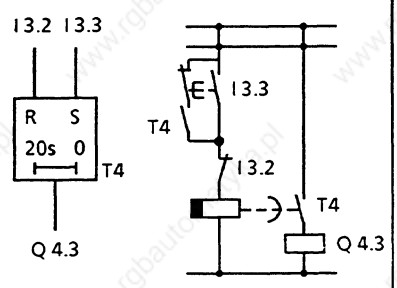
The timer is started during the first processing cycle if the result of the logic operation is "1". The timer remains unaffected during subsequent processing if the result of the logic operation remains "1".
The timer is set to "0" (deleted) if the result of the logic operation is "0".
The A T or O T scans result in a "1" signal when the time has elapsed and the result of the logic operation is still present at the input.
KT 9.2:
The timer is loaded with the specified value (9). The number to the right of the point indicates the time base:
0 = 0.01 s 2 = 1 s
1 = 0.1 s 3 = 10 s



- Timer functions (continued)

Latching "On" delay

Original

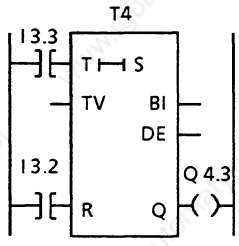


STEP 5 representation

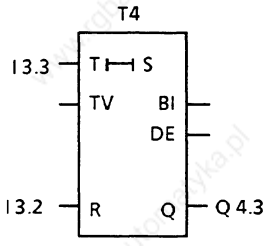
Statement list

```
A  I  3.3
L  KT 10.2
SS T  4
A  I  3.2
R  T  4
A  T  4
=  Q  4.3
```

Ladder diagram



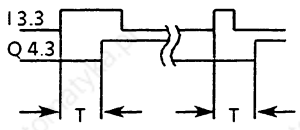
Control system flowchart



The timer is started during the first processing cycle if the result of the logic operation is "1"

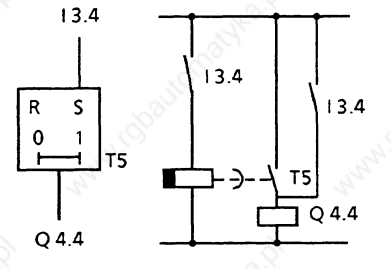
The timer remains unaffected if the result of the logic operation is "0".

The AT or OT scans result in a "1" signal when the time has elapsed. The signal status only changes to "0" when the timer is reset by the RT function.



"Off" delay

Original

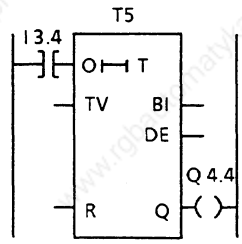


STEP 5 representation

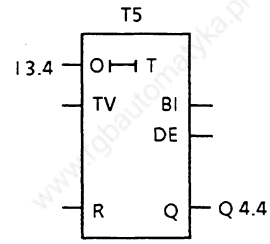
Statement list

```
A  I  3.4
L  KT 10.2
SQT 5
A  T  S
=  Q  4.4
```

Ladder diagram



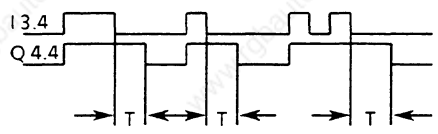
Control system flowchart



The timer is started when the result of the logic operation at the start input changes from "1" to "0". It runs for the time programmed.

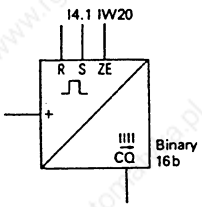
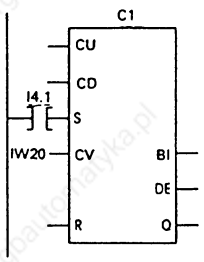
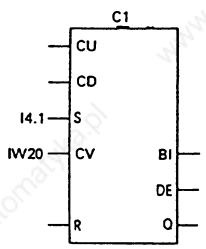
The timer is set to zero (reset) if the result of the logic operation is "1"

The AT or OT scans result in a "1" signal if the timer is running **or** the result of the logic operation is still present at the input.



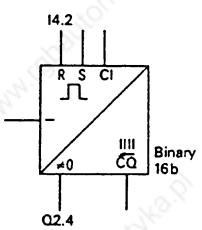
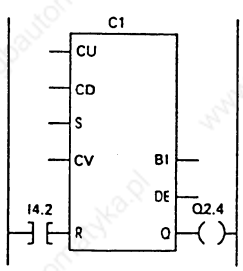
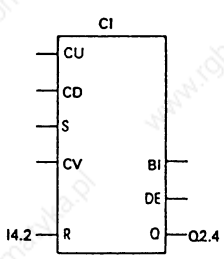
- Counter functions

Set counter

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 4.1 L IW20 S C 1		

The counter is set during the first processing cycle if the result of the logic operation is "1". The counter remains unchanged during subsequent processing (no matter whether the result of the logic operation is "1" or "0"). The counter is set again (pulse edge evaluation) at the next processing cycle if the result of the logic operation is "1".
The flag necessary for pulse edge evaluation of the set input is included in the counter word.
BI and DE are digital outputs of the counter location. The count values are binary coded at output BI and BCD at output DE.

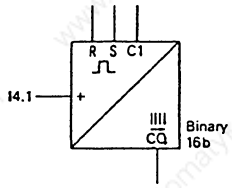
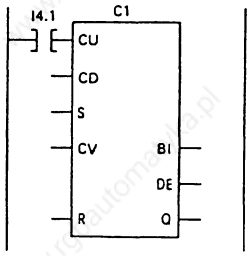
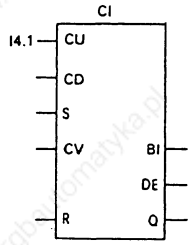
Reset counter

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 4.2 R C 1 A C 1 = Q 2.4		

The counter is set to zero (reset) when the result of the logic operation is "1".
The counter remains unchanged even if the result of the logic operation becomes "0".

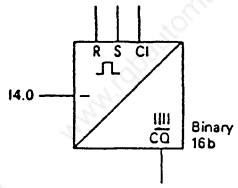
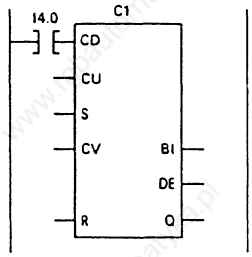
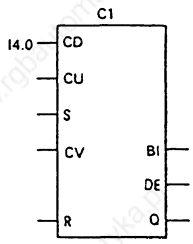
- Counter functions (continued)

Counting up

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 4.1 CU C 1		

The value of the addressed counter is incremented by 1 up to a maximum of 999. The CU function is effective only on a positive-going pulse edge (from "0" to "1") of the logic operation programmed before CU. The flags necessary for pulse edge evaluation of the counter inputs are included in the counter word.
A counter with two different inputs can be used as an up/down counter by means of the two separate pulse-edge flags for CU and CD.

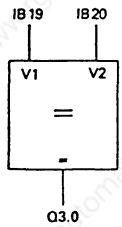
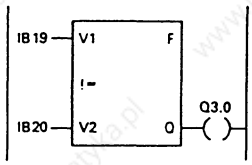
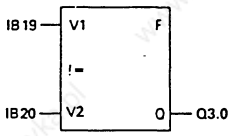
Counting down

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	A I 4.0 CD C 1		

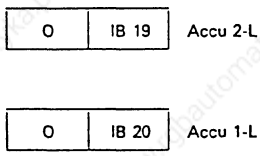
The value of the addressed counter is decremented by 1 to a minimum 0. The CD function is only effective with a positive-going edge (from "0" to "1") of the logic operation programmed before CD.
The flags necessary for pulse edge evaluation of the counter inputs are included in the counter word.
A counter with two different inputs can be used as an up/down counter by means of the two separate pulse-edge flags for CU and CD.

- Compare functions

Comparing for equal to

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	L IB 19 L IB 20 ! = F = Q 3.0		

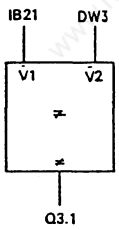
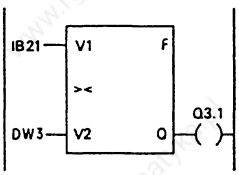
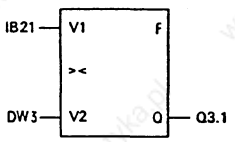
The first operand specified is compared with the following operand according to the comparison function.
The comparison produces a binary logic operation result:
RLO = "1": the condition is fulfilled, if accu 1-L = accu 2-L
RLO = "0": the condition is not fulfilled, if accu 1-L ≠ accu 2-L



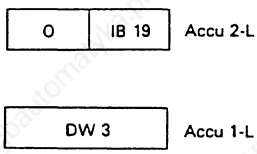
The condition codes CNC1 and CNC0 are set as explained in 4.1
Accu 2-H and accu 1-H remain unaffected during the 16-bit fixed point comparision.
During fixed point comparison (! = F) and floating point comparison (! = G) the total contents of accu 1 and accu 2 (32-bit) are compared with each other.
During the comparison the numberical representation of the operands is taken into account, i.e. the contents of accu 1-L and accu 2-L are interpreted as a fixed point number.

- Compare functions (continued)

Comparing for not equal to

Original	STEP 5 representation		
	Statement list	Ladder diagram	Control system flowchart
	L IB21 L DW3 ><F = Q3.1		

The first operand specified is compared with the following operand according to to the comparison function.
The comparison produces a binary logic operation result.
RLO = "1" the condition is fulfilled, if accu 1-L ≠ accu 2-L
RLO = "0" the condition is not fulfilled, if accu 1-L = accu 2-L



The condition codes CNC1 and CNC0 are set according to the table in section 4.1.
Accu 2-H and accu 1-H remain unaffected during the 16-bit fixed point comparison.
During the 32-bit fixed point comparison and the floating point comparison accu 2-H and accu 1-H are involved.
This also applies to comparing for greater than, greater than or equal to, less than and less than or equal to (see operations list).
When comparing, the numerical representation of the operands is taken into account, i.e. the contents of accu 1-L and accu 2-L are interpreted as a fixed point number.

4.3 Supplementary operation set

In contrast to the other blocks, function blocks can be programmed with an extended operation set. The entire operation set for function blocks consists of the basic operations and the supplementary operations.

Together with the basic functions and the supplementary functions, the system functions complete the operation set of the STEP 5 programming language.

With the system functions it is possible to intervene in the running of the system program; the memory can be overwritten at any point, and the contents of the working register of the central processor can be changed. Therefore, the system functions should only be used (if at all) with the utmost caution.

The system functions are clearly indicated in the following lists:

Function block operations are only represented in STL. The programs of the function blocks cannot therefore be programmed in graphic form (LAD or CSF).

The following description shows the supplementary operations and system functions which can only be used with function blocks. The possible combinations of substitution operations with actual operands are also given.

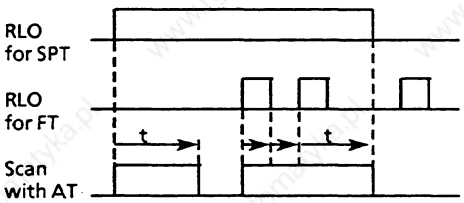
- Binary logic operations

Operations	Description
A = ☐	AND operation, scanning a formal operand for signal status "1".
AN = ☐	AND operation, scanning a formal operand for signal status "0"
O = ☐	OR operation, scanning a formal operand for signal status "1"
ON = ☐	OR operation, scanning a formal operand for signal status "0"
↑	Assign formal operand
	Inputs, outputs, data and flags addressed in binary code (parameter class I, Q; parameter type BI) and also timers and counters (parameter class T, C) are permitted as actual operands.

