System Description

ABB Procontic T300

Functional Description PLC
35 ZE 94
Version 8

Order No.: GATS 1315 11 R2001

ABB Schalt- und Steuerungstechnik GmbH

ASEA BROWN BOVERI

Preface

Abbreviations

AWP	User program
DBV	Data block directory
MPST	Multi-processor control system for industrial machines
PLC	Programmable logic controller
SBn	Control block No. n
SRQ	Service Request
ZST	Central Control Unit

Definitions

Active station	MPST bus station, which is accessing the MPST bus actively
Passive station	MPST bus station, which is not accessing the MPST bus

Additional capabilities of the 35 ZE 94 in comparison with the 35 ZE 93

Hardware:
- Bus clock adjustable on the front panel
- Station address adjustable on the front panel
- Hardware clock
- Restart switch on the front panel

Software:
- User program memory doubling to 120 kbytes
- Changes can be made to the running PLC program
- Function block: Square root
- Function block: Positioning of NC axes
- Function block: Clock
- Operator control functions on the keyboard for the clock
- Additional monitor functions
- Additional remanent flags

Table of contents

1	**General**	1-1
2	**Performance features**	2-1
3	**Hardware and commissioning**	3-1
3.1	As delivered condition and notes on operation	3-1
3.2	Initialization by cold start	3-1
3.3	Commissioning with 35 LB 90 battery module	3-2
3.4	Battery replacement	3-2
3.5	Functions of the LEDs on the front panel	3-2
3.6	Restart key	3-2
3.7	Setting of address, bus clock	3-4
3.8	Small program memory with comment RAM	3-4
3.9	Small program memory with program EPROM/comment EPROM	3-5
3.10	Location of the connectors	3-5
3.11	Expansion by two serial interfaces using the 35 DS 91	3-6
3.12	Position of the modules in the module rack	3-8
4	**Programming**	4-1
4.1	Programming aids	4-1
4.2	Procedure for creating a program	4-1
4.3	Input/modification of a program	4-2
4.3.1	Input/modification of a program in the "aborted" state	4-2
4.3.2	Input/modification of a program in the "running state"	4-2
4.4	EPROM programming	4-3
4.4.1	Programming the EPROMs for the large memory configuration (120 kbyte)	4-3
4.4.2	Programming the EPROMS for the small memory configuration (56 kbyte)	4-5
5	**PLC operating modes**	5-1
5.1	Explanation of the individual operating modes	5-2
5.2	Defaults	5-5
5.3	Changes to the operating modes by the user	5-5
6	**Language repertoire**	6-1
6.1	Operators for planning sentences (DIN 19239)	6-1
6.2	Block calls	6-2
6.3	NOP	6-2
6.4	Labels	6-2
6.5	Program end	6-2
6.6	Data types/formats	6-2
6.7	Operand identifier	6-3
6.8	Numbering of operands (variables)	6-3
6.9	Time values	6-4
6.10	Text processing	6-4
6.11	Step chains	6-6
6.11.1	Functionality of the step chain	6-6
6.11.2	Joining step chains	6-6
6.12	Syntax diagrams of the PLC language	6-7
6.12.1	Syntax diagram: BOOLEAN SENTENCE	6-7
6.12.2	Syntax diagram: ARITHMETIC SENTENCE	6-8
6.12.3	Syntax diagram: HYBRID SENTENCE	6-9
6.12.4	Global syntax diagram: blocks	6-10
6.12.5	Syntax diagram: PLC program	6-11
7	**Comment processing**	7-1
7.1	Prerequisites	7-1
7.2	Comment inputs	7-1
7.2.1	Input format	7-1
7.2.2	Display format with command: D	7-1
7.2.3	Display format with command: S	7-1
7.2.4	Deleting a comment	7-1
7.2.5	Deleting all comments in a PLC program	7-2
7.2.6	Programming of comment EPROMs	7-2
8	**Operating and test functions**	8-1
8.1	Commands for creating user program	8-3
8.2	Commands for testing the user program	8-11
8.3	Commands for configurating the PLC	8-19
9	**Data block directory (DBV)**	9-1
9.1	Functions of the data block directory	9-1
9.1.1	Default data block directory	9-1
9.1.2	Modification of the default data directory	9-1
9.1.3	Resetting the data block directory to the default configuration	9-1
9.2	Location and structure of the data block directory in the transfer memory	9-2
9.2.1	Start of the data block directory in the transfer memory	9-2
9.2.2	Pointers to physical devices	9-2

10 **Communication with I/O modules and preprocessors** 10-1
10.1 Communication with I/O modules ... 10-1
10.2 Default jumpering of the I/O signals and I/O modules 10-1
10.3 Preprocessors 10-4
10.3.1 Process display and control 10-4
10.3.2 Industrial computer 35 IR 93 10-16
10.3.3 Coupling: ABB Procontic T300 to the ABB Master process control system. Communication processor 35 KP 91 10-18
10.3.4 Coupling: ABB Procontic T300 to VERITRON PAD power converters. Communication processor 35 KP 92 10-20
10.3.5 Positioning modules for axes 10-22
10.3.6 Coupling: ABB Procontic T300 to field bus ZB 10. Coupler 35 ZB 91 . 10-25
10.3.7 Video sensor system, OMS-F, modules 35 IV 90 and 35 KI 90 10-26

11 **Communication between the two PLC programs** 11-1

12 **Multiprocessor operation** 12-1
12.1 Planning of superglobal values 12-1
12.2 Addressing binary superglobal groups 12-2

13 **TURBO operation** 13-1
13.1 TURBO concept 13-1
13.2 Testing the PLC program 13-1
13.3 Advantages of this procedure 13-1
13.4 Execution time data 13-2

14 **Operation of the PLC via the MPST bus** 14-1
14.1 Communication path between the PLC processors 14-1
14.2 Description of the logical interface in the transfer RAM 14-1

15 **Serial interfaces** 15-1
15.1 Basic initialization: operator interface and interface 1 15-1
15.1.1 Additional 35 DS 91 module 15-1
15.1.2 Interface initialization by the user ... 15-1
15.1.3 Port addresses of the serial interfaces 15-1
15.2 Synchronization 15-1
15.3 Setting the baud rates 15-2
15.4 Pin assignment 15-2
15.5 INTEL HEX file for creating the user program or comment EPROMs 15-2
15.6 Connection of an additional 35 DS 91 interface module 15-3
15.7 Operation of the interfaces by the PLC 15-3
15.8 Interface conversion from RS-232-C to RS-422 15-5

16 **PLC controlled by a central control unit (ZST)** 16-1
16.1 Initialization of the PLC 16-1
16.2 Assigning jobs to programs 16-4
16.2.1 Definition of the control blocks 16-4
16.2.2 Realizing job assignment 16-6

17 **Starting behaviour** 17-1

18 **Self-diagnosis and reactions to errors** 18-1
18.1 Error messages 18-1
18.2 Error types and reactions 18-1

19 **Monitor functions** 19-1

20 **Memory subdivision** 20-1
20.1 Summary of the PLC's complete address area 20-1
20.2 Detailed summary of the user program memory 20-2
20.2.1 Subdivisions if the program memory size is 56 kbytes 20-2
20.2.2 Subdivisions if the program memory size is 120 kbytes 20-3
20.3 Detailed summary of the operand memory 20-4
20.4 Detailed summary of the transfer RAM 20-5
20.5 Detailed summary of the standard peripheral addresses 20-5

21 **Extension of old systems and upward compatibility** 21-1

Appendix: Catalog of Blocks

1 General

The ABB Procontic T300 is a programmable logic controller in accordance with DIN 66264 which is capable of multi-processor operation (multi-processor control system for industrial machines). The system has a module structure and embraces function modules such as

- a programmable logic control (PLC)
- an industrial computer
- communication processors

The PLC can be used as

- an autonomous PLC or as
- function block in accordance with DIN 66264

Application range

- Binary processing
- Word processing
- Double word processing
- Closed-loop control engineering including adaptive control
- Control of NC axes
- Video sensor systems
- Special functions

Expandable

- Thanks to its multi-processor abilities and modular structure, an MPST system is capable of easy expansion at all times. For this purpose, a large number of both intelligent and passive modules is available to the user.

Capable of communication

- Direct communication with a
  - terminal
  - printer
  - EPROM programmer
  - personal computer
  - host computer (serial or parallel)
  - telephone modem

  is enabled by a simple ASCII plain language protocol and standard interfaces.

User-friendly

- Transparent program structure by means of a
  - block structure (conventional notation expressed in equations) and
  - module structure
- 2 mutually independent user programs
- Convenient ABB Procontic programming and test software executable in a personal computer
  - instruction list
  - ladder diagram
  - function block diagram
  - extensive documentation and listings
- Programming and test functions integrated in the PLC enable
  - programming (IL) and user program testing on a standard ASCII terminal
  - direct connection of a EPROM programmer
  - direct connection of a printer
- Storage of both user programs in the battery-backed RAM and in the EPROM
- The PLC's I/O designators can be rejumpered to any chosen addresses by simply keying on the terminal. Jumpering is possible to:
  - I/O modules or other passive modules communicating through the MPST BUS
  - other PLC processors
  - other active modules communicating through the MPST BUS
  - couplers

Self-diagnosis

- Extensive diagnostic measures when
  - starting the PLC
  - during the PLC operation
- Automatic shutdown of the PLC and resetting of outputs in the event of fatal errors.
- Display of errors by
  - LEDs on the front panel
  - plain language on the monitor of a terminal
  - entry in an error register which can be read out both through the serial interface and also through the MPST BUS

Hardware

- The PLC's central unit consists of the modules:
  - 35 ZP 93 processor board
  - 35 DS 91 battery-backed memory board with 2 serial interfaces.

2 Performance features

Scope of functions

- Logical operations AND, OR ...
- Step chains
- Integer arithmetics +, −, :, *
- Brackets, nesting depth 15
- Conditional branch
- Timers with real-time presetting
- Counting functions
- Controller functions
- Module processing (see also appendix)
- Video sensor system

Timers

- 80 simultaneously active timers
- Time presetting in ms
- Time range 5 ms ... 24.8 days

Program processing

- 2 mutually independent user programs
- Each user program event- or time-controlled. Minimum presettable cycle time is 5 ms.
- Creation of an image of the input and output signals

Program length

- Large memory capacity (120 kbytes):
  - In total: 61170 words corresponding to 30585 instructions
  - Per program: 30585 words corresponding to 15292 instructions
- Small memory capacity (56 kbytes):
  - In total: 28640 words corresponding to 14320 instructions
  - Per program: 14320 words corresponding to 7160 instructions

Processing time

- BINARY OPERATION < 2.3 ms/k instructions
- ADDITION < 5 ms/k instructions
- SUBTRACTION < 5 ms/k instructions
- MULTIPLICATION < 23 ms/k instructions
- DIVISION < 24 ms/k instructions
- ALLOCATION < 5 ms/k instructions
- COMPARISON < 5 ms/k instructions

RAM back-up

- User program RAM
- Flag and image memory
- Comment memory

Comment memory

- 64 kByte backed RAM/EPROM per program. Only for small program memory configuration

Flags

Per user program there are:
- BINARY flags: 4126
- WORD flags: 4144
- DOUBLE WORD flags: 544

Steps

Per user program there are:
- BINARY steps: 2048

Inputs/outputs

Per PLC there are:
- BINARY inputs: 1024
- BINARY outputs: 1024
- ANALOG (WORD) inputs: 256
- ANALOG (WORD) outputs: 256

Global values/super global values

Per PLC there are:
- BINARY global values: 128
  for communication between the two user programs of one PLC
- BINARY super global values: 400
  for communication between several PLC processors
- WORD global values: 32
  for communication between the two user programs of one PLC
- WORD super global values: 80
  for communication between several PLC processors

Constants

Per user program there are:
- BINARY indirect constants: 2
- WORD indirect constants: 640
  in SENSOR mode additionally 32672 word constants for the user program 2
- DOUBLE WORD indirect constants: 128

Self-diagnosis

- EPROM test
- RAM test
- Power fail
- PLC overload, i.e., the preset cycle time cannot be obeyed

- Unknown operator/module found during program processing
- More timers simultaneously needed than available
- Syntax error in the user program
- No user program loaded on program start
- No cycle time preset for a time-controlled user program

Error messages

Detected errors are reported

- In plain language on the monitor of a terminal connected to the PLC
- By LEDs on the front panel
- By entry of an error number in an error register. This error register can be read out both through the MPST bus and also through the serial interface.

Coupling possibilities

- To other ABB control systems through a serial real-time bus (ZB10).
- To computers, modems, terminals, printers, display terminals and PROM programmers, etc., through a serial interface.
- To ABB industrial computers, to the central control unit ZST, to axis positioning units and communication processors through the MPST bus
- To SIEMENS PLCs via coupling board

3 Hardware and commissioning

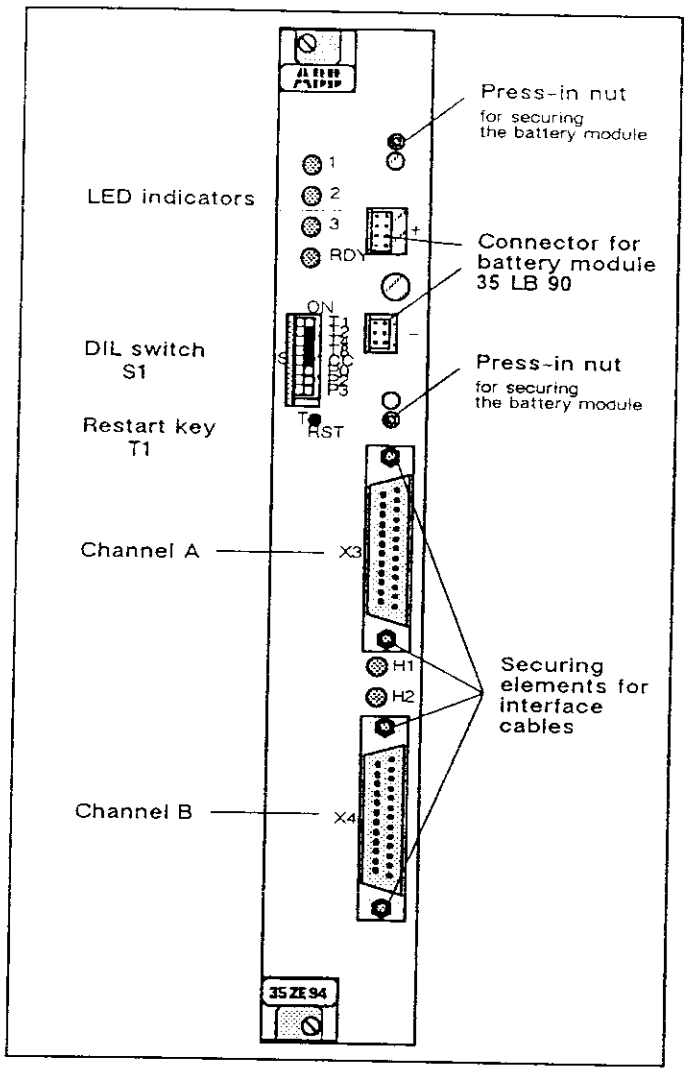


Figure: Front view of the 35 ZE 94 R101

3.1 As-delivered condition and notes on operation

The 35 ZE 94 consists of two pc boards, the interface board (35 DS 91) and the processor board (35 ZP 93).

The **baud rates** of the operator interface (connector X3) and of interface 1 (connector X4) are set to 9600 baud. The serial interface on the ABB Procontic programming and test unit or the VT100 compatible terminal must also be set to 9600 baud. (See chapter entitled "Serial interfaces".)

The **station address** on the MPST bus is set to "01_H" with S1 (see section 3.7).

☞ Check whether a different central control unit already has the set station address. If this is the case, a different, unassigned station address must be set with the switches of T1, T2, T4, T8 by means of the 8 bit DIL switch.

The **MPST bus clock** is output to the MPST bus from *one* module.

☞ Every subrack must be supplied exactly *once* with the MPST bus clock. The CC bus clock is activated by means of the S1 DIL switch, switch 5 = ON. Under no circumstances must the bus clock be output to the MPST bus for two or more modules.

The S1 DIL switch is set to:

☞ the station address	T1, T2, T4, T8
☞ the bus clock	CC
☞ unused	P0, P2, P3

Sockets XA12, XA13 of the module are equipped with 64k * 8 EPROM for the large program memory.

☞ Before changing the components, e.g., configuration of the small program memory, the jumper settings must be changed as shown in the following tables.

☞ Always back up your programs when you have to remove the module from the subrack in order to change the settings on the 35 ZE 94, for example.

☞ As a prerequisite for commissioning the 35 ZE 94, the electrical and mechanical installation guidelines given in section 5, "Planning", must be observed.

☞ If possible, the interface connectors must not be removed while the power supply is activated.