- Memory functions

Operation	Description
S = ☐	Set (binary) formal operand.
RB = ☐	Reset (binary) formal operand.
= = ☐	Assign result of logic operation to formal operand.
↑	Assign formal operand.
	Inputs, outputs, data and flags addressed in binary code (parameter class I, Q, parameter type BI) are permitted as actual operands.

• Timer and counter functions

Operation	Description
FR T 0 to 127	Enabling a timer for restart The operation is only carried out on the leading edge of the result of the logic operation. The timer is restarted if the RLO is "1" at the time of the start operation. 
FR C 0 to 127	Enabling a counter The operation is only carried out on the leading edge of the result of the logic operation. The counter is set (counting up or down) if the result of the logic operation is "1" at the corresponding operation.
FR = ☐	Enabling a formal operand for a restart (for description see FRT or FRC depending on formal operand; parameter class: T, C).
RD = ☐	Resetting (digital) a formal operand (parameter class: T, C).
SP = ☐	Starting a timer, specified as a formal operand, as a pulse with the value stored in the accumulator (parameter class: T).
SR = ☐	Starting a timer, specified as a formal operand, as an on-delay with the value stored in the accumulator (parameter class: T).
SEC = ☐	Starting a timer, specified as a formal operand, as an extended pulse with the value stored in the accumulator or setting a counter specified as a formal operand for the count value stored in accu 1 (parameter class: T, C)
SSU = ☐	Starting a timer, specified as a formal operand, as a latching on-delay with the value stored in the accu or incrementing a counter specified as a formal operand (parameter class: T, C).
SFD = ☐	Starting a timer, specified as a formal operand, as an off-delay with the value stored in the accu or decrementing a counter specified as a formal operand (parameter class: T, C).
	Enter formal operand
	Timers and counters are permitted as actual operand. Exceptions: SP and SR (only timers). The timer or counter value can be assigned as with basic operations: or as a formal operand it can be assigned as follows: Set the timer or counter value with the BCD value, of the IW, QW, FW, DW operands specified as formal operands (parameter class: I, parameter type: W) or as a constant (parameter class: D, parameter type: KT, CC).

Examples

Function block call	Program in function block	Executed program
: JU FB203 NAME : EXAMPLE ANNE : I 10.3 BERT : T 17 FRED : Q 18.4	:A =ANNE :L KT 010.2 :SSU =BERT :A =BERT := =FRED	:A I 10.3 :L KT 010.2 :SS T 17 :A T 17 := Q 18.4
: JU FB204 NAME : EXAMPLE RUTH : I 10.5 PETE : I 10.6 MAUD : I 10.7 DORA : C 15 EMMA : F 58.3	:A =RUTH :SSU =DORA :A =PETE :SFD =DORA :A =MAUD :L KC100 :SEC =DORA :AN =DORA := =EMMA	:A I 10.5 :CU C 15 :A I 10.6 :CD C 15 :A I 10.7 :L KC 100 :S C 15 :AN C 15 := F 58.3
: JU FB205 NAME : EXAMPLE BILL : I 10.4 CARL : T 18 EGON : IW20 DAVE : F 100.7	:A =BILL :L =EGON :SEC =CARL :A =CARL := =DAVE	:A I 10.4 :L IW 20 :SF T 18 :A T 18 := F 100.7

- Loading and transfer functions

Operation	Description
L = ☐	Loading of a formal operand The value of the operand specified as a formal operand is loaded into the accumulator (parameter class: I, T, C, Q; parameter type: BY, W, D)
LD = ☐	Coded loading of a formal operand. The value of the timer or counter location specified as a formal operand is loaded in BCD into the accumulator (parameters: T, C).
LW = ☐	Loading the bit pattern of a formal operand. The bit pattern of the formal operand is loaded into the accumulator (parameter class: D; parameter type: KF, KH, KM, KY, KS, KT, KC).
xxxx = ☐	Loading the bit pattern of a formal operand. The bit pattern of the formal operand is loaded into the accumulator (parameter class: D; parameter type: KG).
T = ☐ Enter | formal operand	Transferring to a formal operand. The accumulator contents are transferred to the operand specified as a formal operand (parameter class: I, Q; parameter type: BY, W, D).

Operands corresponding to the basic operations are permitted as actual operands. For LW, data is permitted in the form of a binary (KM) or hexadecimal (KH) pattern, 2 numbers in bytes (KY), characters (KC), fixed point number (KF), time values (KT) and count values (KC). For LD, a floating point number is permitted as data.

Operation	Parameters	Description
L RI	0 to 255	Loading a word into accu 1 from the "interface data" area
L RS	0 to 255	Loading a word into accu 1 from the "system data" area
T RI [1])	0 to 255	Transferring accu 1 to a word from the "interface data" area
T RS [1])	0 to 255	Transferring accu 1 to a word from the "system data" area
LIR [1])	0 to 15	Load the register (indirect): with the contents of the memory word [2]) addressed by accu 1
TIR [1])	0 to 15	Transfer the register contents (indirect): into the memory word [2]) addressed by the contents of accu 1
TNB [1])	0 to 255	Block transfer in bytes: source in accu 2, destination in accu 1 [3])
TNW [1])	0 to 255	Block transfer in words: source in accu 2, destination in accu 1 [3])

[1]) system function

[2]) register for LIR and TIR (register width = 16 bits)

Register no.	Designation of the register
0	Accu 1-H = more significant word (16 bit) of accu 1
1	Accu 1-L = less significant word (16 bit) of accu 1
2	Accu 2-H
3	Accu 2-L
4	-
5	BSP = block stack pointer
6	DBS = start address of the data block currently selected
7, 8	-
9	Accu 3-H
10	Accu 3-L
11	Accu 4-H
12	Accu 4-L
13, 14	-
15	SAC = step address counter

Auxiliary register

a) Registers 4, 7, 8, 13 and 14 are not present. LIR/TIR on these register numbers will be handled as a no operation (NOP).
b) Access to 8-bit memory (for memory word addresses ≥ EE00H):
TIR: the high byte of the register is lost.
LIR: FFH is written into the high byte of the register.

[3]) The parameter with TNW/TNB specifies the length of the area to be transferred. The source area end address must be loaded in accumulator 2 beforehand, the destination area end address in accumulator 1. The source area and the destination area must each be **completely** in one memory area. The following memory areas are distinguished by their area limits:

	Addresses (hexadecimal)
16-bit user memory	0000 to 7FFF or to 3FFF for 16 KW module
16-bit CPU RAM	AC00 to EDFF
8-bit CPU RAM	EE00 to EFFF
8-bit I/O area	F000 to FFFF

See also address area allocation (Fig. 17).

- Arithmetic operations

Operation	Description
ENT	Enter data in the arithmetic memory The ENT command results in the loading of accus 3 and 4 which are also used in arithmetic operations: accu 4: = accu 3 accu 1: = accu 1 accu 3: = accu 2 accu 2: = accu 2

The former contents of accu 4 are lost.

Example

The following fraction is to be calculated: (30 + 3 x 4)/6 = 7

	Accu 1	Accu 2	Accu 3	Accu 4
Accumulator defaults prior to execution of arithmetic operations	a	b	c	d
L KF 30	30	a	c	d
L KF 3	3	30	c	d
ENT	3	30	30	c
L KF 4	4	3	30	c
* F	12	30	c	c
+ F	42	c	c	c
L KF 6	6	42	c	c
/F	7	c	c	c

Operation	Parameters	Description
ADD BF	-127 to +127	Add byte constant (fixed point) to accu 1 [1]
ADD KF	-32 768 to +32 767	Add fixed point constant (word) to accu 1 [1]
TAK		Exchange the contents of accus 1 and 2 [2]

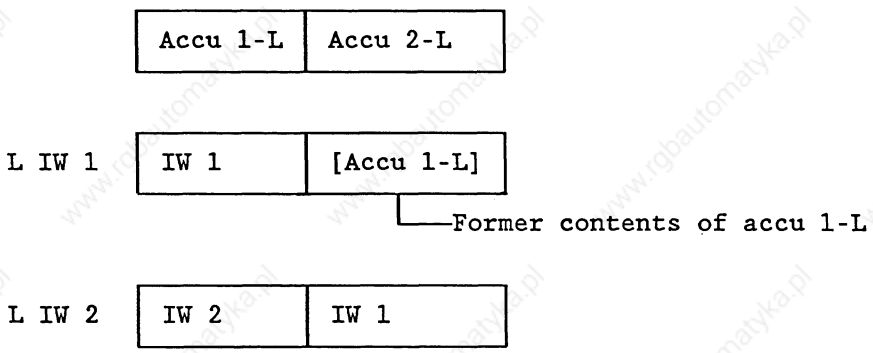
- Digital logic operations

Operation	Description
AW	Digital ANDing of accus 1 and 2
OW	Digital ORing of accus 1 and 2
XOW	Exclusive digital ORing of accus 1 and 2

Accumulators 3 and 4 are not affected, but the condition codes CNC1 and CNC0 are (see section 4.1).

By means of two loading operations, accumulators 1 and 2 can be loaded corresponding to the operands of the loading operation. Then, the contents of both accumulators can be operated on digitally.

Example



[1]) Accus 2, 3 and 4 are not changed
[2]) Accus 3 and 4 are not changed

Organizational functions

- Jump functions

The destination of unconditional and conditional jumps is specified symbolically (a maximum of 4 characters beginning with a letter). The symbolic parameter of the jump instruction is identical to the symbolic address of the statement to be jumped to. When programming, it should be taken into account that the absolute jump distance does not cover more than ±127 words and that a STEP 5 statement can consist of more than one word. Jumps can only be carried out within a block; jumps across segments are not permissible.

Note: jump statement and jump destination must be in one segment. Per segment only one symbolic address is permitted for jump destinations. These conditions do not apply to the JR jump, for which an absolute jump distance is specified as a parameter.

Operation	Description
JU = addr	Jump unconditional. An unconditional jump is carried out under all conditions.
JC = addr	Jump conditional. A conditional jump will be carried out if RLO = 1. If RLO = 0, the statement will not be carried out and the result of the logic operation will be set to RLO = 1.
JZ = addr	Jump condition: CNC1, CNC0. A jump will only be carried out if CNC1 = 0 and CNC0 = 0. The logic operation result is not changed.
JN = addr	Jump condition: CNC1, CNC0. A jump will only be carried out if CNC1 ≠ CNC0. The logic operation result is not changed.
JP = addr	Jump condition: CNC1, CNC0. A jump will only be carried out if CNC1 = 1 and CNC0 = 0. The logic operation result is not changed.
JM = addr	A jump will only be carried out if CNC1 = 0 and CNC0 = 1. The logic operation result is not changed.
JO = addr	Jump on overflow. A jump will be carried out if the condition code OV = 1. If there is no overflow, (OV = 0) the jump will not be carried out. The logic operation result is not changed. An overflow occurs if the permissible area for the numerical representation involved is exceeded by an arithmetic operation.

addr = symbolic address (a maximum of 4 characters)

Operation	Description
JS = addr	Jump if the condition code OS (latching overflow) is set (OS = 1).
JR [1]) -32 768 to +32 767	Jump over the system software.

addr = symbolic address (a maximum of 4 characters)

- Shift functions

Operation	Description
SLW 0 to 15	Shifting to the left (zeros are filled in from the right).
SRW 0 to 15	Shifting to the right (zeros are filled in from the left).
SLD 0 to 32	Shifting a doubleword to the left (zeros are filled in from the right).
SSW 0 to 15	Shifting to the right with sign.
SSD 0 to 32	Shifting a doubleword to the right with sign (sign is filled in from the left).
RLD 0 to 32	Rotating to the left.
RRD 0 to 32	Rotating to the right.