3.2 Initialization by cold start

In all cases, a **cold start** must be executed:

☞ before using the 35 ZE 94 for the first time

☞ after the 35 ZE 94 has been plugged into the subrack, e.g., after replacing the programming memory

☞ after memory components on the 35 DS 91 have been touched.

3.3 Commissioning with 35 LB 90 battery module

In the "as-delivered" state the included 35 LB 90 battery module is not fitted. This avoids placing a burden on the lithium battery before commissioning.

The module must be fitted in a T300 subrack before plugging in the battery module. Then activate the power supply for the subrack.

The gold capacitor on the module is charged up after approximately 5 to 10 minutes.
The 35 LB 90 must be fitted with the included screws.

☞ Execute a **cold start** to establish a defined operating state (see the chapter entitled "Starting response").

3.4 Battery replacement

The battery or the battery module can be replaced while the subrack supply voltage is on.

If the subrack supply voltage is off when replacing the battery, the connection of X36:1-2 must be established by a jumper on the interface board ("as-delivered" state). A gold capacitor then covers the missing battery voltage for a period of more than 15 minutes. The 35 ZE 94 must be fitted in the subrack to ensure that no data destruction will occur.

3.5 Functions of the LEDs on the front panel

The LEDs of 1, 2, 3, RDY, H1 and H2 are located on the front panel of the PLC.

Functions of the LEDs:

1: Run indication for PLC programs

LED 1 is activated at the start of each program cycle and is deactivated again at the end of it. This is why, if PLC programs are short, LED 1 only lights up for a short period during each program cycle.

2: POWER FAIL

LED 2 indicates that the POWER FAIL signal has responded. LED 3 is additionally activated.

3: Error indication

LED 3 basically indicates that an error has occurred (see also the chapter entitled "Self-diagnosis and reactions to errors").

RDY: READY

The RDY LED is connected directly to the READY pin of the processor. This LED must always light up after the voltage has been switched on.

"Running" light of LED 1 ... LED 3

If the PLC discovers defective memory cells during a cold start, the PLC initialization is aborted and the error is additionally signalled by a running light on the LEDs of 1 ... 3.

H1 not used

H2 not used

3.6 Restart key

The restart key is recessed in the front panel. The 35 ZE 94 can be restarted using a switching pin (or also a screwdriver).

☞ Only the 35 ZE 94 is restarted when using the Restart key. All other modules are not influenced.
It must be guaranteed by planning, that outputs, preprocessors and other modules are not set to undefined states.

☞ In multi-processor mode, the Restart key must not be used on the 35 ZE 94 configured as the master PLC because this interrupts all communication.

X3	1-8 2-30 6-13-33 4-7-34 3-15 5-24	With sub-board	GJP5 122910 R1
X7	1-2	Minimum memory access time (microprocessor 8086 without wait state)	
X10	No jumper	MPST station address in the active station range (MPST signal I/O = 1)	

Table: "As-delivered" jumpers on the processor board

X8	5-6	Channel A baud rate 9600 bd	= operator interface
X10	5-6	Channel B baud rate 9600 bd	= serial interface 1
X11	Spare jumper	Special screen 0 V	Connections not necessary
X12	Spare jumper	Special screen 0 V	Connections not necessary
X15	No jumper		
X16	No jumper		
X17	1-2 5-6	35 ZE 94 Clock activated	
X18	No jumper	Address setting for socket XA10-XA13	
X19	1-2.3-4.5-6.7-8	Binary input for bit register	
X20	1-2 5-6 9-10 17-18	XA10. XA11 32k * 8 RAM	Large program memory capacity
X21	3-4 7-8 15-16 19-20	XA12.XA13 64k * 8 EPROM	Large program memory capacity
X22	1-2 5-6	RAM 62256 on XA6-XA9. battery-backed RAM 62256 on XA0-XA5. battery-backed	
X24	No jumper	Chip select to Ground connection for battery-backed RAM	
X27	No jumper	Battery voltage alternatively to 35 LB 90 from the MPST bus	
X28	1-2	Oscillator for channel A/B only for test purposes of the producer	
X32	7-8	XA10.XA11 RAM 32k*8	Address A0000..AFFFF
X33	7-8	XA10.XA11 RAM 32k*8	Address A0000..AFFFF
X35	No jumper		
X36	1-2	Disconnection for the back-up capacitor	
X38	1-2	Chip select for RAM/EPROM XA10. XA11	
X39	1-2	Chip select for RAM/EPROM XA12. XA13	

Table: "As-delivered" jumpers on the interface board

3.7 Setting of address, bus clock

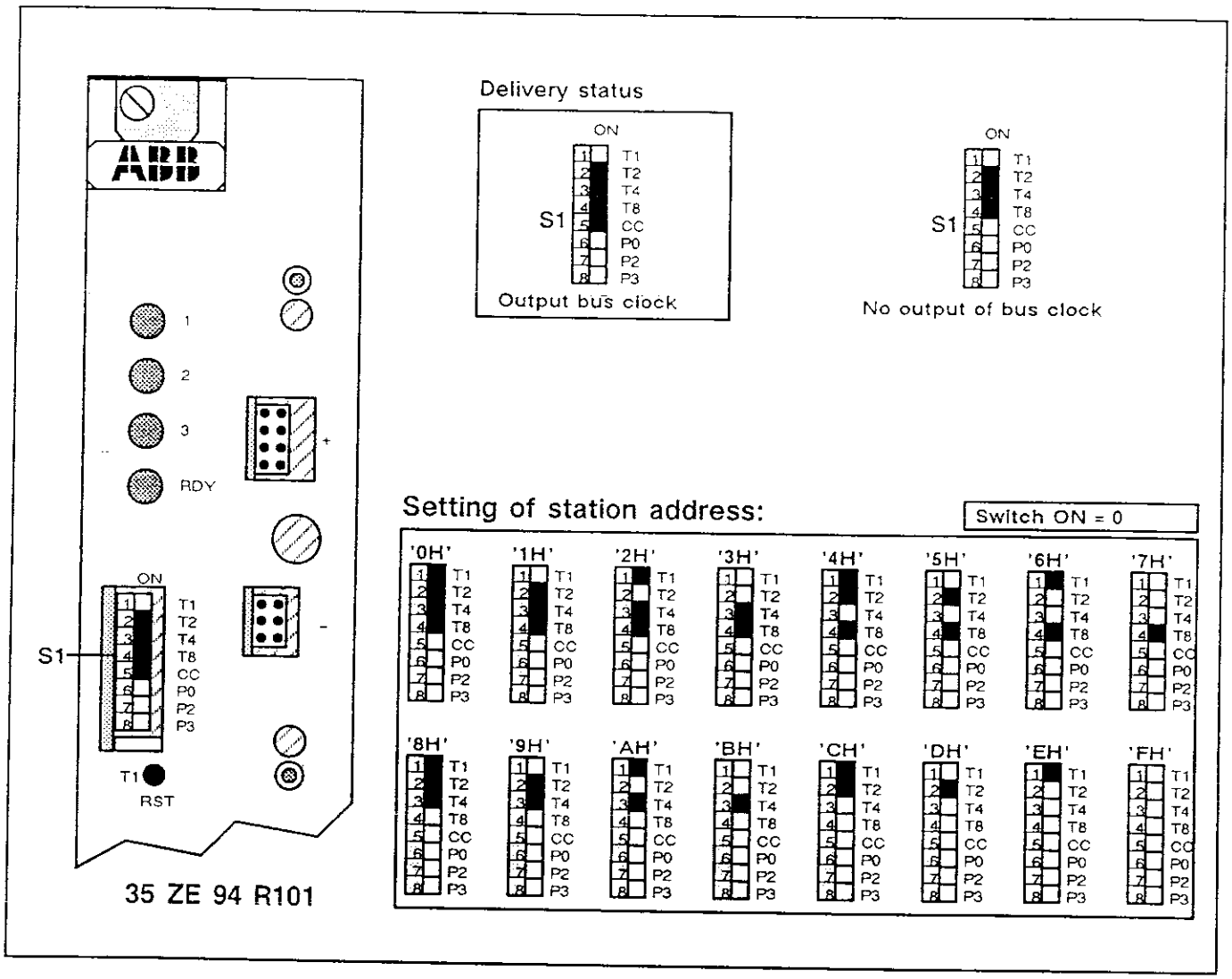


Figure: Setting of station address and bus clock

3.8 Small program memory with comment RAM

The jumpers on connectors X20, X21, X32 and X33 must be set as specified in the following table. All other connectors are left unchanged.

X18	3-4	XA10.XA11 RAM 32k*8	Address B0000...BFFFF
X20	1-2 5-6 9-10 17-18	XA10. XA11 RAM memory 32 k * 8	
X21	1-2 5-6 9-10 17-18	XA12.XA13 RAM memory 32 k * 8	
X32	1-2 5-6	XA12.XA13 RAM 32k*8 XA10.XA11 RAM 32k*8	Address C0000...CFFFF Address B0000...BFFFF
X33	1-2 5-6	XA12.XA13 RAM 32k*8 XA10.XA11 RAM 32k*8	Address C0000...CFFFF Address B0000...BFFFF

Table: Jumpers for small program memory with comment RAM

3.9 Small program memory with program EPROM / comment EPROM

The jumpers on connectors X20, X21, X32 and X33 must be set as specified in the following table. All other connectors are left unchanged.

X18	No jumper	XA10.XA11 EPROM 32k*8 Address A0000...AFFFF
X20	3-4 7-8 11-12 15-16	XA10, XA11 Memory EPROM 32k*8
X21	3-4 7-8 15-16 19-20	XA12.XA13 Memory EPROM 64k*8
X32	2-4	Spare jumper
	3-5	Spare jumper
X33	2-4	Spare jumper
	3-5	Spare jumper

Table: Jumpers for the small program memory with program EPROM / comment EPROM

3.10 Location of the connectors

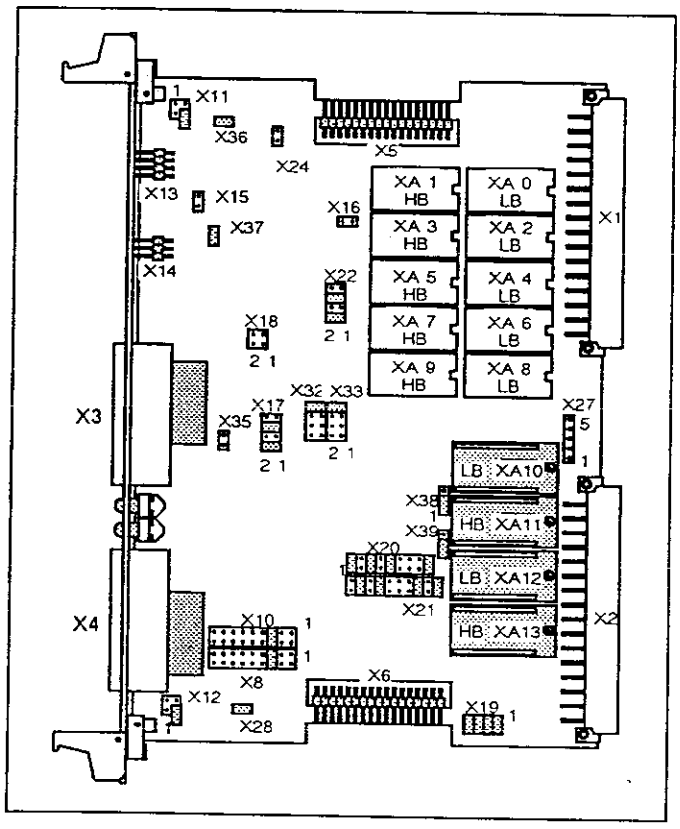


Figure: Interface board

Before the components of memory locations XA10 ... XA13 are changed on the 35 DS 91, the jumpers must be set according to the required component configuration.
A "cold start" must be executed after changing these settings.

☞ The pc board must not come into contact with conductive tools or support surfaces, etc., if the battery module (35 LB 90) is fitted.

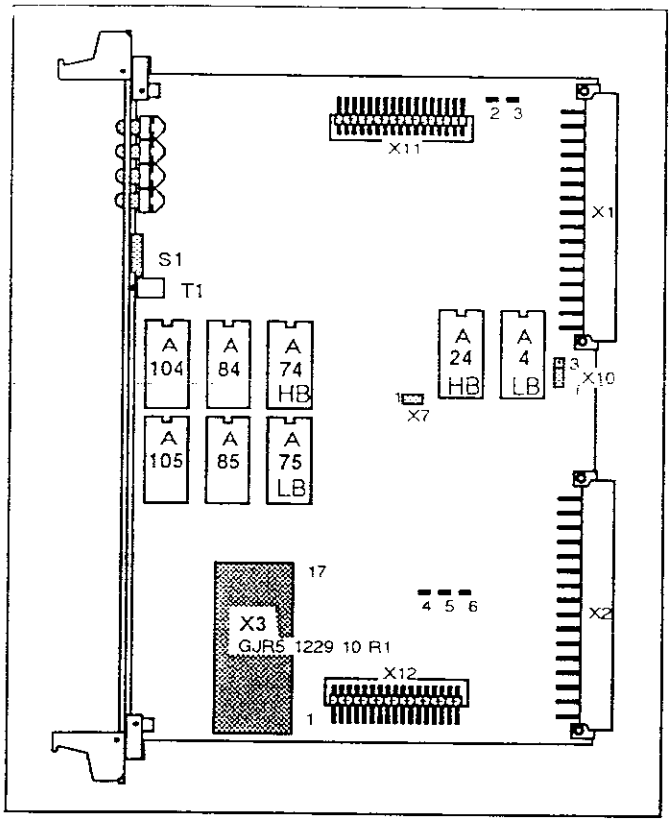


Figure: Processor board

Pins 2, 3, 4, 5 and 6 are intended exclusively for test purposes of the producer.

☞ X10: 1-2 means that the unit operates in the passive station range. This setting must **not** be used with the 35 ZE 94.

3.11 Expansion by two serial interfaces using the 35 DS 91

The 35 ZE 94 module can be expanded by two serial interfaces. A 35 DS 91 module is fitted to the 35 ZE 94 for this purpose. The 35 DS 91 R2 has a separate front panel. The modules are fitted using the included screws and connection parts.

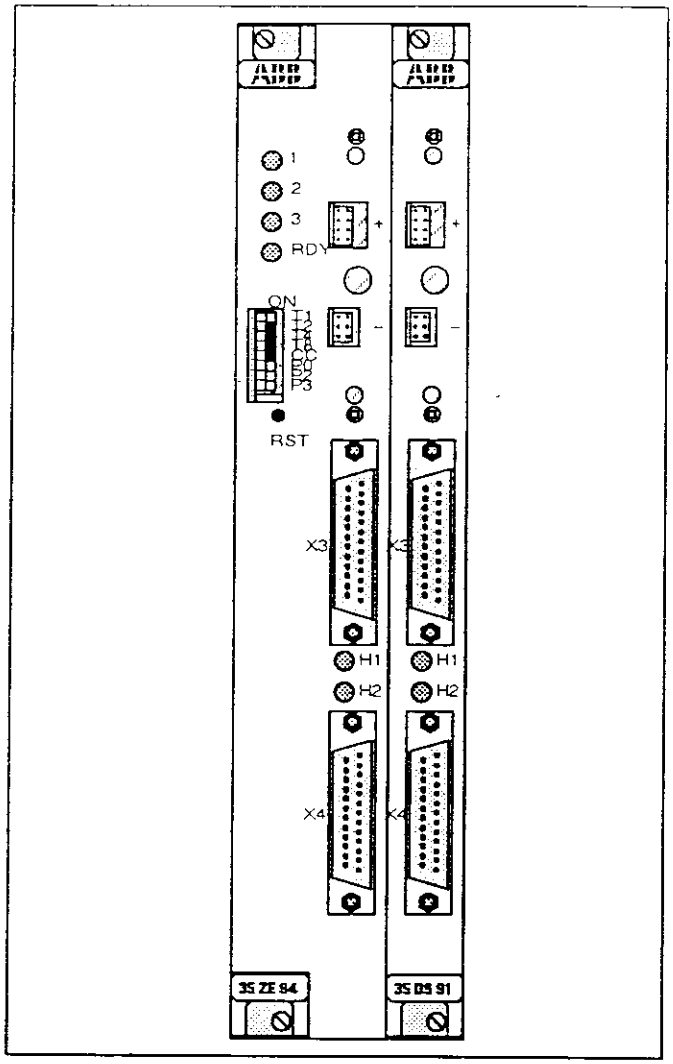


Figure: 35 ZE 94 R101 expanded with 35 DS 91

Settings:

Jumper X17:3-4 must be set on the 35 DS 91 R2 expansion module.

☞ Jumpers X17:5-6 and X17:1-2 must be removed.

All other settings, e.g., baud rate and special screening measures, etc., can be found in the description of the 35 DS 91 R2.

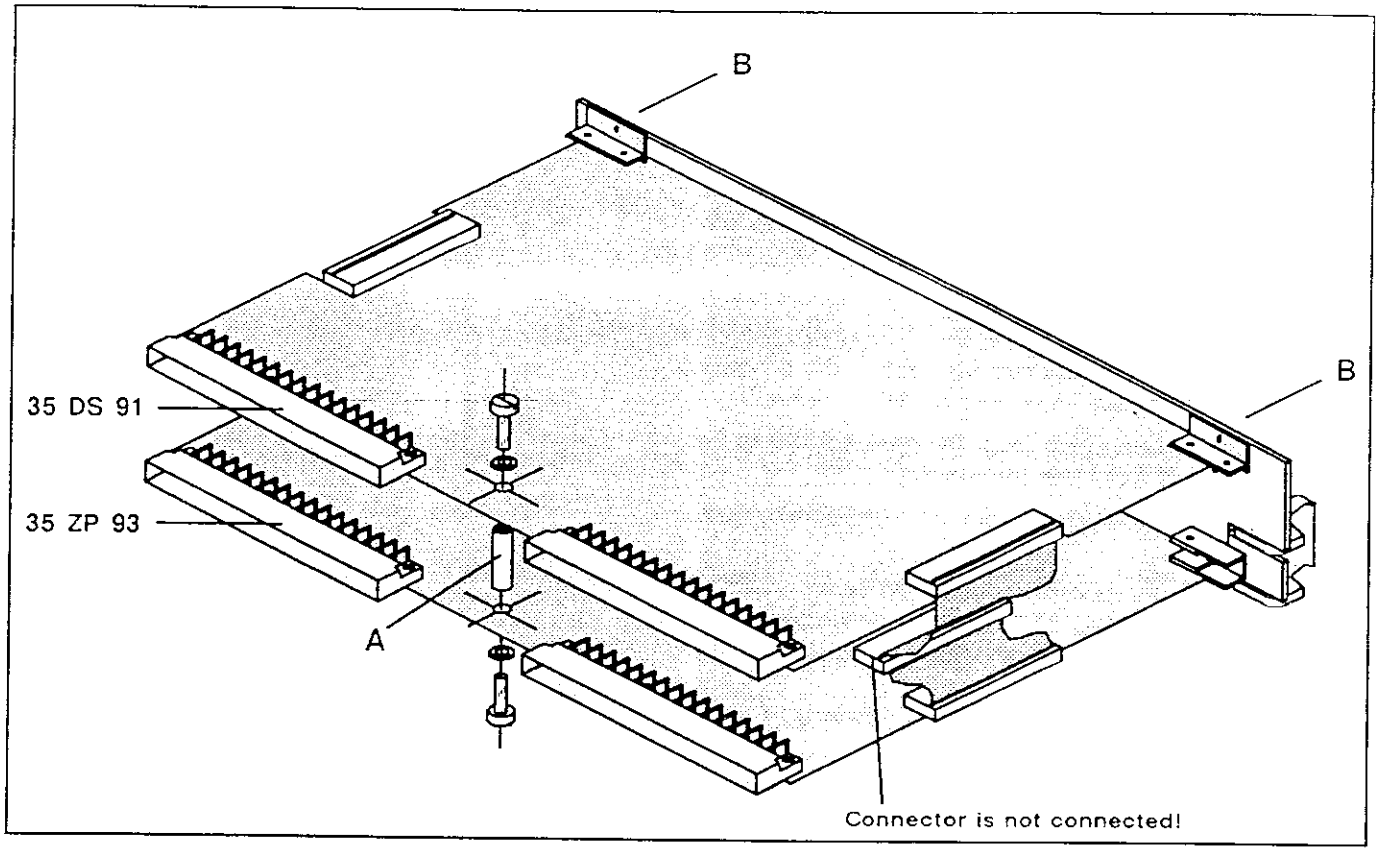


Figure: Fitting the 35 ZE 94 consisting of the pc boards 35 ZP 93 and 35 DS 91

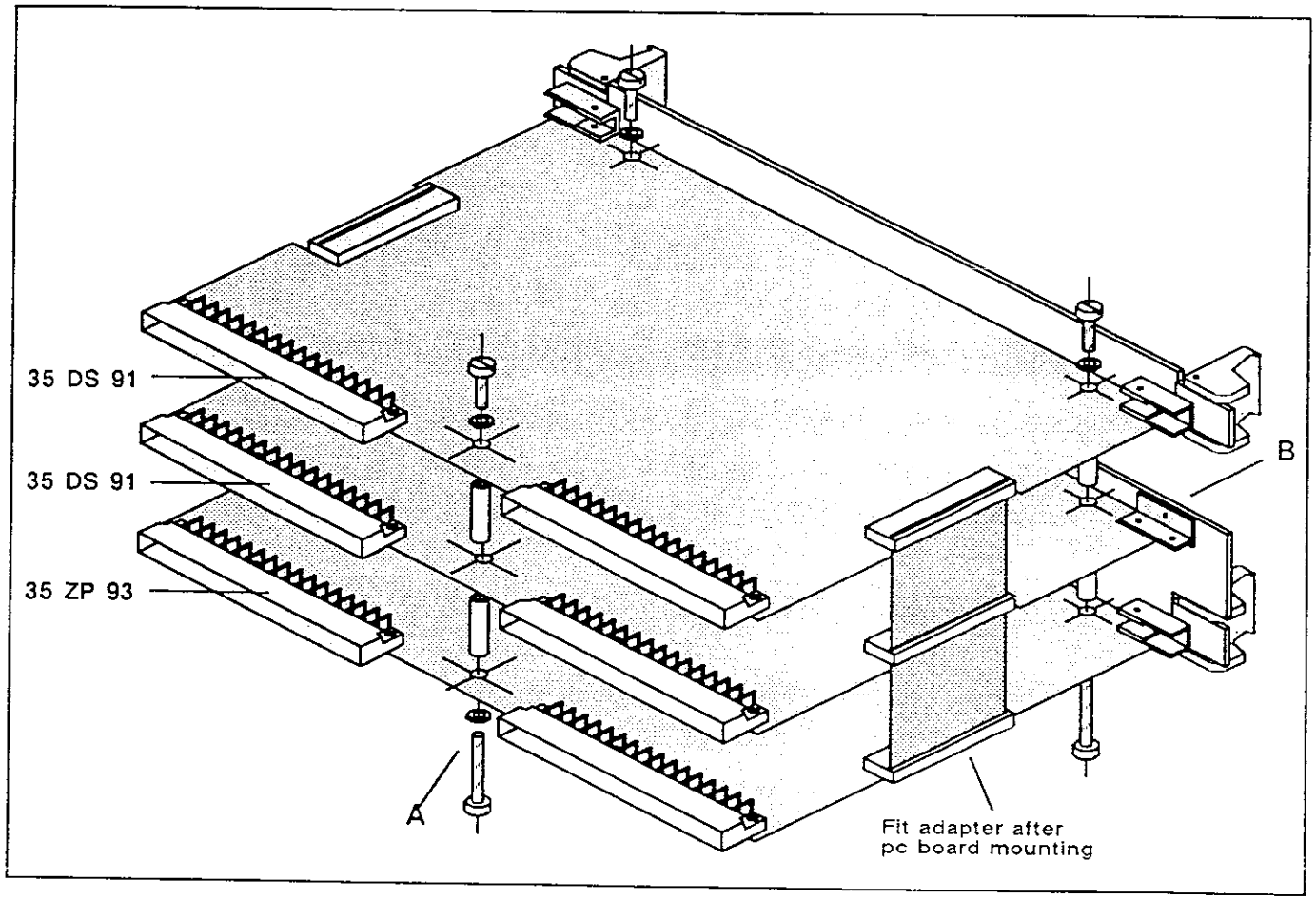


Figure: Assembling 35 ZE 94 and 35 DS 91

3.12 Position of the modules in the module rack

Following aspects are valid for the functionality and access priority of the module at the multi processor bus:

- ☞ Highest access priority for the multi–processor bus has that module, which is positioned in the module rack at the far right side.
- ☞ The access priority decreases, if the module is positioned more to the left.
- ☞ The assignment of the module rack in the processors is carried out *from right to left.*
- ☞ The passive bus users (e.g. I/O devices) are to be arranged left from the processors.
- ☞ Between the individual units no unassigned slots may be left.
- ☞ The 35 KP 92 communication processor, which serves for coupling of the power converter VERITRON PAD, must be the arranged as unit on the left side.
  The 35 KP 92 unit is a processor, however, it does not access the multi–processor bus.

4 Programming

The PLC can be programmed in:

- Instruction list (IL)
- Function block diagram (FBD)
- Ladder diagram (LD)

Instruction list programming conforms to DIN 19239. This DIN standard specifies the structure of the instructions and of the module. The user program is entered into the PLC either directly via the top serial interface connector of the 35 ZE 94 unit or through the MPST bus. The instruction list's ASCII characters are transferred directly to the PLC. During transfer, the programming and test software integrated in the PLC translates the instruction list into the notation required for execution (intermediate code) and stores this in the PLC user program RAM.

The exposed ASCII plain language interface enables access to the PLC by means of instruments offering different degrees of convenience. This ranges from a simple ASCII keyboard with digit display via a terminal to the convenient programming computer.

If the small program memory capacity (56 kbytes) is configured, a 64 kbyte comment memory is also available for each of the two user programs in the PLC.

4.1 Programming aids

The programming aids of the PLC can be subdivided as follows:

- Aids **without** intelligence for programming in instruction list (IL)
  - Commercially available ASCII keyboards with a digital display
  - Commercially available terminal, e.g., VT100
  - Personal computer operated as a terminal
- Aids **with** intelligence
  - ABB Procontic programming system
    (refer to the ABB Procontic programming system for details of the instruction list, function block diagram and ladder diagram)

When using the programming aids **without** intelligence, the PLC program is entered directly into the PLC in the instruction list notation.

When using the programming aids **with** intelligence, the PLC program is written either as

- Instruction list
- Function block diagram or
- Ladder diagram

into the personal computer and stored in this computer on a mass storage medium. The function block diagram/ladder diagram is then translated to the instruction list by the programming system and transferred to the PLC.

The terminal or personal computer is connected to the operator interface (top interface) of the PLC (see also the chapter on the serial interfaces).

4.2 Procedure for creating a program

The creation of an executable PLC program can be subdivided as follows:

- Enter the PLC program and terminate it with the program end identifier (PLC command of S or use the ABB Procontic programming system).
- Enter the values for the indirect constants (PLC command of K or use the ABB Procontic programming system).
- Input of the cycle time:
  The cycle time for the PLC program is given by the double word constant of KD 0,0. The cycle time is specified in milliseconds (ms). Only integral multiples of 5 ms are allowed as numerical values (PLC command of K or use the ABB Procontic programming system).

  No cycle time is entered for event-controlled user programs, since it is not evaluated by the PLC in this case.
- Set the modes (PLC command of KONFS or use the ABB Procontic programming system).
- Set the data block directory (DBV) *only* if it deviates from the default (PLC command of DBV or use the ABB Procontic programming system).
- Create a program EPROM if required (command of PU).
- Commission and test the PLC program (PLC commands for the test functions or use the ABB Procontic programming system).

4.3 Input/modification of a program

A PLC program can be entered or modified through the operator interface in the following states:

- "Aborted" state
- "Running" state

4.3.1 Input/modification of a program in the "aborted" state

Unlimited program changes can be made in the "aborted" state. After the program modification has been made, the program must be started in the INI OPS operating mode (initialization of the operand memory). This ensures that all operands and historical values will be initialized to "0" and that battery-backed values will not lead to malfunctions.

When both programs of the PLC are in use, *one* program can be modified with the *other* is running.

4.3.2 Input/modification of a program in the "running" state

- Modifications to a *running* PLC program are possible both in the NORMAL and in the TURBO operating modes.
- *No* PLC standstill time occurs when adopting the changes.
- The user must pay *extreme caution* when changing a *running* PLC program because the program changes affect the controlled process directly without the possibility of performing tests. Programming errors can have serious consequences.

4.3.2.1 Allowed program changes

- *Any* program changes are allowed as long as no *function modules containing historical values* are affected.
- *Function modules containing historical values* may only be added to the *end of the program* or dropped.
- *Any* changes to indirect constants are allowed. This also includes changing the cycle time.

4.3.2.2 Forbidden program changes

- It is forbidden to delete or add function modules containing historical values *within* the PLC program.

4.3.2.3 Marginal conditions

The following operating modes must be set in order to make changes to a *running* PLC program:

- CHANGE RUN, enabling changes to the running program.
  Only *one* PLC program is available to the user in this mode. The second program memory is needed by the PLC's operating system to execute the changes on the *running* program. This mode can only be terminated by means of a cold start.
- ZEITZYKL., for time-cyclical program running
- NO SENSOR, no video sensor operation
- 120 KB, 120 kbytes program memory length

4.3.2.4 Procedure for changing a running PLC program

- Establish the marginal conditions specified in Section 4.3.2.3.

Terminal operation:

- Enter the AEND command; the PLC is now ready to accept the program changes
- Execute program and constant changes
- Enter the FREI command to release the program changes for processing

ABB Procontic programming system:

- Perform changes
  Transfer changes
  Send changes
  Enable changes

4.3.2.5 Explanation of the AEND and FREI commands

- **AEND command**

This command informs the PLC that changes are to be made to the running PLC program. The PLC creates a copy of the running program in the unused program memory. The PLC is now ready to make the changes required by the user on this copy.
The following commands for program processing and operation of the PLC are permissible after input of the AEND command:
AL, CROSS, D, DRE, DRA, F, IDA, IDR, IDS, K, N, NOP, O, P, PA, PU, S, SO, V, CTRL W, FEHLER, LED, INIT, $.

- **Rejecting changes**

All changes previously made to the program are rejected by entering the AEND command once again. The original program is again available as the basis in order to make new program changes.

- **Rejecting the AEND command and changes**

In reference to the running program, the following commands become active **and** additionally reject the AEND command **and** thus all program changes made since input of the AEND command:

A, BA, BR, BS, EA, EAA, ES, ESA, EZ, EZA, FORC, FORC A, FORC R, G, L, PS, ST, TRACE, TRACE E, W, Y. The command AEND must be entered again if you wish to again make program changes.

- **FREI command**

The changes made to a running PLC program after input of the AEND command are enabled for execution by the PLC.
The program changes made are not yet executed by the PLC **before** input of the FREI command.
The changes made are executed by the PLC **after** input of the FREI command.

4.3.2.7 Rejecting an enabled change to a running PLC program and re-activation of the previous program state

ALT command
The previous program state can be re-activated by means of the ALT command. That is to say that the changes made to a PLC program **and** enabled are rejected again. Additionally, the PLC restores the previous program state. This is the state of the program that prevailed in the PLC **before** the program change, i.e., before input of the AEND command.

Following input of the ALT command, the previous program state is re-activated within an approx. 1 ms without the need for any further actions on the part of the user.
This command can be used if the user recognizes that the program changes made have not had the desired degree of success.

4.4 EPROM programming

The user programs and commands loaded into the PLC user program RAM can be stored in EPROMs. An EPROM programmer connected to interface 1 (bottom interface) is needed for this purpose. The user programs or comments are transferred from the PLC to the EPROM programmer by means of the following operating command:

PU <CR> (see chapter 8)

By this command the user programs or comments are output as an INTEL HEX file through the bottom serial interface and are transferred to the EPROM programmer. The command of PU may be aborted at any time by entering <Ctrl>C. EPROMs are programmed by means of the PU command in a dialogue between the user and the PLC.
The following EPROM types are needed depending on the configured memory size:

- Large memory configuration (120 kbytes)
  Program: 2 EPROMs, type: 27C512
  Comment: Storage in the PLC is not possible
- Small memory configuration (56 kbytes)
  Program: 2 EPROMs, type: 27C256
  Comment: 2 EPROMs, type: 27C512

Note:
The **complete** information belonging to the PLC program is also stored in the EPROMs, i.e.:

- the PLC program
- the configured modes (KONFS)
- the data block directory (DBV)

Please make sure that the required modes and the data block directory are set correctly **before** programming the EPROMs.

Important!

Owing to the sensitivity of battery-backed CMOS RAMs in relation to data security in the event of equipotential bonding currents, attention must be paid to the following points:

- As a safety measure, a cold start must be executed (command of KALT <CR>; see also the chapter entitled "Starting response") after each of the following operations on the PLC:
  - Inserting the PLC in the subrack
  - Inserting or removing the user program EPROMs
  - Inserting or removing the comment EPROMs
  - Inserting or removing the comment RAMs

A cold start destroys the contents of all battery-backed RAMs.