With the shift functions only accu 1 is used. The parameter part of the commands specifies up to how many positions the accu contents are shifted or rotated. With SLW, SRW and SSW, only the less significant word is involved with the shift functions, with SLD, SSD, RLD and RRD the entire contents of accu 1 (32 bits) are used.

Shift functions are carried out unconditionally. The last bit shifted out can be interrogated by means of jump functions. The CNC0 and CNC1 condition codes are affected (see section 4.1).

With JZ, a jump can be carried out if the bit is 0. With JN, a jump can be carried out if the bit is 1.

[1]) System function

Examples

```
STEP 5 program:      Contents of the data words

:L    DW52           H = 14AF
:SLW  4
:T    DW53           H = 4AF0
```

```
STEP 5 program:      Accu 1 contents (hexadecimal):

:L    ED0            2348 ABCD
:SLW    4            2348 BCD0
:SRW    4            2348 0BCD
:SLD    4            3280 BCD0
:SSW    4            3480 FBCD
:SSD    4            0348 0FBC
:RLD    4            3480 FBC0
:RRD    4            0348 0FBC
:BE
```

- Conversion functions

Operation	Meaning
CFW	Forming of one's complement of accu 1 (16 bit)
CSW	Forming of two's complement of accu 1 (16 bit)
CSD	Forming of two's complement of accu 1 (32 bit)
CBW	Fixed point conversion (16 bit) from BCD to binary
BDW	Fixed point conversion (16 bit) from binary to BCD
DED	Doubleword conversion (32 bit) from BCD to binary
BDD	Doubleword conversion (32 bit) from binary to BCD
FDG	Conversion of a fixed point number (32 bit) to a floating point number (32 bit)
GFD	Conversion of a floating point number to a fixed point number (32 bit)

Example

The contents of data word 64 are to be inverted bit by bit and stored in data word 78.

```
STEP 5 program:      Data word assignment:

:L   DW64            BP = 0011111001011011
:CFW
T    DW78            BP = 1100000110100100
```

The contents of data word 207 are to be interpreted as a fixed point number and should be stored in data word 51 with the opposite sign.

```
STEP 5 program:     Data word assignment

:L   DW207          F: + 51
:CSW
:T   DW51           F: - 51
```

- Decrementing/incrementing

Operation	Description
D 1 to 255	decrementing
I 1 to 255	incrementing
parameters	

The contents of accu 1 are decremented or incremented by the number specified as a parameter. The execution of the operation is unconditional. It is limited to the right byte (without carry).

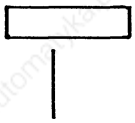
Example

```
STEP 5 program:     Data word assignment

:L   DW7            H = 1010
:I     16
:T   DW8            H = 1020
:D     33
:T   DW9            H = 10FF
```

- Processing functions

Operation	Description
DO DW 0 to 255 (operation)	Process data word The following specified operation will be combined with the parameter specified in the data word and executed.
DO FW 0 to 254 (operation)	Process flag word The following specified operation will be combined with the parameter specified in the flag and executed.

Operation	Description
DO =  enter formal operand	Process formal operand (parameter class: B): Only Q DB, JU, PB, JU FB, JU SB can be substituted.
DOI [1] [2]	Process using a formal operand (indirect). The number of the formal operand to be executed is in accu 1.
DO RS [1], [2] 0 to 255	The command in the system data area (RS) is to be executed.

All operations, with the exception of those listed below, can be combined with DO DW or DO FW:

- Q DB0, Q DB1, Q DB2,
- all two and three word commands (I DB, EX DX, S ES, S EF, CX DX, BC FX and BCC FX),
- operations with formal operand in function blocks,
- JU OBxx and JC OBxx.

The PG does not check whether the combinations are permissible.

Example (process data word)

The contents of data words DW 20 to DW 100 are to be set to signal status "0". The index register for the parameter of the data words is DW 1.

```
        :L   KF 20      load index register
        :T   DW1
F001    :L   KF 0       reset
        :DO  DW1
        :T   DW0
        :L   DW1        increment index register
        :L   KF 1
        :+F
        :T   DW1
        :L   KF 100
        :<=F
        :JC  =F001      jump if index is in the range
        ...             further STEP 5 program
```

[1] System function
[2] The value, which is in the system data or in the formal operand, is interpreted as the operation code of a STEP 5 operation which will then be executed. Permissible operations are as with DO FW and DO DW.

- Disable/enable command output

IA	Disable process interrupts
RA	Enable process interrupts

The "disable/enable interrupts" function can be used, for example, if interrupt-driven processing is to be suppressed during time-driven processing. In the program part situated between statements IA and RA, interrupt-driven processing is no longer possible (see section 3.6.2).

- Special functions

Operation	Parameters	Description
E DB	3 to 255	Generating a data block in the DB RAM
EX DX	1 to 255	Generating an extended data block in the DB RAM
SES [1]	0 to 31	Setting a semaphore
SEF [1]	0 to 31	Enabling a semaphore

- Generate data block

  The command E DBxxx generates a data block with the number xxx (between 3 and 255) in the data block RAM of the CPU. The number of data words in the least significant word in accu 1 is expected. If the corresponding data block already exists or there is not sufficient space in the DB RAM the CPU stops with the error message SFF. The command EXDXxxx generates extended data block DX in the same way as EDXxxx (permissible parameters 1 to 255).

- Set/enable semaphore

  The SESxxx (set semaphore) or SEFxxx (enable semaphore) commands control the data exchange between CPU's or CP's in multiprocessor operation. By setting an SESxxx semaphore, the data area (IPC flags, RI area) designated with the number xxx (0 to 31), which must be determined in the user program, will be disabled for other CPU's. With SEFxxx this data area can be read out of or written into again by other CPU's. A semaphore can only be enabled by the CPU which set it. The commands SES/SEF affect the condition codes (see section 4.1) as follows:

[1]) System operation

CNC1	CNC0	Meaning
0	0	Semaphore has been set by another CPU and cannot be set/enabled.
1	0	Semaphore is set/enabled

The indications can be evaluated by the jump functions.

5 System program - special functions

The organization blocks OB 40 to OB 255 are reserved for special functions and cannot be programmed by the user, but can only be called by means of the unconditional or conditional block call JU OBn or JC OBn. If errors occur during the processing of special functions or if the special function called is not present, the CPU recognises an SFF error (see section 3.7).

List of special functions so far implemented in the R processor

OB 216 to 218	Access to pages
OB 220	Conversion of accu 1 from 16 bit FP to 32 bit FP
OB 221	Set and trigger new cycle time
OB 222	Retrigger cycle time
OB 223	Stop if start-up mode not uniform in multiprocessor operation
OB 224	Block transmission of the IPC flags in multiprocessor operation
OB 226	Read contents of the system program memory location
OB 227	Read cross-checksum of the system program memory
OB 230 to 237	Handling functions
OB 240	Initialization of a shift register
OB 241	Call a shift register
OB 242	Erase a shift register
OB 250	Initialization of a PID controller
OB 251	Call PID controller
OB 254	Copy a DX data block
OB 255	Copy a DB data block

5.1 Fixed point expansion from 16 bits to 32 bits (OB 220)

This special function expands the sign of the 16 bit fixed point number in accu 1-L to the most significant word: if bit $2^{15} = 0$ (positive number), the more significant word will be loaded with zero. Otherwise (with a negative number), it will be loaded with FFFF. This sign extension from 16 bit to 32 bit fixed point numbers is necessary, e.g. before a fixed point-floating point conversion (32 bit, FDG command) of a negative 16 bit fixed point number.

Parameters: none. Possible errors: none.

5.2 Resetting the cycle time (OB 221)

With this special function the user can change the monitoring of the maximum cycle time, (normally preset to 150 ms), to a new value. This cycle time can be between 1 ms and 4000 ms and must be transferred in multiples of milliseconds to accu 1. When this special function is called, the monitoring timer will also be restarted, i.e. the cycle time will be triggered. It is extended by the value set, calculated from the point in time when the special function was called.

Parameters: new cycle time in milliseconds in accu 1.

Possible errors: cycle time not in range 1 ms $\leq$ CYCT $\leq$ 4000 ms.

5.3 Retriggering the cycle time (OB 222)

Special function OB 222 retriggers the cycle time monitoring, i.e. the monitoring timer will be restarted. By calling this special function, the maximum permissible cycle time is extended by the value set (normally 150 ms) or determined by OB 221 from the point in time of the call.

Parameters: none. Possible errors: none.

5.4 Reading a storage location of the system program EPROM (OB 226)

The contents of a program storage location (byte) on the system program EPROM are loaded into the least significant byte of acuu 1. The rest of the contents of accu 1 are erased. The former contents of accu 1 are copied into accu 2. The address of the program storage location to be read must be transferred to accu 2 before OB 226 is called.

Parameters: address of the program storage location to be loaded in accu 1.

Possible errors: none.

5.5 Reading the cross-checksum of the system program EPROM (OB 227)

The cross-checksum of the system program EPROM is loaded into the less significant word of accu 1. The rest of the contents of accu 1 are erased. The former contents of accu 1 are copied into accu 2.

Together with special function OB 226 the user can also check the contents of the system program EPROM during the cyclic program execution by adding up the individual storage locations of the EPROM with fixed point addition and then comparing the final total with the cross-checksum.

Parameters: none. Possible errors: none.

5.6 Shift register (OB 240, 241, 242)

5.6.1 Mode of operation

The following diagram illustrates the principle underlying the software shift register. It consists of rows of 8-bit wide storage locations in the data block RAM of the S5 135 U.

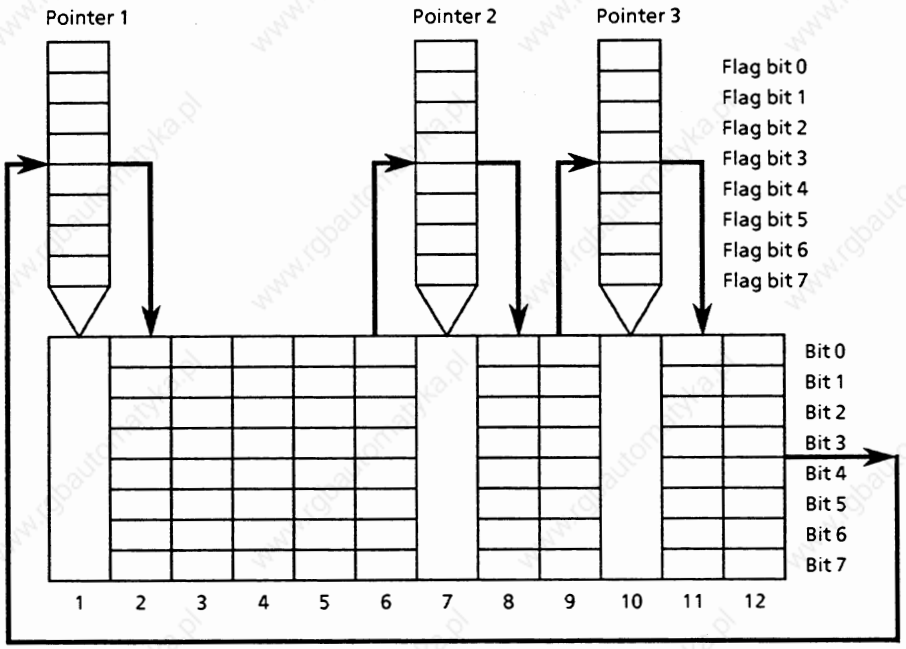


Fig. 18 Schematic diagram of the shift register with 3 pointers and 12 storage locations

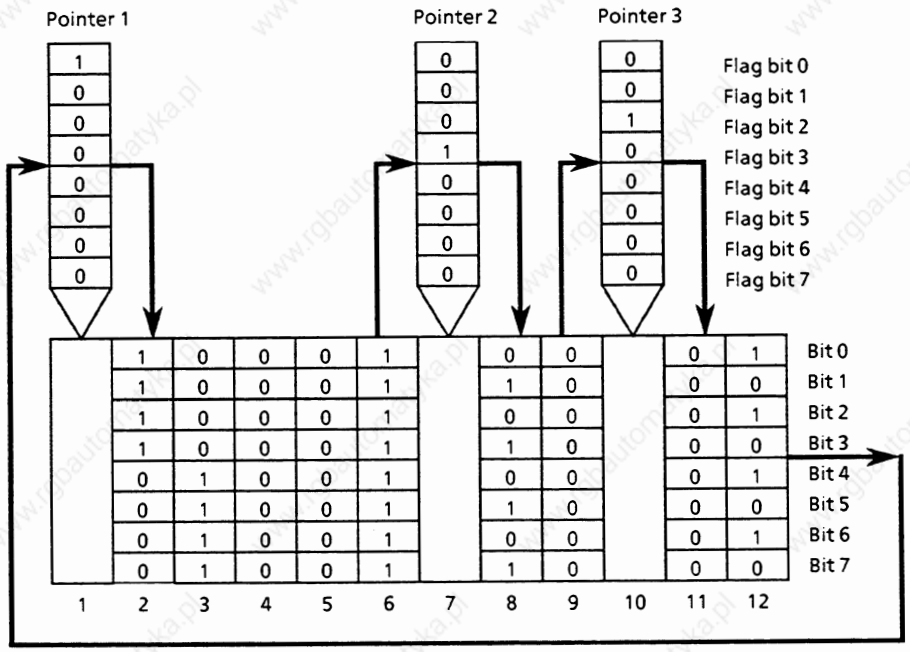


Fig. 19 Schematic diagram of the shift register with 3 pointers and 12 storage locations before the first clock pulse

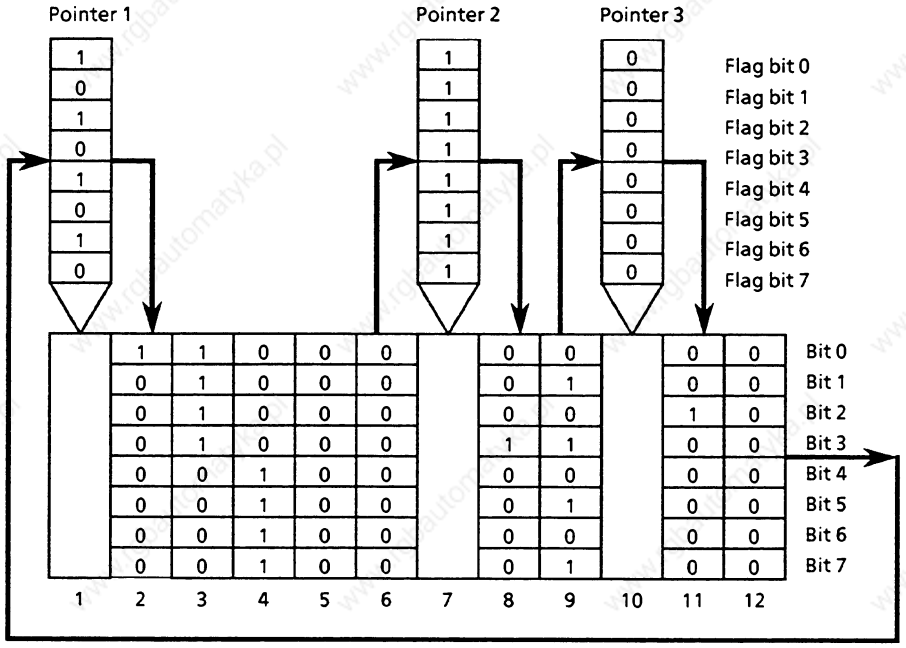


Fig. 20 Schematic diagram of the shift register with 3 pointers and 12 storage locations after the first clock pulse

The number of storage locations between L = 2 and L = 256 can be selected by the user. Individual shift register locations are stored outside the data block RAM in flag bytes, which will be designated as pointers from now on.

The setting of the first pointer (base pointer) is fixed at the first storage cell. All other pointers can be positioned by the user relative to the base pointer. The number of pointers between 1 (base pointer) and 6 can be selected by the user. As in the case of a hardware shift register, the information is shifted through in bytes from storage location 1 as shown by the arrows in the diagrams to storage location L. From there, the information returns to storage location 1. Each shift register function call causes the whole information to be shifted by 1 storage location ≙ 1 clock pulse.

The user can enter information into the shift register, or interpret information from it. This is only possible using the pointers i.e. the flag bytes functioning as pointers.

- Before the SR function call: setting/resetting the flag bits (Fig.19).

		STEP 5
Example:	Set flag bit 0 of pointer 1	S F 0.0
	Set flag bit 3 of pointer 2	S F 1.3
	Set flag bit 2 of pointer 3	S F 2.2

- Call of the shift register function JU OB 241

  Effect: the information (8 bits) of the storage locations is shifted by 1 location.

- Interpretation of the information which is now in the pointers L FB 0

Example: scanning of flag bits 0, 3 and 2 (Fig. 20) at the base pointer.

In the base pointer, therefore, all the information which comes from the entries to all pointers can be interpreted (in the above example, this is only possible after 12 clock pulses).

5.6.2 Initialization and call of the shift register

- Shift register call (OB 241)

In the R processor a maximum of 64 shift registers can be called. The call is carried out in the user program as OB 241. Before the call, the number of the shift register must be loaded in accu 1-L (less significant word), whereby the number must be between 192 and 255.

Before the shift register can be called in the cyclic user program, each shift register must be assigned the same number as a data block e.g. during the initial start. Flag bytes are assigned to each shift register as pointers (a maximum of 6 pointers = 6 flag bytes per shift register).

- Initialization of the shift register (OB 240)

Before a shift register can be called it must first (e.g. during the initial start) be initialized and have parameters assigned to it. This is done by selecting a data block, with the number of the shift register to be initialized and then calling special function OB 240.

The data block is structured according to a fixed pattern which the user must in no way change. It contains all the data necessary for assigning parameters to the shift register.

0	DW 0
Shift register length	DW 1
Number of the 1st flag byte	DW 2
Space n_2	DW 3
.	.
	.
Space n_6	DW 7
0	DW 8

Fig. 21 Structure of the data block for the initialization of a shift register

Data word 0: the contents must always be 0.

Data word 1: the shift register length is the number of byte storage locations of the shift register. It can be in the range $2 \leq L \leq 256$.

Data word 2: the number of the first flag determines the flag block, which is allocated to the pointers. If the user e.g. assigns two pointers there are then three (with the base pointer). Then, the flag specified in the data block, and the two flags which follow it, will be reserved. Care must therefore be taken that enough flags are available for all assigned pointers up to the end of the flag area.

Data word 3 to max. 7: a maximum of 5 entries, i.e. the spaces from the pointers to the base pointer:

n_2 = space from pointer 2 to the base pointer;
n_3 = space from pointer 3 to the base pointer.

The number of pointers including the base pointer must not be greater than the length of the shift register. Also, the space between a pointer and the base pointer must not be greater than the length of the shift register.

The contents of the data word following the last pointer space must always be 0.

A particular memory area at the end of the data block RAM will now be reserved and initialized with the information from this data block. For each shift register, $n = L/2 + 8$ data words are required, i.e. the length of the data block RAM decreases by n data words, whereby the data block RAM end address is shifted to lower addresses.

If a shift register, which is to be initialized, is already present, the area already occupied will be initialized again if the new shift register and the one which is already present, are the same length. Otherwise, the former area will be declared invalid and a new area will be opened.

Parameter assignment: selected data block with $192 \leq$ DB no. ≤ 255

Possible errors:

- illegal data block number;
- insufficient memory space available;
- formal error in the structure of the data block;
- illegal length specification for the shift register;
- parameter assignment error with the pointers.

5.6.3 Erasing the shift register (OB 242)

With this special function a shift register is erased, when its number has to be transferred to accu 2; i.e. the list entry is erased and an invalid identifier is entered in the data block RAM for the corresponding shift register.

Parameter assignment: number of the shift register to be erased in accu 1-L.

Possible error: illegal shift register number.

5.7 PID controller (OB 250 and OB 251)

The user can call one or more PID controllers in the control processor of the S5 135 U.

Each controller must be initialized in the initial start organization block. A data block is used for the transfer of parameters.

The actual control algorithm is integrated in the operating system and can only be called as an organization block by the user. A data block is once again used as data interface between control algorithm and the user program.

- PID controller functions

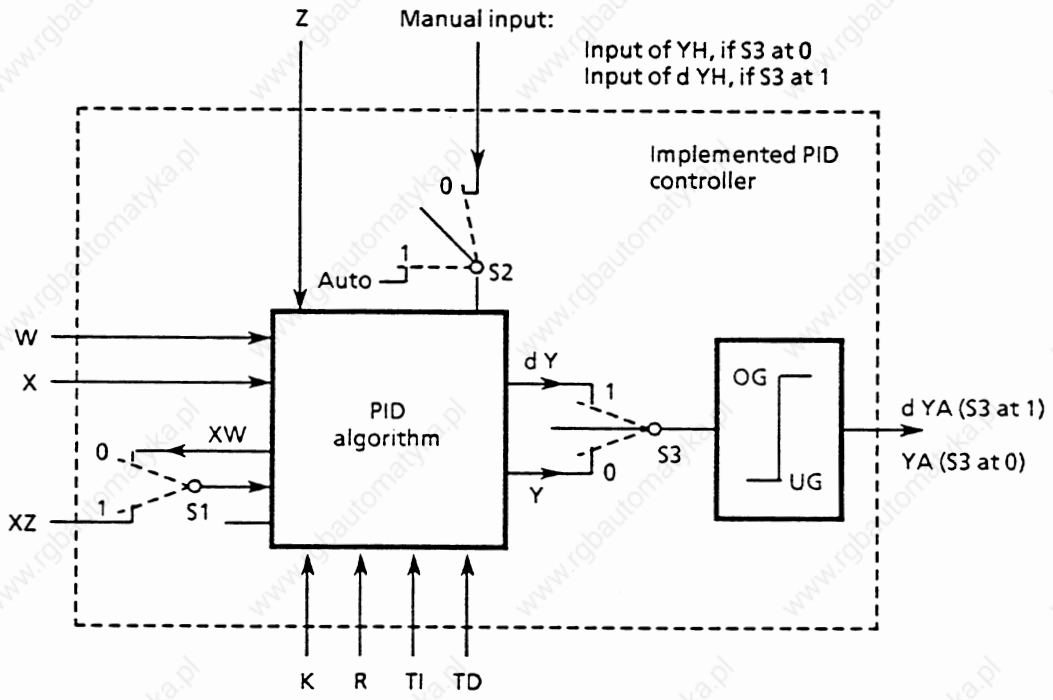


Fig. 22 Block diagram of the PID controller

Index k: kth scan

Switch	Position	Effect
S1 STEU bit 1	 1	The control difference XW_k is supplied to the derivative unit. Via XZ, another signal can be supplied to the derivative unit
S2 STEU bit 0	0 1	Manual operation Automatic operation
S3 STEU bit 3	0 1	Position algorithm Velocity algorithm
S4 STEU bit 4	0 1	With feed forward control Without feed forward control

A function corresponding to the switch positions of this block diagram is achieved during parameter assignment of the PID controller, by setting the control bits appropriately in the control word STEU (see section 5.7.2).