4.4.1 Programming the EPROMs for the large memory configuration (120 kbytes)

4.4.1.1 Programming the two program EPROMs using the 07 PP 31 programmer (SE 4943) or the 07 PP 32 programmer (SE 4944)

EPROM type: 2 x 27C512 (64 k x 8 bits)

1. Connect the programmer to interface 1 (bottom interface) of the PLC using the 07 SK 23 R1 system cable.

2. Set the mode of "Automatic identification of the inserted EPROMs".

   Key sequence: ROMTYPE 0 SET
   Display: ID AUTO

3. For data transfer from the PLC to the programmer, the following characteristic values must be set on the programmer:

Baud rate: 9600
Parity: NONE
Synchronization: XON

Key sequence: SELECT A

- Set the baud rate with the keys: 0 or 1
- Select parity setting with the keys: ↑ or ↓
- Set the parity with the keys: 0 or 1
- Select synchronization setting with the keys: ↑ or ↓
- Set synchronization setting with the keys: 0 or 1
  Note: the brackets must contain the characters XON.
- Terminate the command by pressing the key: SET

4. Set the programmer to reception

   Key sequence: SELECT 6 0 SET

5. By means of the command PU <CR>, transfer the program part for the EPROM LOW BYTE from the PLC to the programmer. The complete contents of the LOW byte of the program EPROM (64 kbytes) can be stored in the programmer's internal RAM. The transfer can therefore take place in one go.

   Therefore in the dialog with the PLC, answer

   "NO" in response to the prompt
   "TRANSFER OF 2*32 K ?"

6. Insert the EPROM in the programmer and program it. On the 07 PP 32, the middle EPROM socket must be used.

   Key sequence: SET DEVICE B.P.R. SET DEVICE SET

7. Program the EPROM HIGH BYTE analogously to steps 4., 5. and 6.

After programming, plug both EPROMs into sockets

XA 12 (EPROM LOW BYTE) and
XA 13 (EPROM HIGH BYTE).

4.4.1.2 Programming the two program EPROMs with a programmer that has an internal RAM of 32 kbytes

The contents of each of the two program EPROM's (64 kbytes) are transferred from the PLC to the programmer's internal RAM in two segments of 32 kbytes each. This is why, in the dialog between the PLC and the user, the answer must be

"YES" in response to the prompt
"TRANSFER OF 2*32 K ?"

- The PU command now transfers, from the PLC to the PROM programmer's RAM, the first half of the program intended for the EPROM on socket XA12 (EPROM LOW BYTE).
- The user now loads the contents of the PROM programmer's RAM into the program EPROM LOW BYTE from EPROM address 0 up to EPROM address $7FFF_H$.
- The PU command now transfers the next half of the program intended for the EPROM on socket XA12 (EPROM LOW BYTE).
- The user now loads the contents of the PROM programmer's RAM into the program EPROM LOW BYTE from EPROM address 8000_H up to EPROM address $FFFF_H$.
- The programming of the program EPROM (LOW BYTE) is now finished. Next the program EPROM (HIGH BYTE) will be programmed.
- The PU command now transfers, from the PLC to the PROM programmer's RAM, the first half of the program intended for the EPROM on socket XA13 (EPROM HIGH BYTE).
- The user now loads the contents of the PROM programmer's RAM into the program EPROM HIGH BYTE from EPROM address 0 up to EPROM address $7FFF_H$.
- The PU command now transfers the next half of the program intended for the EPROM on socket XA13 (EPROM HIGH BYTE).
- The user now loads the contents of the PROM programmer's RAM into the program EPROM (HIGH BYTE) from EPROM address 8000_H up to EPROM address $FFFF_H$.
- The programming of the program EPROM (HIGH BYTE) is now finished.

After programming, the two EPROMs are plugged into the sockets

XA 12 (EPROM LOW BYTE) and
XA 13 (EPROM HIGH BYTE).

4.4.2 Programming the EPROMs for the small memory configuration (56 kbytes)

4.4.2.1 Programming the two program EPROMs using the 07 PP 31 programmer (SE 4943) or the 07 PP 32 programmer (SE 4944)

EPROM type: 2 x 27C256 (32 k x 8 bits)

1. Connect the programmer to interface 1 (bottom interface) of the PLC using the 07 SK 23 R1 system cable.

2. Set the mode of "Automatic identification of the inserted EPROMs".

   Key sequence: ROMTYPE 0 SET
   Display: ID AUTO

3. For data transfer from the PLC to the programmer, the following characteristic values must be set on the programmer:

   Baud rate: 9600
   Parity: NONE
   Synchronization: XON

   Key sequence: SELECT A

   - Set the baud rate with the keys: 0 or 1
   - Select parity setting with the keys: ↑ or ↓
   - Set the parity with the keys: 0 or 1
   - Select synchronization setting with the keys ↑ or ↓
   - Set synchronization setting with the keys: 0 or 1
     Note: the brackets must contain the characters XON.
   - Terminate the command by pressing the key: SET

4. Set the programmer to reception

   Key sequence: SELECT 6 0 SET

5. By means of the command PU <CR>, transfer the program part for the EPROM LOW BYTE from the PLC to the programmer.

6. Insert the EPROM in the programmer and program it. On the 07 PP 32, the middle EPROM socket must be used.

   Key sequence: SET DEVICE B.P.R. SET DEVICE SET

7. Program the EPROM HIGH BYTE analogously to steps 4., 5. and 6.

After programming, plug both EPROMs into sockets

XA 10 (EPROM LOW BYTE) and
XA 11 (EPROM HIGH BYTE).

4.4.2.2 Programming the two comment EPROMs using the 07 PP 31 programmer (SE 4943) or the 07 PP 32 programmer (SE 4944)

EPROM type: 2 x 27C512 (64 k x 8 bits)

1. Connect the programmer to interface 1 (bottom interface) of the PLC using the 07 SK 23 R1 system cable.

2. Set the mode of "Automatic identification of the inserted EPROMs".

   Key sequence: ROMTYPE 0 SET
   Display: ID AUTO

3. For data transfer from the PLC to the programmer, the following characteristic values must be set on the programmer:

   Baud rate: 9600
   Parity: NONE
   Synchronization: XON

   Key sequence: SELECT A

   - Set the baud rate with the keys: 0 or 1
   - Select parity setting with the keys: ↑ or ↓
   - Set the parity with the keys: 0 or 1
   - Select synchronization setting with the keys ↑ or ↓
   - Set synchronization setting with the keys: 0 or 1
     Note: the brackets must contain the characters XON.
   - Terminate the command by pressing the key: SET

4. Set the programmer to reception

   Key sequence: SELECT 6 0 SET

5. By means of the command PU <CR>, transfer the comment part for the EPROM LOW BYTE from the PLC to the programmer. The complete contents of the LOW byte of the comment EPROM (64 kbytes) can be stored in the programmer's internal RAM. The transfer can therefore take place in one go.

   Therefore in the dialog with the PLC, answer

   "NO" in response to the prompt
   "TRANSFER OF 2*32 K ?"

6. Insert the EPROM in the programmer and program it. On the 07 PP 32, the middle EPROM socket must be used.

   Key sequence: SET DEVICE B.P.R. SET DEVICE SET

7. Program the EPROM HIGH BYTE analogously to steps 4., 5. and 6.

After programming, plug both EPROMs into sockets

XA 12 (EPROM LOW BYTE) and
XA 13 (EPROM HIGH BYTE).

4.4.2.3 Programming the two comment EPROMs with a programmer that has an internal RAM of 32 kbytes

The contents of each of the two comment EPROM's (64 kbytes) are transferred from the PLC to the programmer's internal RAM in two segments of 32 kbytes each. This is why, in the dialog between the PLC and the user, the answer must be

"YES" in response to the prompt
"TRANSFER OF 2*32 K ?"

- The PU command now transfers, from the PLC to the PROM programmer's RAM, the first half of the comments intended for the EPROM on socket XA12 (EPROM LOW BYTE).
- The user now loads the contents of the PROM programmer's RAM into the comment EPROM LOW BYTE from EPROM address 0 up to EPROM address $7FFF_H$.
- The PU command now transfers the next half of the comments intended for the EPROM on socket XA12 (EPROM LOW BYTE).
- The user now loads the contents of the PROM programmer's RAM into the comment EPROM LOW BYTE from EPROM address 8000_H up to EPROM address $FFFF_H$.
- The programming of the comment EPROM (LOW BYTE) is now finished. Next the comment EPROM (HIGH BYTE) will be programmed.
- The PU command now transfers, from the PLC to the PROM programmer's RAM, the first half of the comments intended for the EPROM on socket XA13 (EPROM HIGH BYTE).
- The user now loads the contents of the PROM programmer's RAM into the comment EPROM HIGH BYTE from EPROM address 0 up to EPROM address $7FFF_H$.
- The PU command now transfers the next half of the comment intended for the EPROM on socket XA13 (EPROM HIGH BYTE).
- The user now loads the contents of the PROM programmer's RAM into the comment EPROM HIGH BYTE from EPROM address 8000_H up to EPROM address $FFFF_H$.
- The programming of the comment EPROM (HIGH BYTE) is now finished.

After programming, the two EPROMs are plugged into the sockets

XA 12 (EPROM LOW BYTE) and
XA 13 (EPROM HIGH BYTE).

5 PLC operating modes

The user may set various operating modes for the PLC. The operating modes are configured by means of the operator control command KONFS (set configuration). The PLC stores the configured operating modes in its user program memory. If an EPROM of the user program is also programmed, then the mode setting is also stored in it.
There are unit-specific and program-specific modes. The unit-specific modes apply to the complete PLC and the ones specific to the program only apply to one of the respective PLC programs.

Unit-specific modes

- GERMAN / ENGLISH
  The PLC is operated either in German *or* in English
- MASTER PLC / SLAVE PLC
  The PLC is a master *or* a slave PLC
- AUTO PLC / PLC WITH ZST
  The PLC is an autonomous PLC *or* one which is controlled by a central control unit (ZST)
- PTG OPERATION / PLC OPERATION
  The PLC is in programming and test mode *or* in PLC mode
- COPY UP / NOCOPY UP (user program)
  Copying of user programs from the EPROM to the RAM during a warm start *or no* copying of user programs
- 120 kB / 56 kB
  This sets the PLC program memory to a total of 120 kbytes *or* 56 kbytes
- CHANGE ABORT / CHANGE RUN
  Program changes are executed either when the program is aborted *or* when it is running
- NO SENSOR / SENSOR
  No operation with the VIDEO SENSOR *or* operation with the VIDEO SENSOR

Program-specific modes

- NORMAL / TURBO
  The program is operated in NORMAL *or* TURBO mode
- TIME-CYCL. / EVENT
  The program is operated in CYCLE TIME *or* EVENT mode
- INI OPS or NOINI OPS
  Each time the user program is restarted, the operand memory is initialized *or* is not initialized
- PRIO HIGH / PRIO LOW
  The program has higher *or* lower processing priority in relation to the other program

Mutually exclusive modes

The modes which are entered in *one* line are exclusive, i.e. only one mode can be configured. The user can ignore the principle of exclusiveness, since this is guaranteed automatically by the configuration command.

Coupled modes

Some PLC operating modes are coupled to each other, i.e. they *cannot* be set independently of each other. This manifests itself as follows:

- When a mode to which other modes are coupled is configured, the PLC sets them automatically to the correct parameters and these can also no longer be changed by the user.
  Example: if CHANGE RUN is configured, the PLC automatically sets the modes NO SENSOR and CYCLE TIME.

- To set specific modes it is necessary to previously set another one to a specific parameter.
  Example: if CHANGE RUN is to be configured, this is only possible if the large program memory (120 kbytes) has been set beforehand.

The *next modifiable* mode is always set by specifying a semicolon when using the KONFS command. At the same time, coupled modes that must not be modified are displayed, are skipped by the cursor and are thus not modifiable.

- The following modes are coupled to each other:

  - SENSOR
    ---> NORMAL for program 2 and
    ---> EVENT for program 2

  - 56 kbyte
    ---> CHANGE ABORT

  - CHANGE RUN
    ---> NO SENSOR and
    ---> ZEITCYCL and
    ---> 120 kbyte

5.1 Explanation of the individual operating modes

GERMAN / ENGLISH

GERMAN
Screen display in German

Note:

If the PLC is connected to the ABB programming system, it must always be set to GERMAN. This also applies to the English and French versions of the ABB programming system.

ENGLISH
Display in English.

MASTER PLC / SLAVE PLC

MASTER PLC

When operating several PLC processors in one subrack, these may communicate with one another by way of superglobal values. For this purpose, it is necessary for *one* of the PLC processors to be operated as the MASTER and all others to be operated as SLAVES. The physical location of superglobal value storage is in the MASTER PLC. This physical storage location is used both by the MASTER PLC and also by the SLAVE PLCs as the central superglobal value memory.

Important: The module address 1 must be set for the MASTER PLC.

SLAVE PLC

See also MASTER PLC

The slave PLC fetches/brings its superglobal values from/to the superglobal memory in the master PLC. Contrary to the MASTER PLC, no restrictions apply to setting the module address for the SLAVE PLC.

AUTO PLC / PLC WITH ZST

AUTO PLC

In the case of autonomous PLC, the PLC is initialized after power-on or after a RESET and is then ready. The two user programs are started automatically if the PLC mode is additionally set. If the PTG MODE (programming and test mode) is set instead of the PLC mode, the user programs are not started automatically.

PLC WITH ZST

The PLC is master-controlled by the Central Control Unit (ZST). The PLC is *not* initialized automatically after power-on or after a RESET. Instead, it waits until it has been guided through the initialization phase by the ZST in a handshake procedure (see also Chapter "PLC master-controlled by a central control unit").

Master control of the PLC by the ZST is based on the following reasons:

- Controlled start-up of all intelligent (active) units existing in the system (no undefined states in the system resulting from a "race" of the individual processors after the RESET).
- During initialization, the Central Control Unit can send data to the PLC and can thus influence the initialization procedure. In this way, the ZST can define the jumpering between the PLC and I/O modules, for instance.

If a ZST is in existence, it has the station address 0. This address is then no longer available for other active stations.

PTG OPERATION / PLC OPERATION

PTG OPERATION

Programming and test mode is the operating mode for entering and testing PLC programs. When this mode is set, the PLC is always initialized after a RESET (regardless of whether it is configured as an AUTONOMOUS PLC or as a PLC WITH ZST) and is then ready for input and testing of the program. The following applies:

- Any programs already existing in the EPROM are *not* copied into the user program RAM automatically. This prevents destruction of any more recent program version stored in the RAM.
- Programs existing in the user program RAM are not started automatically.

PLC OPERATION

The PLC mode is the operating mode in which the controller is operating in the plant once the PLC programs have been entered and tested. The following applies:

- After a RESET/WARM START, the user program EPROM's contents are copied into the user program RAM if the user has set this configuration (COPY AWP (user program) mode).
- The two PLC programs are started automatically if the PLC is an AUTONOMOUS PLC and PLC programs are loaded.

COPY UP (user program) / NOCOPY UP

COPY UP

The user program EPROM's contents are copied into the user program RAM on each warm start. This does not apply when the PTG mode is set. A warm start takes place after every RESET (see also Chapter "Starting behavior").

NOCOPY UP

The user program EPROM's contents are *not* copied into the user program RAM when a warm start takes place.

120 KB / 56 KB

120 KB

The program memory capacity is 120 kbytes.

- Storage of comments in the PLC is *not* possible.
- Changes can be made while the PLC is running.
- For details of the memory components, see the chapters entitled "Hardware and commissioning" and "Memory subdivisions".

56 KB

The program memory capacity is 56 kbytes.

- Storage of comments in the PLC is possible.
- Changes *cannot* be made while the PLC is running.
- For details of the memory components, see the chapters entitled "Hardware and commissioning" and "Memory subdivisions".

CHANGE ABORT / CHANGE RUN

CHANGE ABORT

Changes to the PLC program are only possible if the PLC program has been aborted.

CHANGE RUN

Changes to the PLC program are *also* possible if the PLC program is running. The following applies:

- Program changes are adopted *without* a program standstill time.
- The operating mode can only be terminated again by means of a cold start (command of KALT <CR>, see also the chapter entitled "Operating and test functions").
- No longer *two*, but now only *one* PLC program is available to the user.
- The program memory must be set to 120 kbytes.
- The PLC program must be running in the CYCLE TIME mode.
- The NO SENSOR mode must be set.

Note:

When processing programs in accordance with DIN 66264 in the CHANGE RUN mode, please pay attention to the chapter entitled "PLC controlled by a central control unit (ZST)".

NO SENSOR / SENSOR

NO SENSOR

The SENSOR mode is deactivated.

SENSOR

The PLC represents a video sensor system together with the iconic image processor 35 IV 90. In the user program 2 the data of the sensor system are evaluated.

The SENSOR mode and the TURBO mode (user program 2) are mutually exclusive because the TURBO memory of program 2 is used to store the sensor constants (KW 1000,00...KW 2041,15).