The continuous controller is intended for high speed control systems e.g. in process engineering to control pressure, temperatures or flow rates.

The controller itself is based on a PID algorithm. Its output signal can either be output as a manipulated variable (position algorithm) or as a manipulated variable change (velocity algorithm).

The individual P, I and D components can be switched off by means of their respective parameters R, TI and TD (see section 5.7.2) by presetting the locations concerned with zero. This means that all required controller structures e.g. PI, PID or PD controllers can easily be implemented.

The control deviation XW or (via the XZ output) any influencing quantity or the inverted actual value x can be supplied to the derivative unit.

To compensate for the influence of disturbances a feed forward control on the control element without time response is provided. A process-related disturbance signal Z is fed forward to the control algorithm.

If an inverted control direction is required, a negative K value should be preset.

If the control information (dY or Y) is at a limit, the I component will be switched off automatically, in order to avoid deterioration of the controller response.

The controller program can be supplied with fixed values or adaptive (dynamic) parameters (K, R, TI, TD). They are entered via the memory locations assigned to the individual parameters.

5.7.1 PID algorithm

The PID controller is based on a velocity algorithm, according to which the corresponding control increment dY_k is calculated at a particular point in time t = k.TA according to the following formula:

$$dY_k = K[(XW_k - XW_{k-1})R + \frac{TA}{2TN}(XW_k + XW_{k-1}) + \frac{1}{2}\frac{TV}{TY}(XU_k - 2_{k-1} + XU_{k-2}) + dD_{k-1}]$$

$$= K(dPW_kR + dI_k + dD_k)$$

P component + I component + D component

$dXXX_k$: change in the quantity xxx at time t.

U can be W or Z, depending on whether XW or XZ is supplied to the derivative unit. The following applies:

With XW_k supply:	With XZ supply:
$PW_k = W_k - X_k$	
$PW_k = XW_k - XW_{k-1}$	$PZ_k = XZ_k - XZ_{k-1}$
$QW_k = PW_k - PW_{k-1}$	$QZ_k = PZ_k - PZ_{k-1}$
$QW_k = XW_k - 2XW_{k-1} + XW_{k-2}$	$QZ_k = XZ_k - 2XZ_{k-1} + XZ_{k-2}$

$$dPW_k = (XW_k - XW_{k-1})R$$

$$dI_k = TI.XW_k \qquad TI = \frac{TA}{TN}$$

$$dD_k = \frac{1}{2}(TD.QU_k + dD_{k-1}) \qquad TD = \frac{TV}{TA}$$

If the manipulated variable Y_k is required as controller output at time t_k, it is formed according to the following formula:

$$Y_k = \sum_{m=o}^{m=k} dY_m$$

With most process control designs it is assumed that R = 1, if a P response is required.

The quantity R can be used to set the proportional component of the PID controller.

5.7.2 Data blocks for the PID controller

Controller specific data are entered using a transfer data block (for initialization and call of PID controller see section 5.7.3). The following data should be entered by the user into the transfer data block x:

K, R, TI, TD, W, STEU, YH, BGOG, BGUG

The structure of the transfer block, which must be made up of 49 data words with the numbers 0 to 48, is explained in more detail below.

- Structure of the transfer data block

Word no.	Name	I/O	Number format	PG format	Remarks
0	-	-	-	-	Reserve
1	K	I	GP	KG	Proportional coefficient K > 0: positive control direction i.e. change of setpoint and manipulated variable in same control direction K < 0: negative control direction; floating point number range
3	R	I	GP	KG	R parameter,usually = 1 for controllers with P component; floating point number range
5	TI	I	GP	KG	TI = TA/TN; floating point floating point number range
7	TD	I	GP	KG	TD = TV/TA; floating point number range
9	W_k	I	GP	KG	Entry of setpoint here, if STEU bit 6=1, otherwise in word no. 19 ($-1 \leq W_k < 1$)
11	STEU	I	BM	KM	Control word (see page 112)
12	YH_k	I	GP	KG	Manual entry here if STEU bit 6 = 1; otherwise in word no. 18 ($-1 \leq YH_k < 1$). With velocity algorithms, manipulated variable increments must be specified here.
14	BGOG	I	GP	KG	Upper limit value $0 \leq BGOG < 1$ (YA_k max.) BGUG < BGOG
16	BGUG	I	GP	KG	Lower limit value $-1 \leq BGUG \leq 0$ (YA_k min)
18	YH_k	I	LP	KF	Manual entry here if STEU bit 6 = 0 ($-1 \leq YH < 1$). With velocity algorithms, manipulated variable increments must be specified here.
19	W_k	I	LP	KF	Entry of setpoint here, if STEU bit 6=0 ($-1 \leq W_k < 1$)

Word no.	Name	I/O	Number format	PG format	Remarks
20	MERK	I	BM	KM	Bit 0 = 1: positive limit exceeded; bit 1 = 1: negative limit exceeded
21	X_k	I	LP	KF	Entry of actual value for STEU bit 7=0 ($-1 \leq X_k < 1$)
22	X_k	I	GP	KG	Entry of actual value for STEU bit 7=1 ($-1 \leq X_k < 1$)
24	Z_k	I	LP	KF	Influencing quantity ($-1 \leq Z_k < 1$)
25	Z_k	I	GP	KG	Influencing quantity input here, if STEU bit 7 = 1 ($-1 \leq Z_k < 1$)
27	Z_{k-1}		GP	KG	Previous value of influencing quantity
29	XZ_k	I	LP	KF	Value supplied to the derivative unit via input XZ ($-1 \leq XZ_k < 1$); entry here if STEU bit 7 = 0
30	XZ_k	I	GP	KG	XZ entry here, if STEU bit 7 = 1 ($-1 \leq XZ_k < 1$)
32	XZ_{k-1}		GP	KG	Previous limit of XZ_k
34	PZ_{k-1}		GP	KG	$XZ_{k-1} - XZ_{k-2}$
36	dD_{k-1}		GP	KG	Derivative component
38	XW_{k-1}		GP	KG	Previous value of control deviation
40	P_{wk-1}		GP	KG	$XW_{k-1} - XW_{k-2}$
42	-	-	-	-	Reserve
44	Y_{k-1}		GP	KG	Previous value of the calculated manipulated variable Yk-1 or dY_{k-1} before the limiter
46	YA_k	Q	GP	KG	Output quantity
48	YA_k	Q	LP	KF	Output quantity BGUG ≤ YA ≤ BGOG

Recommended format (KH, KM also possible)

GP = floating point LP = binary or decimal fractions

I = input, Q = output

• Assignment of the control word STEU (word 11 in the transfer block)

Bit	Name	Meaning
0	AUTO	=1: automatic operation =0: manual operation
1	XZ_ON	=1: a different quantity, which must not be XW_k, is supplied to the derivative unit via the XZ input. =0: XW_k is supplied to the derivative unit. The XZ input is ignored.
2	REG_AUS	=1: when calling the controller (OB 251) all values in the controller DB (DW 20 to DW 48) are erased once (except K, R, TI, TD, BGOG, BGUG, STEU, YHk, W_k, Z_k and Z_{k-1}). The controller is switched off. The previous value of the influencing quantity is updated. =0: controller on
3	VELOC	=1: velocity algorithm =0: position algorithm
4[1])	MAN	=1: for GESCHW = 0 (position algorithm) the manipulated variable output last is retained. For GESCHW = 1 (velocity algorithm), positioning increment $dY_k = 0$ is set. =0: for GESCHW = 0, after switching over to manual operation, the value of the manipulated variable YA output is brought exponentially to the manually set value in 4 sampling steps. After this, further manually set values are immediately acepted at the controller output. For GESCHW = 1, the manually set values are immediately switched to the controller output. With manual operation, the limits are effective, and the following quantities are updated: 1) X_k, XW_{k-1} and PW_{k-1} 2) XZ_k, XZ_{k-1} and PZ_{k-1}, if STEU bit 1 = 1 3) Z_k and Z_{k-1}, if STEU bit 5 = 0 The value dD_{k-1} is set to 0. The algorithm is not calculated.
5	NO_Z	=1: no feed forward control =0: with feed forward control
6	PGDG	=1: W_{k-}, Y_{Hk-} input as a floating point number. =0: input as a binary/decimal fraction.
7	VAR_GP	=1: the variables X_k, XZ_k and Z_k are entered as floating point numbers. =0: input of the variables as a binary/decimal fraction.
8	STOS	=1: no bumpless manual-automatic transfer. =0: bumpless manual-automatic transfer.
9..15		No significance

[1]) Only relevant for manual operation (AUTO = 0)

5.7.3 Initialization and call up of the PID controller in the STEP 5 program

- Initialization in the start up OB's 20/21/22

- Selection of the transfer DB x (contains parameters)
- Call of OB 250 (controller initialization routine)

For data transfer, each controller must use its own DB x ($x \leq 254$). After copying this DB x into the data block RAM, the system program automatically generates another DB x + 1, which the controller uses as a data field during cyclic operation; therefore the appropriate DB numbers must be kept available. These DB x + 1's are the data interface between the controller and the user or peripherals.

Caution! If the DB x + 1 was not kept available during initialization, it will be used, without any indication from the operating system, as a controller data field, provided that it is the same length as a controller DB (48 doublewords) and data words 20 to 48 will be erased. Otherwise, the PC stops.

As well as DB data blocks, DX extended data blocks can also be used. The initialization of the DX is analogous to that of the DB.

- Calling up the controller during the cycle

Calling up the controller after the scan time has elapsed

- Select data field DB x + 1
- Load input data X_k, XZ_k, Z_k and YH_k or a subset of them (see page 110)
- Input data are converted to the correct format and transferred into DB x + 1
- Call OB 251 (PID controller)
- Load output data YA_k from DB + 1
- Conversion of the data and transfer to process I/O's

5.7.4 Format of controller inputs and outputs

The PID controller algorithm uses the floating point format internally to represent numbers and can be supplied with floating point values. It can also be supplied with binary or decimal fractions (see bit 6 and 7 in control word STEU, page 112). In this case, the controller automatically converts the words into floating point format whenever it is called.

In the STEP 5 program the conversion of words from the input and output modules requires less runtime if the binary or decimal fraction format is used.

- Inputs

W, YH, X, Z and XZ can be input as floating numbers or binary/decimal fractions. The data transfer block has different memory locations for each quantity.

- Input as binary/decimal fractions

For details of these numbers see section 5.7.7.