When configurating the SENSOR mode, the EVENT and NORMAL modes are set automatically for user program 2. For this program, the EVENT and NORMAL modes cannot be altered as long as the SENSOR mode is configured.

NORMAL / TURBO

NORMAL

The PLC program is executed by interpretation of the intermediate code.

TURBO

On starting of the program in the PLC, the PLC program is translated to 8086 machine code and is stored in the TURBO RAMs. Before translation, a RAM test is carried out for the TURBO RAMs and a syntax check is carried out for the PLC program. The machine code of the PLC program is then executed directly from the TURBO RAM. The speed advantage in comparison with NORMAL MODE amounts to a factor of 6...7.

CYCLE TIME/EVENT

CYCLE TIME

The corresponding user program is executed cyclically in respect of time, i.e., regularly at equal time intervals. The cycle time is predetermined by the double word constant of KD 0,0.

EVENT

The affiliated user program is executed by event control, i.e., the user program is run through *once* with program start and with every event (SRQ interrupt).

INI OPS / NOINI OPS

INI OPS

When starting the PLC program, the operands (flags, step chains, I/O image) and historical values belonging to it are initialized to "zero". Sometimes it is necessary to "save" some of the values of flags beyond initialization. This is why from PLC version V8 onwards, a limited number of flags is available which is not affected by initialization in INI OPS mode. These are:

- BINARY flags:	M 256,00... M 257,13
- WORD flags:	MW 256,00... MW 258,15
- DOUBLE WORD flags:	MD 32,00... MD 33,15

The following applies to older PLC versions:
Flags to be saved before initialization can be written into the areas of operand memory marked as "free" and can be read back from these again after starting the program. These free areas are not initialized when starting the program. Writing and reading back can be done with the following function blocks, for instance: WOS, WOL, DWOS, DWOL or COPY.

The operand memory contains a total of more than 500 bytes of free areas (refer also to the respective PLC description, Fig. "Detailed overview of the operand memory").

Important:

The operand memory must be initialized after every program change affecting the sequence of modules containing *historical values*. As a result of the program change, invalid historical values are assigned to the modules, which can lead to incorrect behaviour of the module.

NOINI OPS

When the PLC program is started, the operands and historical values belonging to it are *not* initialized, i.e., the flags, step chains, the image of the inputs/outputs and the historical values are retained.

The binary flag of M 255,15, however, is always initialized to 0 each time the user program is started. When this flag is interrogated in the user program and then set to 1, it is possible to determine whether or not the user program has been restarted.

As of version V8.0, also in NOINI OPS the timers (ESV, ASV, VVZ, MOA, MOK) are initialized after the *first* start of the PLC program *after a warm start* (power ON, RESET or WARM command). Therefore, after a warm start timers behave identically in the INI OPS and NOINI OPS modes. Running timers are not influenced by ABORTING and subsequent starting of a PLC program.

Specific initialization:

In NOINI OPS mode also, by planning the user can *specifically* initialize individual areas in the operand memory with the value of 0. This may be necessary if the PLC status stored *in its entirety* would have a disturbing influence on reactivation. The function block of "Initialization of a memory area" (INITS) (see also module catalogue) is available for specific initialization.

The area to be initialized must be specified by:

- the name of the *first* operand to be initialized (E, EW, A, AW, M, MW, MD, S)
- and the number of words to be initialized.

In this way, the "historical value memory" (VWS) can also be initialized with the value of 0. The VWS are to be initialized is specified as follows:

- Starting operand: step flag of S 127,14
- Number of words: C00 hex or 3072 decimal

The step flags of S 127,14 and S 127,15 occupy the last memory word *before* the historical value memory. S127,14 is located at an even address and, for reasons of computing time, is more suitable than the step flag of S 127,15 for marking the start of the area. Attention must be paid to the fact that, as a result of this, the two step flags are also initialized to the value of 0.

Important:

As of version 935 PC 83 V 6.0 (GJP5 1229 00 R301), the configured mode of INI OPS or NOINI OPS can be rendered inactive by means of an entry in the control block of the transfer memory of the corresponding user program. This entry has a higher priority than the configured mode and, via the MPST bus, can be entered in the required control block of the PLC. The control blocks are defined in chapter "PLC controlled by a Central Control Unit (ZST)". Control block 1 is crucial for user program 1. Control block 2 is crucial for user program 2. The entry in the control block is defined as follows:

Init flag 1 Byte 0FH, Bit 7	**Init flag 2** Byte 0EH, Bit 7	**Meaning**
0	X	the configured mode applies
1	0	*no* initialization of the corresponding program's operands takes place
1	1	the corresponding program's operands *are* initialized.

PRIO HIGH / PRIO LOW

The two programs have different priorities as regards execution by the processor. The program with the high priority is capable of interrupting the program with the low priority. This is why the program with the shorter cycle time must be assigned the high priority so that this shorter cycle time can also be obeyed exactly.

PRIO HIGH

The selected program is assigned the high priority. At the same time, the low priority is assigned to the other program.

PRIO LOW

The selected program is assigned the low priority. At the same time, the high priority is assigned to the other program.

5.2 Defaults

The following mode defaults are set after a cold start: (cold start: COLD command or after power-on for the PLC)

- GERMAN
- MASTER PLC
- AUTO PLC
- PTG OPERATION
- COPY AWP (user program)
- 120 KB
- CHANGE ABORT
- NOSENSOR
- NORMAL
- CYCLE TIME
- INI-OPS
- PRIO HIGH for program 1
- PRIO LOW for program 2

5.3 Changes to the operating modes by the user

Prerequisites

- The two PLC programs must be in "ABORTED" state.

The following mode changes take effect immediately:

- GERMAN <---> ENGLISH
- 120 KB <---> 56 KB
- CHANGE ABORT ---> CHANGE RUN

The following modes take effect when PLC programs are started:

- NO SENSOR <---> SENSOR
- NORMAL <---> TURBO
- CYCLE TIME <---> EVENT
- INI OPS <---> NOINI OPS
- PRIO HIGH <---> PRIO LOW

The following mode changes do not take effect until after a warm start:

(Warm start: WARM command or RESET or power OFF/ON)

- AUTO PLC <---> PLC WITH ZST

  When PLC WITH ZST and PLC MODE are set, no warm start can be realized by means of the warm start command because, in this case, the PLC is started up by the ZST. If the warm start command is issued inadvertently, the PLC will wait for initialization by the ZST. This waiting state can be aborted by entering the command <Ctrl>C.

- PTG OPERATION <---> PLC OPERATION
- COPY AWP <---> NOCOPY AWP

The following mode changes do not take effect until after the DBV RES (reset data block directory) command followed by a warm start/reset:

- MASTER PLC <---> SLAVE PLC

The following mode change can only be achieved by a cold start:

- CHANGE RUN ---> CHANGE ABORT

6 Language repertoire

Two user programs can be planned in the PLC. A user program consists of

- sentences (equation notation) and/or
- block calls.

The sentences consist of individual instructions conforming to DIN 19239. The sentences and block calls may be mixed as required. The program end is indicated by !PE.

Possibilities of program input:

- With the ABB Procontic programming system, the PLC program can be created, transferred to the PLC and tested using a very user-friendly programming interface.
- The language objects described in this chapter and the blocks described in the block library can, however, also be entered directly in the PLC and tested as an instruction list on a terminal.

6.1 Operators for planning sentences (DIN 19239)

Sentence start operators for Boolean sentences

!	Sentence start
!N	Sentence start with negation

Boolean operators

&	AND
&N	AND with negation
/	OR
/N	OR with negation

Sentence start operators for arithmetic sentences

!	Sentence start
!-	Sentence start with negation

Arithmetic operators

+	PLUS
-	MINUS
*	MULTIPLIED by
*-	MULTIPLIED by with negation
:	DIVIDED by
:-	DIVIDED by with negation

Brackets

(	OPEN BRACKET
(N	OPEN BRACKET with negation (binary)
(-	OPEN BRACKET with negation (arithmetic)
)	CLOSE BRACKET

The maximum bracket depth is 15.

Comparison operators

Note:

If a comparison operator is not followed by a single variable on the left and/or right, but by a complete expression, each expression must be placed in brackets.

Example:

!(EW0,0+MW0,0*KW0,5) > (MW5,8+MW3,4) = A0,0

Comparison expressions must *not* be combined further with Boolean expressions in one sentence. Intermediate flags must be planned for this purpose.

Example:

- correct: !MW 0,0 > MW 0,1 = M 5,5
  !M 5,5 & M 0,1 = A 0,0
- incorrect: !MW 0,0 > MW 0,1 & M 0,1 = A 0,0 or
  !(MW 0,0 > MW 0,1) & M 0,0 = A 0,0

>	Greater than
>-	Greater than with negation
<	Less than
<-	Less than with negation
>=	Greater than or equal to
>=-	Greater than or equal to with negation
=?	Equal to
=?-	Equal to with negation
<>	Unequal to
<>-	Unequal to with negation
<=	Less than or equal to
<=-	Less than or equal to with negation

Allocations

=	Equal to
=N	Equal to with negation (binary)
=-	Equal to with negation (arithmetic)
=SM	Set flag
=RM	Reset flag
=S	Set step
=PE	Conditional program end

6.2 Block calls

The available blocks and their calls are described in the block catalog.
Calls in the instruction list have the following format:
!BA number (this parameter is optional)
Name

where:

!BA	::= key word in accordance with DIN 19239
Number	::= 0...999, No. of the call
Name	::= name of the calling block

6.3 NOP

The user program may contain NOPs.

NOP (unwritten memory cell)

Restrictions:

- Block calls: block calls must *not* contain NOPs.
- Sentences: NOPs may be placed within sentences, but only additionally directly before an operator. They must *not* be used instead of an operand (e.g. flag).

NOPs may be placed without restriction between sentences/blocks.

Example:

- correct:

```
NOP NOP ! M 0,0 NOP & M 0,1 NOP = A 0,2 NOP
NOP
NOP
!BA 123
  ESV
  E 0,0
  KD 0,5
  A 12,13
NOP
```

- incorrect:

```
! NOP & NOP M 0,1 = NOP A 0,2
!BA 123
  ESV
  NOP
  E 0,0
  KD 0,5
  A 12,13
```

6.4 Lables

Lables are used as branch destinations for forward branches in the blocks SPRUNG and LZB.

They are identified by: MA Number, with a number of 0...999.

6.5 Program end

Absolute

The absolute program end is always identified by: !PE

Conditional

The conditional program end is identified by: =PE

Example: !M 00,00 & M 00,01 = PE

6.6 Data types/formats

The PLC processes the following data types and formats:

- Binary: Boolean algebra
- INTEGER WORD (16 bits): Integer-arithmetic
  Number range: -32767 (8001_H) ... +32767 ($7FFF_H$)
- INTEGER DOUBLE WORD (32 bits):
  for presetting time values for
  - timers and
  - cycle times of PLC programs

  and for double word blocks.
  Number range: -2 147 483 647...+2 147 483 647
  ($8000\ 0001_H$) ... ($7FFF\ FFFF_H$).
  Only positive numbers may be specified for time values.

Number range

Word range:

- Lower limit: 8001_H -32767
- Upper limit: $7FFF_H$ +32767
- Invalid value: 8000_H /////

In two's complement arithmetics, the values 8000_H (-32768) is *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value should enter the PLC

- by bit manipulations by the user or
- by reading in from outside the PLC or
- by an indirect word constant

under *no* circumstances may negation or subtraction be applied to this value.

A permissible value is generated again by means of an allocation (=), an addition (+), a multiplication(*) or by division (:).

When the allocation (=) is used, the *forbidden* value 8000_H (-32768) is corrected to the allowed value 8001_H (-32767).

Double word range

- Lower limit: $8000\ 0001_H$ -2 147 483 647
- Upper limit: $7FFF\ FFFF_H$ +2 147 483 647
- Invalid value: $8000\ 0000_H$ ////////////

All blocks for double word arithmetics check the values to be processed for admissibility. If an admissible value occurs, it is corrected to the permissible value 8000 0001 $_H$ (−2 147 483 647).

6.7 Operand identifiers

The PLC processes the following operand identifiers (DIN 19239):

M	BINARY flag
S	BINARY step
MW	WORD flag
MD	DOUBLE WORD flag
K	Constant, binary (indirectly)
KW	Constant, word (indirectly)
KD	Constant, DOUBLE WORD (indirectly)
E	BINARY input binary
EW	WORD input
A	BINARY output
AW	WORD output
#	Direct constant, decimal (-32768...32767) *)
#H	Direct constant, hexadecimal (0000_H...$FFFF_H$) *)

*) Direct constants are permissible with specific blocks only (see also block catalogue). They are *not* permissible in instructions for bit and word processing.

6.8 Numbering of operands (variables)

The operands' syntax is orientated to DIN 19239 and has the following structure:

operand :: OPERAND IDENTIFIER FORMAT IDENTIFIER NUMBER

Example:

MW	123,15
A	11,11
E	04,08
EW	10,01
KD	00,00
S	127,15

The following are available:

- OPERAND IDENTIFIERS
  - E Input
  - A Output
  - M Flag
  - S Step
  - K Indirect Constant,
- FORMAT IDENTIFIERS
  - Binary
  - W Word
  - D Double word (for indirect constants and flags only, i.e. *not* for inputs and outputs)
- NUMBER
  - GROUP NUMBER, CHANNEL NUMBER

The number of an operand consists of a group number and a channel number. The group numbers and the channel numbers are separated by a comma. The GROUP NUMBER has a range of 0... max. The maximum group number depends on the OPERAND IDENTIFIER and on the FORMAT IDENTIFIER (see below). The CHANNEL NUMBER has a range of 0...15. The CHANNEL NUMBER 15 is again followed by the CHANNEL NUMBER 0, whereby the GROUP NUMBER is incremented by 1.

Binary operands

• Inputs:	E 00,00 ... E 63,15
• Outputs:	A 00,00 ... A 63,15
• Write global values:	A 64,00 ... A 71,15
• Read global values:	E 64,00 ... E 71,15
• Write superglobal values:	A 72,00 ... A 96,15
• Read superglobal values:	E 72,00 ... E 96,15
• Flags:	M 00,00 ... M 255,15 *)
• Remanent flags:	M256,00... M 257,13
• Steps:	S 00,00 ... S 127,15
• Indirect constants:	K 00,00 ... K 00,01

*) The binary flag M 255,15 is always initialized to 0 each time the user program is started. When this flag is interrogated in the user program and then set to 1, it is possible to determine whether or not the user program has been restarted.

Word operands

• Inputs:	EW 00,00...EW 13,15
• Read axis boards:	EW 14,00...EW 15,15
• Outputs:	AW 00,00...AW 13,15
• Write axis boards:	AW 14,00...AW 15,15
• Write global values:	AW 16,00...AW 17,15
• Read global values:	EW 16,00...EW 17,15
• Write superglobal values:	AW 18,00...AW 22,15
• Read superglobal values:	EW 18,00...EW 22,15
• Flags:	MW 00,00...MW 255,15
• Remanent flags:	MW256,00...MW 258,15
• Indirect constants:	KW 00,00...KW 39,15

In sensor mode, sensor constants can additionally be stored in the TURBO memory for user program 2. Input and/or display takes place in the same way as for the normal indirect constants. The sensor constants have the following logical names:

KW 1000,00 ... KW 2041,15

Double word operands

• Flags:	MD 00,00 ... MD 31,15
• Remanent flags:	MD 32,00 ... MD 33,15
• Indirect constants:	KD 00,00 ... KD 07,15

6.9 Time values

Real-time parameters are always planned as indirect double word constants or as double word flags. Real-time parameters are the time values which are specified direct in milliseconds, e.g., cycle times or the delays of timers. In the case of the time values specified in milliseconds, only integral multiples of the basic time amounting to 5 ms are allowed (the PLC rounds off other values to integral multiples of 5 ms).

Cycle time

The first double word constant in each program (KD00,00) is the cycle time which can be planned for this program.

Example: KD 00,00 = 50
This corresponds to a cycle time of 50 ms.

Time values scaled to the cycle time as occurring in controller blocks, for instance, are always word flags or indirect word constants. This is described exactly for each block concerned.

6.10 Text processing

Some PLC blocks (DRUCK, EMAS) process texts which are stored in the user program.

Entering texts into the user program

When entered, a text is embedded in the key symbols of #" and "#. The key symbol of #" identifies the start of a text string and the key symbol of "# identifies its end. The key symbols are *not* stored in the user program.

All ASCII characters between 1_H and FF_H may be entered.

Storage of texts in the user program

Each entered text character occupies *one* word in the user program. The ASCII identifier of the text character is stored in the low byte and the prefix of FA is stored in the high byte.

Example:

Text input and storage from address 100 in the PLC program:

Input: S 100 <CR>
00100 NOP #"ABB"#<CR>

Storage: 00100 FA41
00101 FA42
00102 FA42

Summary of the text characters, their input and **output**

ASCII character	Hex code	User input	Output
NUL	00	<CTRL><SP>	<NUL> *
SOH	01	<CTRL> A	<SOH>
STX	02	<CTRL> B	<STX>
ETX	03	<CTRL> C	<ETX>
EOT	04	<CTRL> D	<EOT>
ENQ	05	<CTRL> E	<ENQ>
ACK	06	<CTRL> F	<ACK>
BEL	07	<CTRL> G	<BEL>
BS	08	<CTRL> H	<BS>
HT	09	<CTRL> I	<HT>
LF	0A	<CTRL> J	<LF>
VT	0B	<CTRL> K	<VT>
FF	0C	<CTRL> L	<FF>
CR	0D	<CTRL> M	<CR>
SO	0E	<CTRL> N	<SO>
SI	0F	<CTRL> O	<SI>
DLE	10	<CTRL> P	<DLE>
DC1	11	<CTRL> Q	<DC1>
DC2	12	<CTRL> R	<DC2>
DC3	13	<CTRL> S	<DC3>
DC4	14	<CTRL> T	<DC4>
NAK	15	<CTRL> U	<NAK>
SYN	16	<CTRL> V	<SYN>
ETB	17	<CTRL> W	<ETB>
CAN	18	<CTRL> X	<CAN>
EM	19	<CTRL> Y	<EM>
SUB	1A	<CTRL> Z	<SUB>
ESC	1B	<CTRL> [	<ESC>
FS	1C	<CTRL> /	<FS>
GS	1D	<CTRL>]	<GS>
RS	1E	<CTRL> ~	<RS> *
US	1F	<CTRL> ?	<US> *
SP	20	<SP>	<SP> **
!	21	!	!
"	22	"	"
#	23	#	#
$	24	$	$
%	25	%	%
&	26	&	&
'	27	'	'
(	28	(	(
)	29	)	)
*	2A	*	*
+	2B	+	+
,	2C	,	,
-	2D	-	-
.	2E	.	.
/	2F	/	/
0	30	0	0
1	31	1	1
2	32	2	2
3	33	3	3
4	34	4	4
5	35	5	5

ASCII character	Hex code	User input	Output
6	36	6	6
7	37	7	7
8	38	8	8
9	39	9	9
:	3A	:	:
;	3B	;	;
<	3C	<	<
=	3D	=	=
>	3E	>	>
?	3F	?	?
@	40	@	@
A	41	A	A
B	42	B	B
C	43	C	C
D	44	D	D
E	45	E	E
F	46	F	F
G	47	G	G
H	48	H	H
I	49	I	I
J	4A	J	J
K	4B	K	K
L	4C	L	L
M	4D	M	M
N	4E	N	N
O	4F	O	O
P	50	P	P
Q	51	Q	Q
R	52	R	R
S	53	S	S
T	54	T	T
U	55	U	U
V	56	V	V
W	57	W	W
X	58	X	X
Y	59	Y	Y
Z	5A	Z	Z
[	5B	[	[
\	5C	\	\
]	5D	]	]
↑	5E	↑	↑
_	5F	_	_
`	60	`	`
a	61	a	a
b	62	b	b
c	63	c	c
d	64	d	d
e	65	e	e
f	66	f	f
g	67	g	g
h	68	h	h
i	69	i	i
j	6A	j	j
k	6B	k	k
l	6C	l	l
m	6D	m	m
n	6E	n	n
o	6F	o	o
p	70	p	p

ASCII character	Hex code	User input	Output
q	71	q	q
r	72	r	r
s	73	s	s
t	74	t	t
u	75	u	u
v	76	v	v
w	77	w	w
x	78	x	x
y	79	y	y
z	7A	z	z
{	7B	{	{
|	7C	|	|
}	7D	}	}
~	7E	~	~
DEL	7F	<DEL>	<DEL>

* On older terminals, the following applied:

ASCII character	Hex code	User input	Output
NUL	0	<CTRL> @	<NUL>
RS	1E	<CTRL> ↑	<RS>
US	1F	<CTRL> _	<US>

** On *text input*, a SPACE is indicated as a blank but, on outputs of the user program, it is indicated as <SP> for better recognition.

6.11 Step chains

6.11.1 Functionality of the step chain

Step chains serve to clearly control processes whose sequence is distinguished by sequential steps. The activation of a step is defined by the state of its previous step and by an additional enabling condition (transition).

Step chain length

A step chain consists of 16 steps (Sxxx,00...Sxxx,15).

Number of step chains

A total of 128 step chains is available per user program (S 000,yy...S 127,yy).

Number of steps

A total of 2048 steps is available per user program (S 000,00...S 127,15)

Idle state of a step chain

The idle state of a step chain is identified by the fact that the *first* step of the chain (Sxxx,00) *is set*.

The idle state of each chain is set:

- always when a cold start of the PLC is carried out
- but when starting the PLC program only when the "initialized operand memory" (INI OPS) mode is configured. If this mode is not configured, the step chains and all other operands retain their current state.

Setting a step

A step is set by means of an allocation.

It remains set until a *different* step is set in the *same* chain.

Example IM 1,5 & NE 3,3 = S11,11

Automatic reset of the previous step

One essential characteristic of the step chain is that only *one* step may be set in *one* step chain. That is to say, all other steps in the chain are reset automatically each time a step is set.

Step operand in sentences

In the *sentences* of the user program, the step is handled like a BIT operand. Therefore, the step may be placed both in the interrogation part as well as in the allocation part of the sentence.

Example ! (E 0,0 / M 7,13) & S 0,3 = M 1,1

Contrary to a BIT flag, however, the following instructions are inadmissible in the step and are rejected by the PLC when entered:

= NS xxx,yy
= SS xxx,yy
= RS xxx,yy

These allocations could lead to inadmissible states of step chains (no step is set or several steps are set).

Step operand in blocks: Important!!!

In *blocks*, the step may be present *only* at inputs but *not* at outputs as otherwise the predecessor step will not be reset automatically.

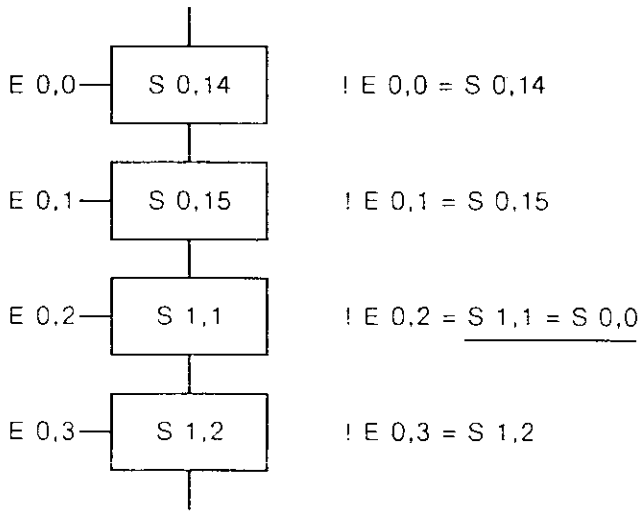
Example:	IBA 123	"block number"
	ESV	"ON delay type"
	S 1,1	"Input"
	KD 0,3	"Delay time"
	M 0,2	"Output"; *no* step operand may be present at the output

6.11.2 Joining step chains

The step chains are completely independent of each other.

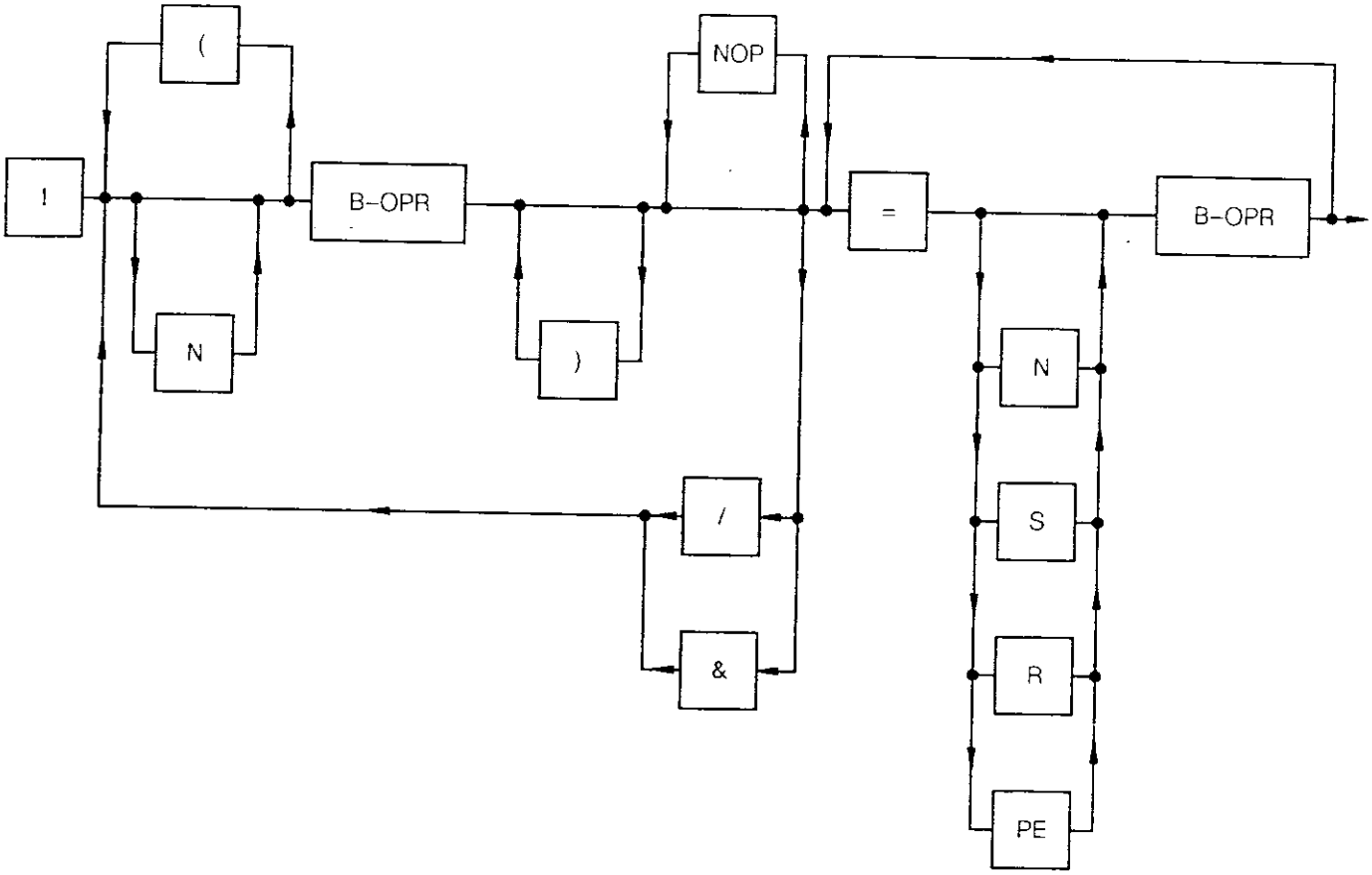
If more than 16 steps are needed within a PLC program, several step chains may be appended to each other. At the transition from one step chain to another, the step chain you are leaving must be set to idle state (0 step).

Example: Joining step chain 1 with step chain 2.



6.12 Syntax diagrams of the PLC language

6.12.1 Syntax diagram: BOOLEAN SENTENCE

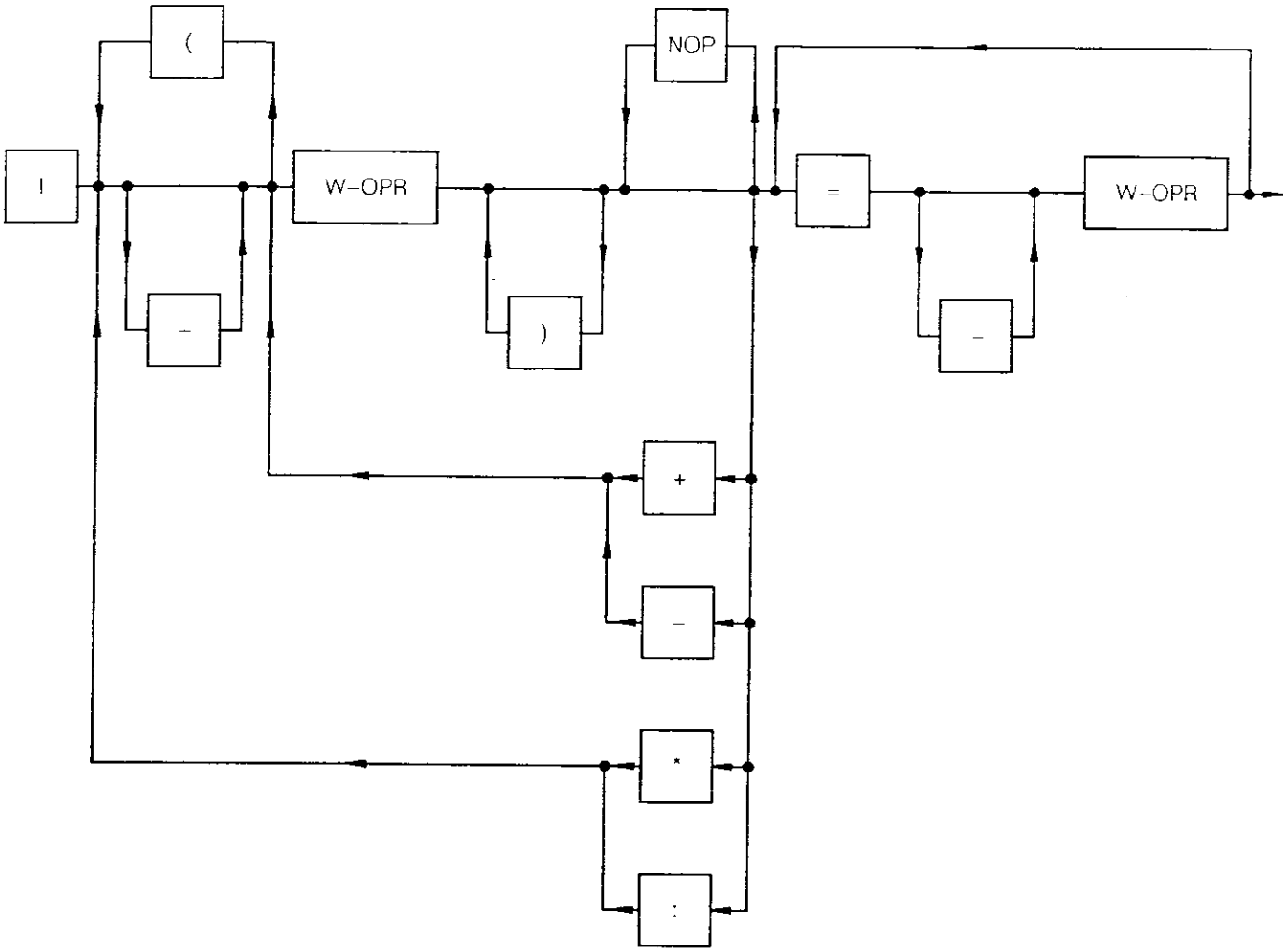


Signal flow: In the direction of the arrows; otherwise from left to right.