Caution! In order to keep within the nominal input range of the analog input modules, it is important to remember that the bit pattern for a particular input value is different from that when the whole input range is used. This fact must be considered, particularly when setting a setpoint, otherwise it might be possible that a setpoint, input via the PG, is not reached, although the actual value is far higher than the desired value.

If the analog-digital converter used supplies negative numbers as values with a sign, then the two's complement must be formed from them before they are transferred into the controller DB. Binary position 15 must then be set to 1.

If, with the analog-digital converter, the number 0 can occur as a value with sign as follows:

1000000000000000

then the two's complement must not be formed from it. Instead, the number must be transferred to the controller DB as +0:

0000000000000000

- Output

The controller output YA is available in the DB as a left and fixed point number. The format of binary/decimal fraction inputs and outputs must be converted depending on the input and output modules used (analog-digital converter, digital-analog converter) before and after the controller call up in the STEP 5 user program, before they are transferred into or out of the controller DB.

5.7.5 General notes

If STOS (STEU bit 8) is set at zero, switching over from manual to automatic operation is bumpless; i.e. any large control deviation will only be compensated by the I component. If, however, TI = TA/TN = 0 is selected (P or PD controller), the control deviation does not change the manipulated variable during the switchover.

This can be avoided by setting STOS = 1. A control deviation will then be compensated quickly when switching over manual-automatic, irrespective of whether TI = 0 or not. The manipulated variable jump resulting from this corresponds to the value of the control deviation and is therefore not an arbitrary influence on the operation of the controller.

If required, bits 0 and 1 of MERK can be displayed in order to indicate that the manipulated variable (with velocity algorithm, the positioning increment) is in the upper or lower limit. Since these bits are interpreted by the algorithm for disabling the I component, they must not be overwritten.

The controller data blocks DB x + 1 cannot be reloaded during cyclic operation.

If a cascaded control with two or more controllers is set up, the following must be observed:

- If the cascade is to be split, then either all controllers must go over to manual operation at the same time, so that no controller can drift owing to its I component, or at least the controller of the outer loop must be on manual, so that the last manipulated variable, which corresponds to the setpoint of the inner loop, is maintained or can be set to a safe value.

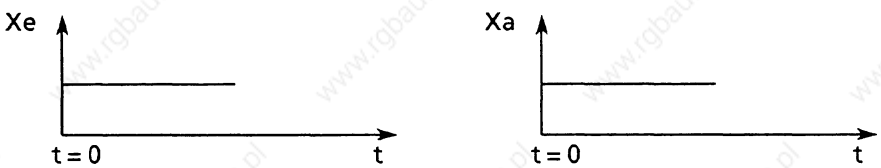
- If the cascade is to be closed, then both loops should operate simultaneously on automatic or at least the inner loop so that the manipulated variable of the outer loop can be taken as the setpoint.

If during the switchover to manual operation, the controlled system is separated from the controller and is adjusted directly at the final control element, the resulting manipulated variable must be supplied to the controller by manual input. This ensures that when switching over from manual to automatic operation, the controller output corresponds to the manipulated variable set in manual operation. With the velocity algorithm this is the manipulated variable deviation.

5.7.6 Controller characteristic quantities

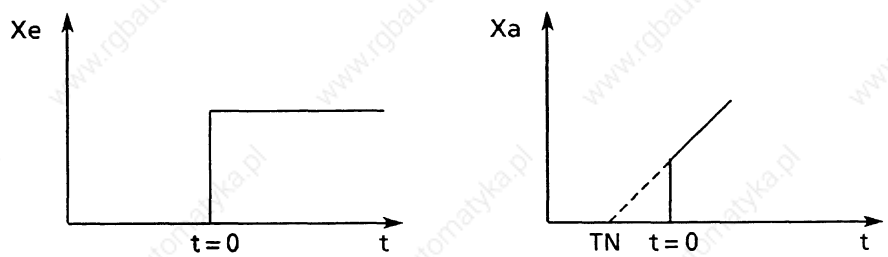
- P controller

The characteristic quantity for a P controller is K. This is the quotient of output and input quantity: $K = X_{out}/X_{in}$.



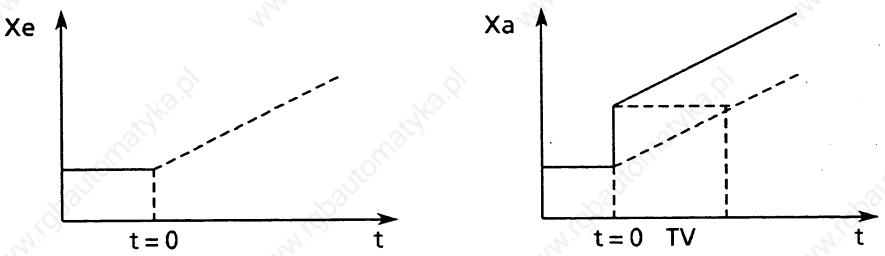
- PI controller

The proportional coefficient K and integral time constant (reset time) TN are the characteristic quantities for a PI controller. Proportional coefficient K is the quotient of the input and output quantities and determines the P response. The reset time TN is the time required to respond and achieve the same change in the manipulated variable by means of the I action, as is brought about by the P component.



- PD controller

Proportional coefficient K (see above) and derivative time constant TV are the characteristic quantities for a PD controller. The feed forward time is the time which a P controller would need given a constant rate of change in order to bring about the same change in the output quantity, as a PD controller immediately brings about as a result of its D component. To determine the feed forward time a linear change of the input value is used instead of a step function.



- PID controller

The characteristic quantities of a PID controller are the proportional coefficient K, the reset time TN and the feed forward time TV. These determine the P, I and D responses.

Abbreviations for PID controllers

dY_k	calculated positioning increment
dZ_k	influencing increment
GP	floating point representation
k	kth scan
K	proportional coefficient
LP	left point representation
UL	upper limit (limiter)
R	R parameter
TA	scan time
TD	TV/TA
TI	TA/TN
t	scan point (time) = k.TA
TN	reset time
TV	feed forward time
LL	lower limit (limiter)
W_k	setpoint
X_k	actual value
XW_k	control derivation
Y_k	calculated manipulated variable
YA_k	value of manipulated variable (positioning increment or manipulated variable)
Z_k	influencing quantity

5.7.7 Binary/decimal fractions

To represent a binary/decimal fraction in the data block, one word is necessary. The relationship between a decimal number, a binary number and the representation in KF format on the programmer is shown in the following example

Fraction		Fixed point
Decimal	Binary	number
-0.999...	1000000000000001	-32767
-0.75	1010000000000000	-24576
-0.5	1100000000000000	-16384
-0.25	1110000000000000	- 8192
0	0000000000000000	0
+0.25	0010000000000000	+ 8192
+0.5	0100000000000000	+16384
+0.75	0110000000000000	+24576
+0.999...	0111111111111111	+32767

In binary representation negative fractions are derived from positive fractions by forming the two's complement.

Binary/decimal fractions (LP) can be converted into the values displayed (KF) on the programmer with the following formula:

LP x 32767 = KF

with $-1 < LP < +1$ and $-32767 \leq KF \leq +32767$

5.8 Data block copying function (OB 254, OB 255)

The OB 254 and 255 special functions operate identically, whereby the OB 254 routine is for DX blocks, and the OB 255 is for DB blocks. In terms of their execution a difference must be made between the two following situations:

- Copying a data block from the user memory into the DB RAM.

When a data block in the user memory is copied into the DB RAM it keeps its original block number. The new start address of the data block is entered in the address list in DB 0, and the old address of the block is overwritten.

Parameter assignment: accu 1-L: number of data block to be copied
accu 1-H: 0

Possible errors:

- Block to be copied is not present.
- Block is already in the DB RAM (carry out function only once - preferably during the initial start).
- Not sufficient memory space in the DB RAM.

If one of these errors occurs, the function will not be carried out and the error message SFF (see section 3.7) will be sent.

- Duplicating a data block into the DB RAM

The original data block can reside both in the user memory and in the DB RAM and is duplicated onto a new data block with another number in the DB RAM. The start address of the new data block is entered in the address list in DB 0. The start address of the old block remains in DB 0.

The start address will only be entered in DB 0 when the transfer is completed and all identifiers are correctly entered in the block header. Therefore, the copied block is only recognised as valid or present by the system program after the transfer has been completed.

Parameter assignment: accu 1-L: number of the data block to be copied
accu 1-H: number of the new data block

Possible errors:

- block to be copied is not present
- new block is already present
- not sufficient memory space in DB RAM

5.9 Block transfer of the IPC flags

Normally, the IPC flags specified in DB 1 by the user are transmitted **in bytes** by the system program to the CPU's during multiprocessor operation (see section 2.3.1 and Fig. 14). In multiprocessor operation this transfer takes place with each CPU acting independently. This is intended to keep the time that the bus is blocked by a CPU to a minimum, the bus assignment being controlled by COR. Therefore, only bytes can be used as coherent units of information.

By calling OB 224 during the initial start with each start up mode used and with each CPU involved in the IPC flag transfer, the user can transmit all the IPC flags specified in DB 1 in blocks. Each CPU can only start its IPC transfer when the transfer by another CPU has been completed. The cyclic program execution will be appropriately delayed (cycle time!). By means of this special function, IPC interleaved updating will be separated out for the individual CPU's in order to clarify the assignment of all the IPC flag information. The function is only effective in the COR operating mode "RUN" with a bus enable time of < 8 us (see COR operating instructions).

5.10 Comparing the start-up mode

By calling OB 233 - e.g. during the initial start or at the beginning of the cyclic program execution - the corresponding CPU will check during multiprocessor operation whether the start up modes of **all** the CPU's involved are the same. If they are not, then the corresponding CPU stops with the SFF error message (see section 3.7).

5.11 Access to pages (OB 216 to 218)

These organization blocks make access possible to so called pages. These pages are memory areas, one or more of which are present on communications processors, certain intelligent I/O modules and certain coordinators for multiprocessing. If the page size is 1024 bytes, the pages occupy the address space

F400H to F7FFH.

If the page size is 2048 bytes, the pages occupy the address space

F400H to FBFFH.

Selection (addressing) of the "current" pages is carried out via the select or ident. register (similar to chip select). The location with the address FEFF cannot be read. The operation blocks contain the following functions:

OB 216	Writing a byte/word/doubleword on a page
OB 217	Reading a byte/word/doubleword from a page
OB 218	Occupation of a page by the CPU (used for coordination in multiprocessor operation)

On the one hand these functions are used for test purposes; on the other, these elementary functions make possible the programming of handling blocks or similar.

Notation:

Accu 1:	Accu 1,	32 bits
Accu 1:1-L:	Accu 1, low word	16 bits
Accu 1:1-LL:	Accu 1, low word, low byte,	8 bits
Accu 1:1-LH	Accu 1, low word, high byte,	8 bits

etc.

5.11.1 Writing data on a page (OB 216)

The block transmits one byte/word/doubleword from accu 1 (right justified) to the page. The destination address on the page must be present in accu 2-L and the page number in accu 3-LL. Accu 3-LH contains 0, 1 or 2 as an identifier, depending on whether a byte/word/doubleword is to be transmitted. The page number can, therefore, have values between 0 and 255, the permissible addresses (in accu 2-L) must be between 0 and 2047.

The addressing of the page and transmission of the complete data (1/2/4 byte) are an inseparable unit.

If no transmission is possible, because:

- accu 3-LH contains an illegal value
- the address specified is illegal or does not exist on this page
- the page specified does not exist, or
- no pages exist whatsoever

the accu contents remain unchanged; all (!) condition codes in the logic module (e.g. RLO) will be erased.

If the transmission is carried out successfully, the contents of accus 1 and 3 remain unchanged; accu 2.L contains a value increased by 1/2/4 (depending on the data length); RLO will be set; the remaining condition codes in the logic module will be erased.

Example (not runtime optimized)

On page 7, an area with the address 50 to 69 (= 20 bytes) is to be erased (assuming the area does exist):

```
         L    KY    1.7  ;1 = word transmission, 7 = page number
         L    KB    50   ;start address
         ENT             ;writing into accu 3
         L    KB    0    ;writing into accus 2 and 1
MARK :   JU   OB    216  ;erase word, increase address by 2
         +F              ;TAK also possible
         L    KB    70   ;loop counter final value
         ><F             ;address = end address?
         SLD  16         ;erase accu 1-L, accu 2-L contains address
         JC=  MARK       ;jump if address < end address
         .               ;continuation
         .               ;following end of loop
```

The same example, but also with a check for whether the area exists:

```
                         ;begin initialization
         L    KY    1.7  ;1 = word transfer, 7 = page number
         L    KB    50   ;start address
         ENT             ;writing into accu 3
         L    KB    0    ;writing into accu 2, accu 1 irrelevant
                         ;end initialization

                         ;begin loop
LOOP :   TAK             ;+F also possible
         L    KB    70   ;cycle counter final value
         !=F             ;address = end address?
         JC=END          ;if yes, exit loop
         SLD  16         ;erase accu 1-L, accu 2-L contains address
         JU   OB216      ;erase word, increase address by 2
         JC=LOOP         ;jump if transmission free of error
                         ;end loop

                         ;begin error handling
ERR :    .               ;
         .               ;
         BEU             ;end error handling

                         ;begin continuation
END :    .               ;
         .               ;
         .               ;
         BE              ;end continuation
```

5.11.2 Reading data from a page (OB 217)

The block transmits a byte/word/doubleword from a page to accu 1 (right-justified). The destination address on the page must exist in accu 2-L, the page number in accu 3-LL. Accu 3-LH contains 0, 1 or 2 as an identifier, depending on whether a byte/word/double-word is to be transmitted. Therefore, the page number can have values between 0 and 255; the permissible addresses (in accu 2.L) must be between 0 and 2047.

The addressing of the page and the transmission of the complete data (1/2/4 bytes) form an inseparable unit.

If transmission is not possible because:

- accu 3-LH contains an illegal value
- the address specified is illegal or does not exist on this page
- the page specified does not exist, or
- no pages exist whatsoever,

the accu contents remain unchanged; all (!) condition codes in the logic module (e.g. RLO) will be erased.

If the transmission is carried out successfully, the contents of accu 3 remain unchanged; accu 2-L contains a value increased by 1/2/4 (depending on the data length); accu 1 contains the value read out (right-justified). Anything remaining in the 32 bit accu will be erased. RLO will be reset, the remaining condition codes in the logic module will be erased.

Example

A data block with the addresses 100 to 107 (= 8 bytes) of page 7 are to be transferred to flags 200 to 207:

```
L    KY    2.7   ;2 = doubleword transmission, 7 = page
                   number
L    KB    100   ;start address on page
ENT              ;write into accu 3
L    KB    0     ;write into accu 2, accu 1 is
                   irrelevant
JU   OB    217   ;read bytes 0 to 3, increase address by 4
T    FD    200   ;FB 200 to FB 203
JU   OB    217   ;read bytes 4 - 7, increase address by 4
T    FD    204   ;FB 204 to FB 207
```

5.11.3 Occupying a page (OB 218)

The block transmits the slot identifier of this CPU to a page, if the contents of the addressed location are equal to zero. The destination address on the page must exist in accu 1-L, the page number in accu 2-LL. Therefore the page number can have values between 0 and 255. The permissible addresses (in accu 1-L) must be between 0 and 2047.

The addressing of the page, the reading, and, if applicable, writing of the slot identifier form an inseparable unit.

If it is not possible to enter the slot identifier because:

- the address specified contains a value not equal to zero
- the address specified is illegal or does not exist on this page
- the page specified does not exist, or
- no pages exist whatsoever,

the accu contents remain unchanged; all (!) condition codes in the logic module (e.g. RLO) will be erased.

If the transmission is carried out successfully, the accu contents remain unchanged; RLO will be set; the remaining condition codes in the logic module will be erased.

5.12 System program auxiliary functions (OB 230 to 237)

The special functions OB 230 to 237 contain auxiliary functions. Calling them is not permitted and can lead to errors.

6 Overview of STEP 5 Operations

Basic functions

Operation	Parameters
• Binary logic operations:	
A I	0.0 to 127.7
A Q	0.0 to 127.7
A F	0.0 to 255.7
A D [1]	0.0 to 255.15
A T	0.0 to 127
A C	0.0 to 127
AN I	0.0 to 127.7
AN Q	0.0 to 127.7
AN F	0.0 to 255.7
AN D [1]	0.0 to 255.15
AN T	0.0 to 127
AN C	0.0 to 127
O I	0.0 to 127.7
O Q	0.0 to 127.7
O F	0.0 to 255.7
O D [1]	0.0 to 255.15
O T	0.0 to 127
O C	0.0 to 127
O I	0.0 to 127.7
ON Q	0.0 to 127.7
ON F	0.0 to 255.7
ON D [1]	0.0 to 255.15
ON T	0.0 to 127
ON C	0.0 to 127
)	
A(	
O(	
O	
• Compare functions:	
!=F	
><F	
>F	
>=F	
<F	
<=F	
!=D	
><D	
>D	
>=D	
<D	
<=D	
!=G	
><G	
>G	
>=G	
<G	
<=G	

[1]) Word 1: B2 + bit address; B3 + relative address

Operation	Parameters
• Memory operations:	
S I	0.0 to 127.7
S Q	0.0 to 127.7
S F	0.0 to 255.7
S D [1]	0.0 to 255.5
R I	0.0 to 127.7
R Q	0.0 to 127.7
R F	0.0 to 255.7
R D [1]	0.0 to 255.15
= I	0.0 to 127.7
= Q	0.0 to 127.7
= F	0.0 to 255.7
= D [1]	0.0 to 255.15
• Loading functions:	
L IB	0 to 127
L IW	0 to 126
L ID	0 to 124
L QB	0 to 127
L QW	0 to 126
L QD	0 to 124
L FB	0 to 255
L FW	0 to 254
L FD	0 to 252
L DL	0 to 255
L DR	0 to 255
L DW	0 to 255
L DD	0 to 254
L T	0 to 127
L C	0 to 127
L PB	0 to 127 128 to 255
L PW	0 to 126 128 to 254
L OB	0 to 255
L OW	0 to 254
LD T	0 to 127
LD C	0 to 127
L KB	0 to 255
L KC	2 alphanumeric characters
L KM	bit pattern (16 bit)
L KH	0 to FFFF
L KF	-32 768 to +32 767
L KY	0 to 255 for each byte
L KT	0.0 to 999.3
L KC	0 to 999
L KG	[2])

[2]) $\pm 0.1469368 \times 10^{-38}$ to $\pm 0.1701412 \times 10^{39}$

Operation	Parameters
• Timer and counter operations:	
SP T	0 to 127
SE T	0 to 127
SR T	0 to 127
SS T	0 to 127
SF T	0 to 127
R T	0 to 127
S C	0 to 127
R C	0 to 127
CU C	0 to 127
CD C	0.0 to 127
• Transfer functions:	
T IB	0 to 127
T IW	0 to 126
T ID	0 to 124
T QB	0 to 127
T QW	0 to 126
T QD	0 to 124
T FB	0 to 255
T FW	0 to 254
T FD	0 to 252
T DR	0 to 255
T DL	0 to 255
T DW	0 to 255
T DD	0 to 254
T PB	0 to 127
	128 to 255
T PW	0 to 126
	128 to 254
T OB	0 to 255
T OW	0 to 254
• Block calls:	
JU PB	0 to 255
JU FB	0 to 255
JU SB	0 to 255
JC PB	0 to 255
JC FB	0 to 255
JC SB	0 to 255
C DB	0 to 255
BE	
BEC	
BEU	
JU OB	1 to 39
JC OB	1 to 39
CX DX	1 to 255
BC FX	0 to 255
BCC FX	0 to 255
JU OB [1]	40 to 255
JC OB [1]	40 to 255

[1] special function call
[2] system operation

Operation	Parameters
• Arithmetic operations:	
+F	
-F	
xF	
:F	
+G	
-G	
xG	
:G	
• Other functions:	
NOP 0	
NOP 1	
STP	
BLD	

Supplementary functions

Operation	Parameters
• Logic functions, in words:	
AW	
OW	
XOW	
• Timer and counter functions:	
FR T	0 to 127
FR C	0 to 127
FR =	formal operand
SP =	formal operand
SR =	formal operand
SEC =	formal operand
SSU =	formal operand
SED =	formal operand
RD =	formal operand
• Loading and transfer functions:	
L =	formal operand
LC =	formal operand
LW =	formal operand
LD =	formal operand
T =	formal operand
L RS	0 to 255
L RI	0 to 255
T RI [2]	0 to 255
T RS [2]	0 to 255
LIR [2]	0 to 15
TIR [2]	0 to 15
TNB [2]	0 to 255
TNW [2]	0 to 255

Operation	Parameters
• Binary logic operations	
A =	formal operand
AW =	formal operand
O =	formal operand
ON =	formal operand
• Conversion functions:	
CFW	
CSW	
CSD	
CBW	
BDW	
DED	
BDD	
FDG	
GFD	
• Shift functions:	
SLW	0 to 15
SRW	0 to 15
SLD	0 to 32
SVD	0 to 32
RLD	0 to 32
RRD	0 to 32
SVW	0 to 15
• Jump functions:	
JU =	Symbolic address
JC =	Symbolic address
JZ =	Symbolic address
JN =	Symbolic address
JP =	Symbolic address
JM =	Symbolic address
JO =	Symbolic address
JS [2]	Symbolic address
JR [1]	-32 768 to +32 767

Operation	Parameters
• Setting operations:	
S =	formal operand
RB =	formal operand
= =	formal operand
• Other functions	
RA	
IA	
ENT	
D	0 to 255
I	0 to 255
DO =	formal operand
DO DW	0 to 255
DO FW	0 to 255
BI [1]	
DO RS [1]	0 to 255
TAK	
BLD	0 to 255
• Arithmetic operations:	
ADD BF	-127 to +127
ADD KF	-32 768 to +32 767
• Special functions:	
E DB	0 to 255
EX DX	0 to 255
SES 1)	0 to 31
SEF 1)	0 to 31

[1]) System operation
[2]) Word 1: jump distance (2 bytes)

7 Error information

7.1 Error information in system data 3 and 4

- Error information during setting up of the block address lists

Error identifier SD3	SD4	Explanation	Reaction of the system in the initial start
8001H	yyyyH	Incorrect block length yyyy = address of block with incorrect length	Abort
8002H	yyyyH	Calculated end address of block incorrect in memory yyyy = block address	Abort
8003H	yyyyH	Invalid block address yyyy = address of incorrect identifier	Abort
8004H	yyyyH	Organization block number too great (permitted: OB 1 to 47) yyyy = address of block with incorrect number	Abort