Brackets: Sum of "OPEN BRACKET" = Sum of "CLOSE BRACKET", bracket depth: 15

B-OPR: Bit operand (E, A, M, S, K)
Example: E 00,03 A 07,06 M 05,01
S 05,04 K 00,01

6.12.2 Syntax diagram: ARITHMETIC SENTENCE

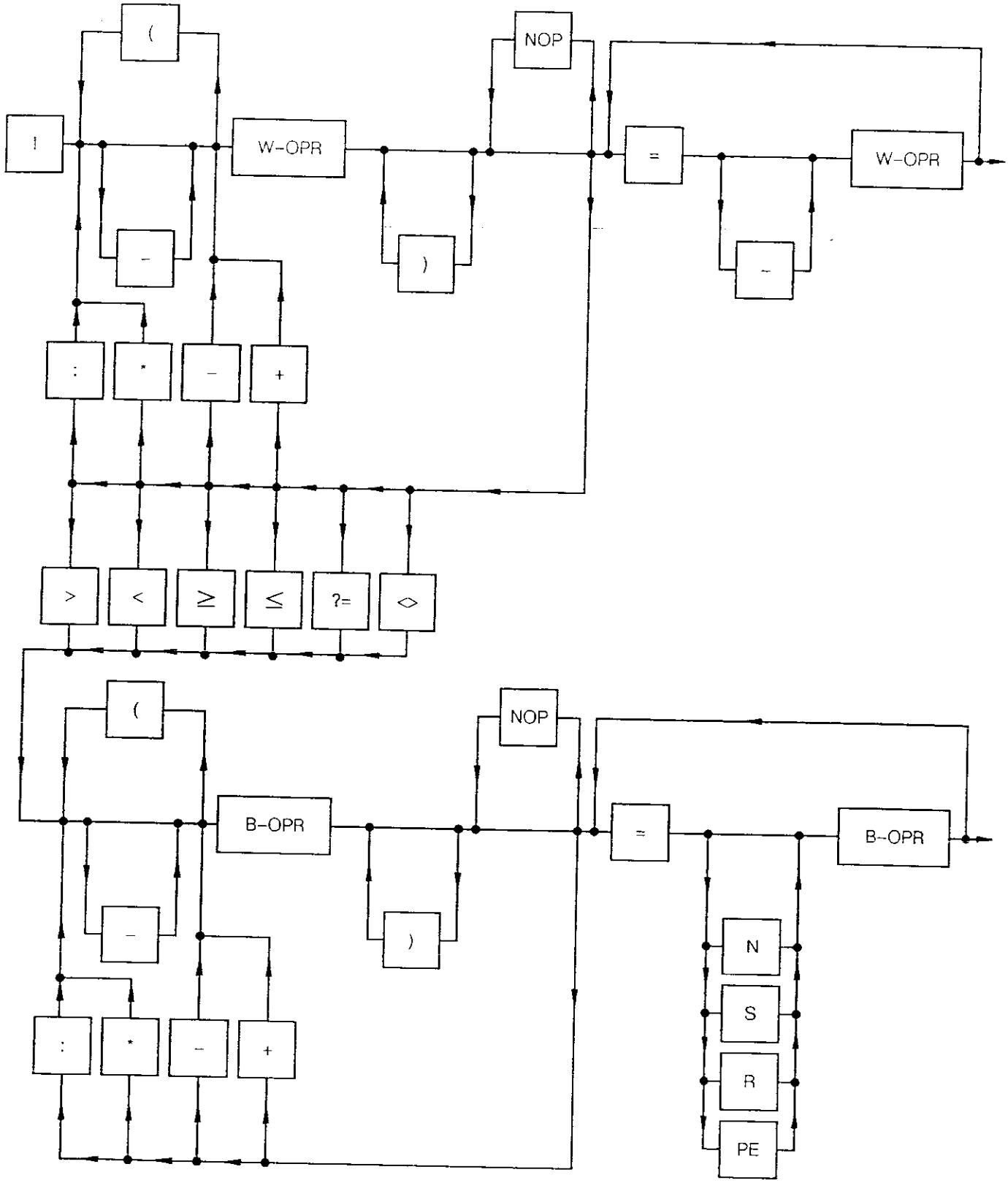


Signal flow: In the direction of the arrows; otherwise from left to right.

Brackets: Sum of "OPEN BRACKET" = Sum of "CLOSE BRACKET", bracket depth: 15

W-OPR: Word operand (EW, AW, MW, KW)
Example: EW 03,05 AW 11,12
MW 22,15 KW 09,06

6.12.3 Syntax diagram: HYBRID SENTENCE, see also section 6.1, Note



Signal flow: In the direction of the arrows; otherwise from left to right.

Brackets: Sum of "OPEN BRACKET" = Sum of "CLOSE BRACKET", bracket depth: 15

W-OPR: Word operand (EW, AW, MW, KW)
Example: EW 03,05 AW 11,12
MW 22,15 KW 09,06

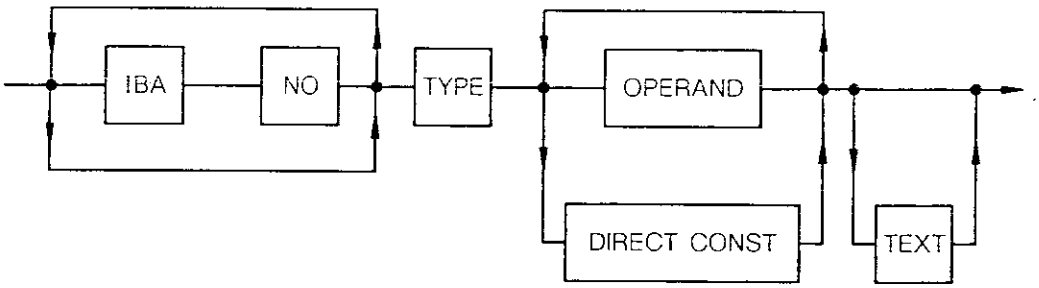
B-OPR: Bit operand (E, A, M, S, K)
Example: E 00,03 A 07,06 M 05,01
S 05,04 K 00,01

6.12.4 Global syntax diagram: blocks

Important:

Refer to the appendix for information on each block's detailed syntax.

In blocks, step operands may *only* be present at inputs and *not* at outputs:



Signal flow: In the direction of the arrows; otherwise from left to right.

No.: block number 0...999

TYPE: block type, see block catalogue

OPERAND: Bit operand (E, A, M, S, K)
Example: E 00,03 A 07,06 M 05,01
K 00,01 S 03,08

Word operand (EW, AW, MW, KW)
Example: EW 03,05 AW 11,1
MW 22,15 KW 09,06

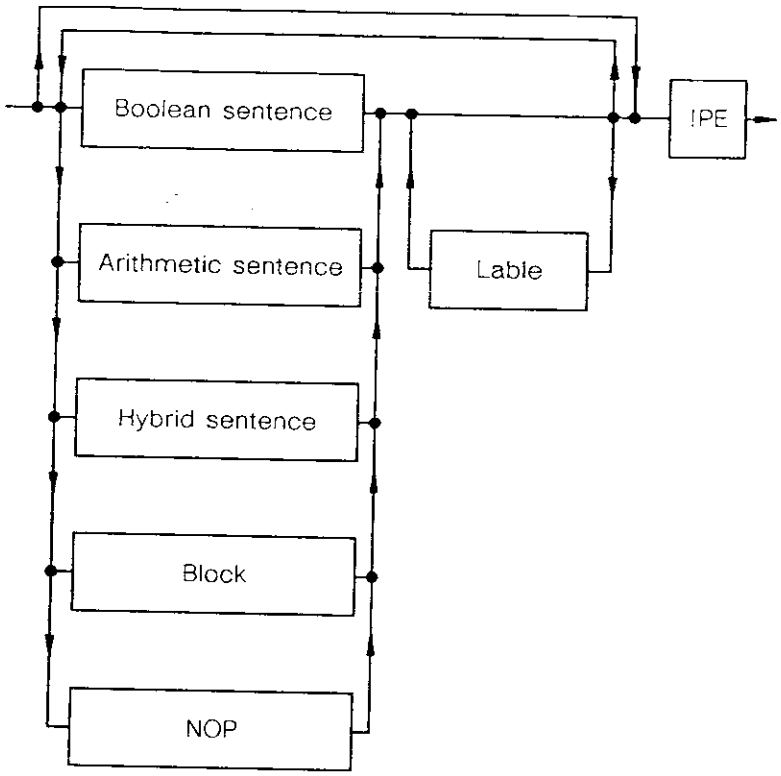
Double word operand (MD, KD)
Example: MD 02,15 KD 07,06

DIRECT CONST: -32767...32767 or 0_H...$FFFF_H$

TEXT: See blocks DRUCK and EMAS.

Note:
The step variable S may only be set at the input of a block, not at the output. The step variable for a block can only be called and not assigned. Otherwise incorrect functions of the step chains will be the result.

6.12.5 Syntax diagram: PLC program



Lable: MA 0 ... MA 999

7 Comment processing

Note:

Comments can be stored in the PLC:

- if entered **directly** in the PLC, i.e. this is not possible for comments which have been created using the ABB Procontic programming system. These comments remain in the programming system's mass storage.

- if the program memory has a capacity of 56 kbytes. (For reasons of space, when using a memory capacity of 120 kbytes, no comments can be stored in the PLC.)

The PLC offers the possibility of assigning comments to a user program entered directly in the PLC as instruction list and storing them in it. Comments are assigned to the instructions or module parameters when entering the program. They are stored in battery-backed RAMs fitted in sockets XA10...XA13. The comments can be transferred to two EPROMs (type 27C512) after terminating creation of the program. The EPROMs are then plugged onto the sockets instead of the RAMs.

7.1 Prerequisites

- On the 35 DS 91 unit, the four sockets

  - XA 10
  - XA 11
  - XA 12
  - XA 13

  must be equipped with RAMs (32k x 8) for creating and editing comments.

- If a comment has been transferred to EPROMs (27C512) and if these are plugged onto the sockets XA 12 and XA 13, the comment can now only be displayed.

- The following is needed to store comments in EPROM: A EPROM programmer with an internal RAM to which the comments are transferred as INTEL HEX file (for details see chapter entitled "Programming").

7.2 Comment inputs

With the exception of the NOP command, every instruction and every module parameter can be provided with a comment.

Input of a comment is commenced by quotation marks (") and is terminated by quotation marks.

After 255 characters have been entered, no further characters are accepted and the "conclude comment" message appears.

Comment length:

- 255 characters per instruction/block parameter
- (65535 - 4*n) characters per PLC program (n: number of comment blocks, i.e. number of commented instructions and module parameters.)

The following are permissible as comment characters:

- Letters, uppercase and lowercase
- Special characters
- <CR>
- Blanks

Control characters are not accepted because they are interpreted as control characters when displayed or printed out.

7.2.1 Input format

Comments are entered in formatted form. The comment line begins automatically in column 29 and ends in column 80. The transition to the next comment line is automatic. Even before you have reached column 80, you can move to a new comment line by entering <CR>.

7.2.2 Display format with command: D

The comment block belonging to an instruction/block parameter is displayed in the same way as it has been entered.

7.2.3 Display format with command: S

The comment block belonging to the selected instruction/block parameter is displayed in the same way as it has been entered. The instruction/block parameter is then repeated again *without* the comment. If necessary, the user may then recomment this instruction/block parameter.

7.2.4 Deleting a comment

You can work with the DELETE key while entering a comment block. A completed comment block is deleted by overwriting it with a blank comment block. The blank comment block consists of the comment start character followed directly by the comment end character.

7.2.5 Deleting all comments in a PLC program

All comments in a PLC program are deleted with the following command:

NOP KOM <CR>

7.2.6 Programming of comment EPROMs

The comments are transferred as an INTEL HEX file through the bottom serial interface of the PLC to an EPROM programmer.

Operator control command: PU <CR> (program transfer)

The further sequence takes place in a dialog between the PLC and the user. The procedure can be aborted at all times by pressing <CTRL> C. For details of the exact proceeding see chapter entitled "Programming".

8 Operating and test functions

The PLC operating and test functions can be called by means of a terminal or the ABB Procontic programming system.

When the required program number is entered, all subsequent inputs refer to this PLC program.
With the exception of the test function "Force", all previously entered test functions are deleted by input of a program number.

Note:

A user-friendly operator interface is available to users working with the ABB Procontic programming system. This user interface partly conceals or substitutes the syntax of the commands for the operator control and test functions listed in this chapter. In ONLINE mode, communication between the ABB Procontic programming system and the PLC takes place exclusively by means of these commands. For the user, however, this is not visible because he is working in the programming system's user interface. There is a separate instruction manual for the ABB Procontic programming system.

Operator control commands

The operator control commands can be subdivided into:

- commands for creating and editing user programs
- commands for testing user programs
- commands for configurating the PLC

Notes:

- No blanks are needed in user inputs and any blanks entered will be ignored.
- To improve clarity, in the description of the commands the user inputs for
  - key words are given in

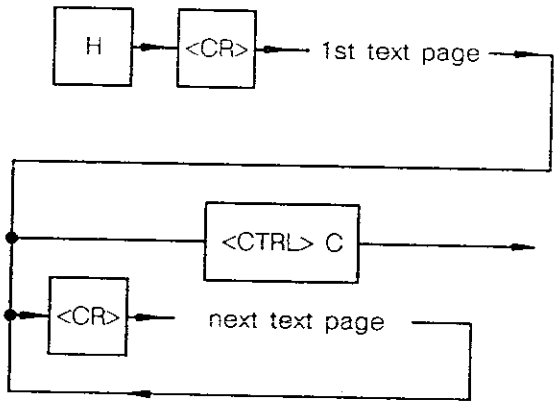
    UPPER CASE LETTERS
  - and, for other inputs (addresses etc.) in

    lower case letters.
- Displayed outputs generated by the PLC software are given in

  lower case letters in italics.

The HELP command can be used to display all available commands.

Help command



Function:
All available operating and test functions are displayed.
You can page through the HELP text by pressing <CR>.

Summary of commands for creating the user program

Command	Function	Page
AEND	Prepare a program change on a running program	8-3
ALT	Reject a program change on a running program	8-3
AL	Display PLC capacity utilization	8-3
CROSS	Display CROSS reference list	8-4
D	Display the user program	8-5
DRE	Activate printer	8-5
DRA	Deactivate printer	8-5
F	Search for string as from address	8-5
FREI	Enable a program change on a running program	8-6
IDA	Display program identification	8-6
IDR	Delete program identification	8-6
IDS	Enter program identification	8-6
K	Enter/edit constants	8-6
N	Preset/edit program number	8-7
NOP	Overwrite program (parts) with NOPs	8-7
NOP KOM	Delete comments of a program	8-8
O	Optimize program	8-8
P	Display free program memory area	8-8
PA	Program preparation	8-8
PU	Transfer program via serial interface as an INTEL hex file	8-9
S	Display/edit program memory contents	8-9
SO	Display/edit program memory contents without echo	8-9
V	Move user program	8-10

Summary of the commands for testing user programs

Command	Function	Page
A	Abort program	8-11
BA	Display breakpoints	8-11
BR	Reset breakpoints	8-11
BS	Set breakpoints	8-11
<CTRL>W	Change-over between operator control functions and monitor	8-12
EA	I/O test mode	8-12
EAA	Deactivate I/O mode	8-12
ES	Single step mode ON	8-12
ESA	Single step mode OFF	8-12
EZ	Single cycle mode ON	8-13
EZA	Single cycle mode OFF	8-13
FEHLER	Display contents of error register	8-13
FORC	Enter force value	8-14
FORC A	Display force value	8-14
FORC R	Delete force value	8-14
G	START program	8-14
KALT	Perform cold start for the PLC	8-15
L	Continue program	8-15
LED	LED on front panel ON/OFF	8-15
PS	Display program status	8-15
ST	Display PLC status	8-15
TRACE	Display TRACE memory	8-15
TRACE E	Activate TRACE mode	8-15
TRACE A	Deactivate TRACE mode	8-15
W	STOP program	8-16
WARM	Perform warm start for the PLC	8-16
Y	Set variable value	8-16
Z	Display variable status, 22 variables maximum	8-16
ZO	Display variable status, 120 variables maximum	8-17
ZZ	Display variable status without ESC sequences, 120 variables maximum	8-17
ZD	Display and continuously update variable status	8-17
INIT	Initialization of interfaces	8-17
$	Operation of several controls through *one* serial interface	8-18

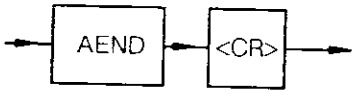
Summary of commands for configurating the PLC

Command	Function	Page
KONFA	Display PLC configuration (modes)	8-19
KONFS	Set PLC configuration (modes)	8-19
DBV	Display and edit data block directory	8-22
DBV RES	Reset data block directory to default configuration	8-23
UHR	Display the date and time	8-23
UHRS	Set the date and time	8-23

8.1 Commands for creating user programs

Prepare program change on a running PLC program

Command:



Function:

This command announces to the PLC that changes are to be made to the running PLC program. After input of this command the PLC is ready to accept program and constant changes.
The command is allowed only if CHANGE RUN mode is set because changes to a running PLC program can only be made in this mode.

By input of the AEND command, all momentarily active test functions become inactive, but force values of I/O signals remain active.

The following commands for program editing and operation of the PLC are permissible after input of the AEND command:
AL, CROSS, D, DRE, DRA, F, IDA, IDR, IDS, K, N, NOP, O, P, PA, PU, S, SO, V, CTRL-W, FEHLER, LED, INIT, $.

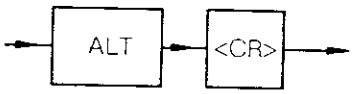
Rejecting a program change not yet enabled

Repeated input of the AEND command rejects all program changes previously made and the PLC is again ready to accept program changes.

The following commands become active when the program is running and additionally cancel the AEND command and thus all program changes made after input of the AEND command:
A, BA, BR, BS, EA, EAA, ES, ESA, EZ, EZA, FORC, FORC A, FORC R, G, L, PS, ST, TRACE, TRACE E, W, Y. The AEND command must be entered again if you wish to make program changes again.