- Error information during setting up of the address lists for the updating of the process image

Error identifier SD3	SD4	Explanation	Reaction of the system in the initial start
0400H	-	No error occurred	Continue
0410H	yyyyH	Invalid identifier yyyy = incorrect identifier	Abort
0411H	yyyyH	Incorrect parameter in "digital inputs" address list yyyy = address of incorrectly specified input byte	Abort
0412H	yyyyH	Incorrect parameter in "digital outputs" address list yyyy = address of incorrectly specified output byte	Abort
0413H	yyyyH	Incorrect parameter in "IPC flag input" address list yyyy = address of incorrectly specified output byte	Abort

Error identifier SD3	SD4	Explanation	Reaction of the system in the initial start
0414H	yyyyH	Incorrect parameter in "IPC flag output" address list yyyy = address of incorrectly specified flag byte	Abort
0415H	yyyyH	Invalid number of timer locations (permitted: 128) yyyy = incorrect number of timer locations	Abort
0419H	yyyyH	Acknowledgement delay at digital inputs yyyy = address of input byte not acknowledging	Abort
041AH	yyyyH	Acknowledgement delay at digital outputs yyyy = address of output byte not acknowledging	Abort
041BH	yyyyH	Acknowledgement delay at IPC flag input yyyy = address of flag byte not acknowledging	Abort
041CH	yyyyH	Acknowledgement delay at IPC flag output yyyy = address of flag byte not acknowledging	Abort

- Error information during evaluation of DB 2 - initialization of the controller call distributor

Error identifier SD3	SD4	Explanation	Reaction of the system in the initial start
0421H	DByyH	Data block not loaded yy = number of data block not loaded	Abort
0422H	FByyH	Function block not loaded yy = number of function block not loaded	Abort

- Error information during evaluation of the DX 0 operating system parameter assignment

Error identifier SD3	SD4	Explanation	Reaction of the system in the initial start
0431H	yyyyH	Invalid block identifier yyyy = incorrect identifier	Abort
0432H	yyyyH	Unknown parameter yyyy = incorrect parameter	Abort
0433H	yyyyH	Parameter not permitted yyyy = incorrect parameter	Abort
0434H	yyyyH	Illegal number of timer locations (permitted: 128) yyyy = incorrect number of timer locations	Abort
0435H	yyyyH	Cycle time not permitted (permitted: 1 ms to 4 sec) yyyy = incorrect time value	Abort

7.2 Error information about Accu 1 and Accu 2

- Error information during controller processing

Error identifier Accu1	Accu 2	Explanation	Reaction of the system in the initial start
0801H	DByyH	Scan time error yy = number of controller data block concerned	Call OB 34 (if not masked)
0802H	DByyH	Controller data block not loaded yy = number of data block not loaded	Call OB 34
0803H	FByyH	Controller function block not loaded yy = number of function block not loaded	Call OB 34
0880H	yyyyH	Acknowledgement delay (AKD) during controller process image updating	Call OB 34

- Error information with MC5 command code errors

Error identifier Accu1	Accu2	Explanation	Reaction of the system in the initial start
1801H	-	Substitution error with DO RS command	Call OB 27
1802H	-	Substitution error with DO DW/DO FW command	Call OB 27
1803H	-	Substitution error with DO X/DO IX command	Call OB 27
1804H	-	Substitution error with LX/TX command	Call OB 27
1805H	-	Substitution error with AX/ANX/OX/ONX/=X/SX and RBX command	Call OB 27
1811H	-	Command with illegal opcode	Call OB 29
1812H	-	Illegal opcode extension	Call OB 29
1813H	-	Illegal opcode extension	Call OB 29

Error identifier Accu1	Accu2	Explanation	Reaction of the system in the initial start
1814H	-	Illegal opcode extension	Call OB 29
1815H	-	Illegal opcode extension	Call OB 29
1821H	-	Illegal parameter with CDB 0, 1 and 2	Call OB 30
1822H	-	Invalid timer location number with LT command	Call OB 30
1823H	-	Invalid timer location number with FRT command	Call OB 30
1824H	-	Invalid timer location number with LDT command	Call OB 30
1825H	-	Invalid timer location number with RT command	Call OB 30
1826H	-	Invalid timer location number with SFT command	Call OB 30
1827H	-	Invalid timer location number with SRT command	Call OB 30
1828H	-	Invalid timer location number with SPT command	Call OB 30
1829H	-	Invalid timer location number with SST command	Call OB 30
182AH	-	Invalid timer location number with SET command	Call OB 30
182BH	-	Illegal parameter with JU/JC OB 0	Call OB 30
182CH	-	Illegal parameter with JU/JC OB > 39: special funciton not present	Call OB 30

- Error information with MC5 runtime

Error identifier Accu1	Accu2	Explanation	Reaction of the system in the initial start
1A01H	-	Data block not loaded with CDB	Call OB 19
1A02H	-	Data block not loaded with CXDX	Call OB 19
1A03H	-	Block not loaded JU/JC, FB, OB, PB and SB	Call OB 19
1A04H	-	Block not loaded with BC(C) FX	Call OB 19
1A05H	-	Data block not loaded with OB 254 or 255	Call OB 19
1A11H	-	Transfer error with bit FD on an undefined data word	Call OB 32
1A12H	-	Transfer error with TDR on an undefined data word	Call OB 32
1A13H	-	Transfer error with TDL on an undefined data word	Call OB 32
1A14H	-	Transfer error with TDW on an undefined data word	Call OB 32
1A15H	-	Transfer error with TDD on an undefined data word	Call OB 32
1A21H	-	Special function error with EDB, EXDX: data block already exists	Call OB 31
1A22H	-	Special function error with EDB, EXDX: illegal data block length (< 5 words or > 4 x 2^{10} words	Call OB 31
1A23H	-	Special function error with EDB, EXDX: insufficient memory space in RAM	Call OB 31
1A25H	-	Special function error with BI = illegal parameter in accu 1	Call OB 31
1A27H	-	Special function error with A(: nesting stack overflow	Call OB 31
1A28H	-	Special function error with O(: nesting stack overflow	Call OB 31
1A31H	-	OB 254 or OB 255 data block already present in RAM	Call OB 31

Error identifier Accu1	Accu2	Explanation	Reaction of the system in the initial start
1A32H	-	OB 254 or OB 255 new data block already present	Call OB 31
1A33H	-	OB 254 or OB 255 insufficient memory space in RAM	Call OB 31

- Error information with acknowledgement delay (time out) processing

Error identifier Accu1	Accu2	Explanation	Reaction of the system in the initial start
1E23H	yyyyH	Acknowledgement delay (AKD) with individual I/O access yyyy = AKD address	Call OB 23
1E25H	yyyyH	Acknowledgement delay with process image of digital outputs yyyy = address of the unacknowledged output byte	Call OB 24
1E26H	yyyyH	Acknowledgement delay with process image of digital inputs yyyy = address of the unacknowledged input byte	Call OB 24
1E27H	yyyyH	Acknowledgement delay with process image of IPC flag outputs yyyy = address of the unacknowledged flag byte	Call OB 24
1E28H	yyyyH	Acknowledgement delay with process image of IPC input flags yyyy = address of the unacknowledged flag byte	Call OB 24

Siemens Aktiengesellschaft

The reproduction, transmission or use of this document or its contents is not permitted without express written authority.
Offenders will be liable for damages. All rights, including rights created by patent grant or registration of a utility model or design, are reserved.

© Siemens AG 1986

Order No.. C79000-B8576-C364-02
Order from Gerätewerk Karlsruhe
Printed in the Federal Republic of Germany
AG 0189 0.5 SK 132 En

SIEMENS

SIMATIC S5

Appendix
for Manual CPU 928, CPU 921, CPU 922

Notes C79000-A8576-C260-01

Ordering Information

In this section you will find the order numbers of the products mentioned in the manual. The order numbers are listed according to the parts in which the products are mentioned. Please also refer to the current catalogs.

For Part 2	Order No.
Power supply unit with fan for central controller and expansion units EG-183, EG-185	
230/120 V AC, floating, 5 V, 18 A	6ES5 955-3LC14
230/120 V AC, floating, 5 V, 40 A	6ES5 955-3LF12
24 V DC, non-floating, 5 V, 10 A	6ES5 955-3NA12
24 V DC, floating, 5 V, 18 A	6ES5 955-3NC13
24 V DC, floating, 5 V, 40 A	6ES5 955-3NF11
15-V supplementary module	6ES5 956-0AA12
Power supply unit without fan (only for EG-186U expansion unit)	
230/120 V AC, floating, 5 V, 15 A	6ES5 955-3LB11
DC 24 V, floating, 5 V, 15 A	6ES5 955-5NB11
Fan subassembly (for EG-184 expansion unit)	
240/120 V AC	6ES5 988-3LA11
24 V DC	6ES5 988-3NA11
Load supply 951	6ES5 951-4LB11
Enable supply 958	6ES5 958-4UA11
Accessories	
Air baffle	6ES5 981-0DA11
Dust filter holder	6ES5 981-0FA11
Dust filter (10 off)	6ES5 981-0EA11
Replacement fan	
for 6ES5 955-3LC14	
6ES5 955-3LF12	
6ES5 988-3LA11	6ES5 998-3LB21
for 6ES5 955-3NC13	
6ES5 955-3NA12	
6ES5 955-3NF11	
6ES5 988-3NA11	6ES5 988-3NB11

	Order No.
Fuses (6.3 x 32 mm)	
15 A, slow-blow	299461
6A, quick-blow	300095
4A, quick-blow	291963
Dummy front panels	
1 slot wide	6XF2 008-6KB00
2 slots wide	6XF2 016-6KB00

For Part 3

Central controller

with power supply unit	
6ES5 955-3LC14	6ES5 135-3UA11
6ES5 955-3LF12	6ES5 135-3UA21
6ES5 955-3NC13	6ES5 135-3UA31
6ES5 955-3NA12	6ES5 135-3UA41
6ES5 955-3NF11	6ES5 135-3UA51

Accessories

Back-up battery for power supply plug-ins	6EW1 000-7AA
Battery plug-in	6XG3 400-2AT00
Air baffle	6ES5 981-0DA11

For Parts 4 and 5

EPROM submodule 376

16 x 2^{10} bytes	6ES5 376-0AA11
32 x 2^{10} bytes	6ES5 376-0AA21
64 x 2^{10} bytes	6ES5 376-0AA31

RAM submodule 377

16 x 2^{10} bytes	6ES5 377-0AA11
32 x 2^{10} bytes	6ES5 377-0AA21
64 x 2^{10} bytes	6ES5 377-0AA32
64 x 2^{10} bytes (with back-up battery)	6ES5 377-0BA31
Spare back-up battery for RAM submodule 377	6ES5 950-0AA11

For Part 6

923A Coordinator	6ES5 923-3UA11
Spare parts for 923A coordinator	
Coding plug	C79334-A3011-B12

	Order No.
For Part 7	
923C coordinator	6ES5 923-3UC11
Spare parts for 923C coordinator	
Coding plug	C79334-A3011-B12
Front cover	C79451-A3079-C251
Connecting cable 725	
from 923C coordinator to CP 530, 143 and 5430	
0.9 m	6ES5 725-0AK00
2.5 m	6ES5 725-0BC50
For Part 9	
Standard function blocks	see catalog ST 57 or catalog ST 50