Rejecting an enabled change to a running PLC program and reactivating the previous program state

Command:



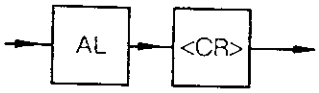
Function:

The changes made to a running PLC program and enabled are rejected again. Additionally, the PLC restores the previous program state. This is the state of the program that prevailed **before** the program change and so before input of the AEND command in the PLC.

After input of the ALT command the previous program state is reactivated within around 1 ms without any further actions on the part of the user.
The command can be used whenever the user recognizes that the program changes made are not having the desired effect.

Display capacity utilization

Command:



Function:

The current capacity utilization on the PLC is displayed as a percentage. The display states to what extent the PLC is loaded by execution of the user programs.

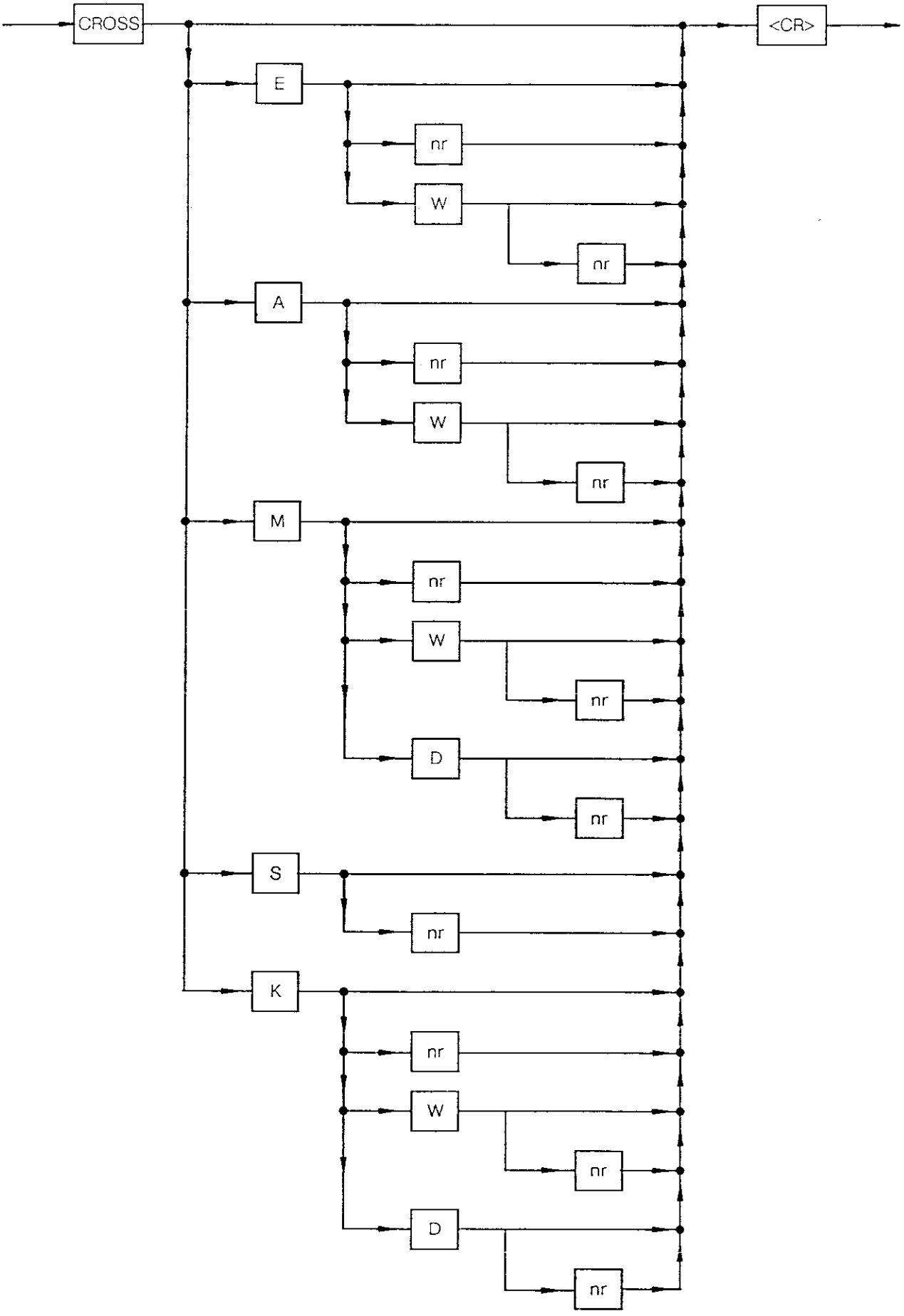
The processor capacity, which corresponds to the difference between 100 % and the capacity utilization display, is available for operation of the serial interfaces, i.e. for communication with the modules connected to the serial interfaces. To ensure that reasonable communication will still be possible through the serial interfaces, the capacity utilization with the longest program path should not be greater than 95 %. Please note that the PLC's load is also determined by the current program branches (conditional branches and run number blocks).

Note:

The display of capacity utilization only correctly shows the load caused by the user programs when *no* communication is taking place through the serial interfaces at the time of the display.

CROSS-reference list output

Command:



Where:

E: Abbreviation for input
A: Abbreviation for output
S: Abbreviation for step
M: Abbreviation for flag
K: Abbreviation for constant
W: Abbreviation for word size
D: Abbreviation for double word size
nr: Number of the operands

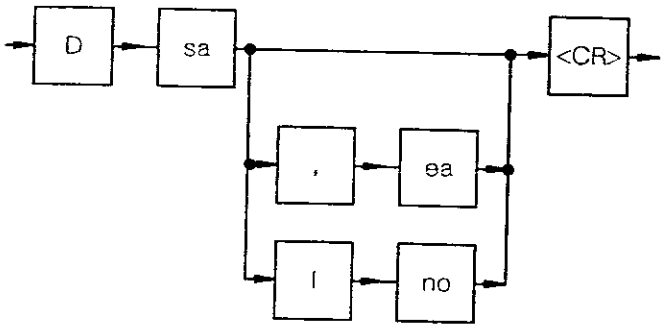
Function:

The cross-reference list shows the assignments of operands to the program memory addresses where they occur. The cross-reference list can be output for:

- all operands occurring in the program
  Enter: CROSS <CR>
- one specific operand type
  Enter e.g.: CROSS E <CR>
- one single operand
  Enter, e.g.: CROSS KD 00,12 <CR>

Display program

Command:



Where:

sa: Start address from which the program is to be displayed

ea: End address of the program to be displayed

l: Length (key word)

no: Number of program memory words to be output

Function:

The specific program part is displayed.

Examples:

- D 0,20 <CR>
  The user program is displayed from address 0 to address 20.
- D 10 L 20 <CR>
  20 program memory words are displayed from address 10 onwards.

Display format for sentences:

start address operator operand
:
:

Display format for block calls:

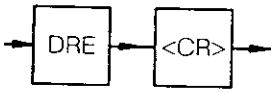
address n *!ba number*
address n+1 *type*
address n+2 *contents of adr n+2*

Example:

```
0000   !E 00,00
0002   &E 00,01
0004   =A 00,00
0006   !BA001
0007   AWT
0008   A 00,00
0009   KW 00,00
0010   KW 00,01
0011   AW 00,00
```

Printer ON/OFF

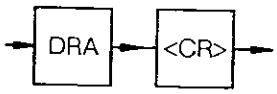
Command: printer ON



Function:

Switch on the printer.
A printer is connected to the bottom serial interface, interface 1 (8 data bits, no parity bit) of the PLC. The printer is operated in parallel with the monitor, i.e. all characters displayed on the monitor are also printed out.

Command: printer off

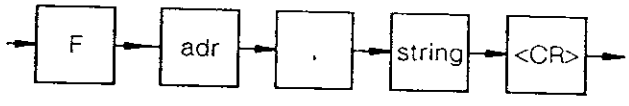


Function:

Switch off the printer.

Find string in the user program

Command:



adr: Start address from which the search for the string is to take place. The search begins from address 0 if no start address is entered.

string: Up to 8 commands, i.e. 16 words of the intermediate code.

Function:

The user program memory is searched through for the string entered by the user as from the entered start address up to the end of the user program memory. If the string is found, its address is displayed. If the string occurs several times in the program, the next program address agreeing with the string is displayed when you enter a semicolon (;).

Example:

F, E 0,0 & E 0,1 <CR>

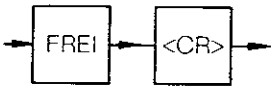
The entered string is sought as from start address 0.

F 100, !BA1 <CR>

The block call 1 is sought as from address 100.

Enabling a program change on a running PLC program

Command:



Function:

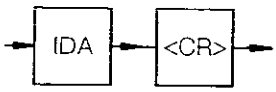
The changes made to a running PLC program after input of the AEND command are enabled for execution.

The program changes made are not processed by the PLC **until** input of the FREI command.

The changes made are processed by the PLC **after** input of the FREI command. The previous program state can be reactivated by entering the ALT command. The functional range of the PLC program can be modified further by making a new program change.

Display program identification

Command:

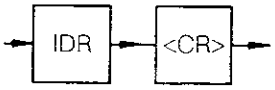


Function:

The identification entered by the user for the current user program is displayed. If no identification has been issued for the current program, nothing will be displayed (see also command: IDS).

Delete program identification

Command:

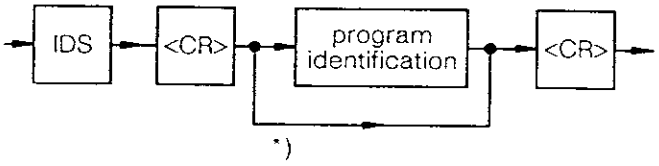


Function:

The identification entered by the user for the current user program is deleted.

Enter program identification

Command:



Program identification: These characters are assigned to the current user program as its identifier.

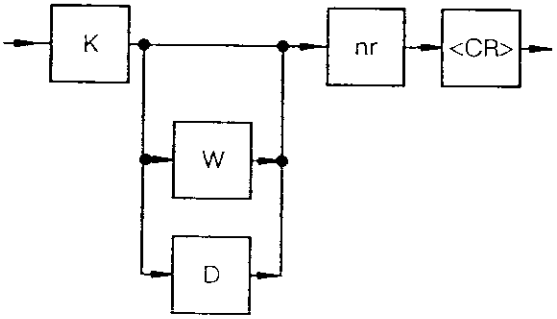
*): No program identification is entered for this path. Any existing program identification is deleted.

Function:

The identification entered by the user for the current user program is stored in the program memory. The identification may comprise up to 16 characters. For instance, it serves to store the project name and the date of program creation in the PLC.

Enter/edit values of indirect constants

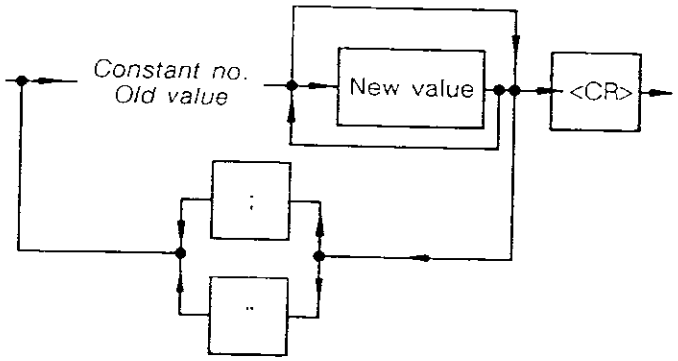
Command:



W: Abbreviation for word constants

D: Abbreviation for double word constants

nr: Entered number of the constant



Constant no. Old value: Displayed number and value of the constants.

New value: The user may overwrite the value of the displayed constant by a new value. Instead of a decimal value, a hex value may also be entered for the word and double word constants. For this purpose, the numerical value is preceded by an H.

Caution: The values of H8000 and H8000 0000 are forbidden in two's complement arithmetics (expedient only with masks, for instance).

;: Input of a semicolon results in the display of the number and value of the constant with the next highest number. If the semicolon is entered without a new value, the old value of the displayed constant is retained.

↑: Input of the "↑" character results in a display of the number and value of the constant with the next lowest number. If the "↑" character is entered without a new value, the old value of the displayed constant is retained. (Use the " ^ " symbol key for the PC.)

<CR>: The command is terminated by entering a <CR>.

Function:

The required numerical values are assigned to the indirect constants belonging to the user program.

From version 6.0 (935 PC 83 R0301), this value allocation can also take place while the user program is running. In this way, time values of timers, for instance, can be altered while the system is running.

Cycle time:

The cycle time is set with the double word constant of KD 00,00. The set cycle time must be an integral multiple of the basic time amounting to 5 ms, i.e., 5 ms, 10 ms, 15 ms, etc.

Example:

K 0,0 <CR>

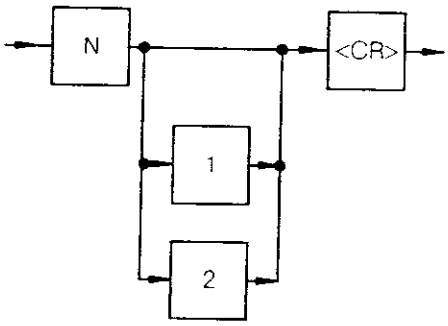
Output of the number and value of the binary constant of K 00,00. This value can be overwritten when required. The input of a semicolon results in an output of the number and value of the next binary constant (K 00,01).

KW 0,4 <CR>: Output of the number and value of the word constant of KW 00,04.

KD 0,0 <CR>: Output of the number and value of the double word constant of KD 00,00. The cycle time is specified with this constant.

Display/edit program number

Command:

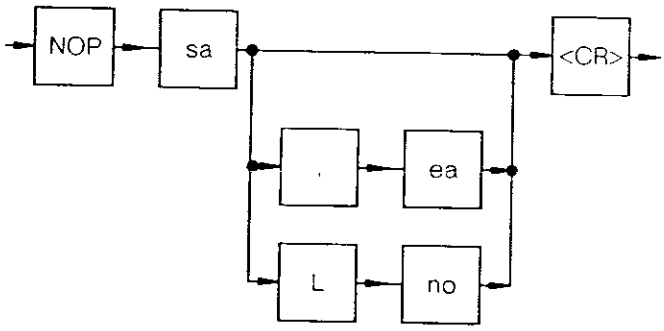


Function:

As the PLC is capable of processing two user programs, the user must specify the required program. With the exception of forcing, all previously entered test functions are rendered inactive. All subsequently entered commands refer to the user program with the entered program number. The currently set program number is displayed when N <CR> is entered.

Delete program parts, i.e., overwrite with NOPs

Command:



sa: Start address of the program part to be deleted

ea: End address of the program part to be deleted

l: Length (key word)

no: Number of program memory words to be deleted

Function:

The specified program part is deleted.

Examples:

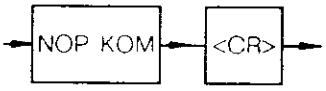
NOP 0,20 <CR>

The user program is deleted from address 0 to address 20.

NOP 10 L 20 <CR>
20 program memory words from address 10 onwards are deleted.

Deleting a program's comment

Command:

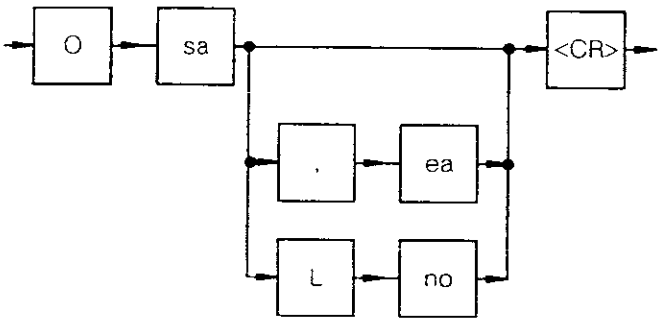


Function:

All comments in the set PLC program are deleted.

Optimize program

Command:



sa: Start address from which the program memory is to be optimized

ea: End address of the range

l: Length (key word)

no: Number of program memory words

Function:

All NOPs are removed from the specified program section and the program is condensed accordingly.

Examples:

O 0 <CR>

The complete program memory is optimized

O 0,10 <CR>

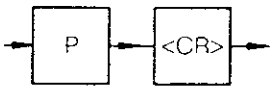
The program memory is optimized from address 0 to address 10.

O 10 L 10 < CR>

The NOPs within the next 10 program memory words from address 10 onwards are removed and the program is condensed accordingly.

Display of the free program memory

Command:

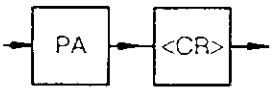


Function:

The program memory is checked for NOPs from its end. If a word is found in the intermediate code which does not correspond to an NOP, the number of NOPs found, i.e. the free program memory words, is displayed.

Configure user program

Command:

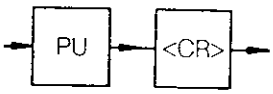


Function:

The I/O signals planned in the user program are enabled in the PLC's I/O configuration list. In addition, a syntax check is carried out for the user program.
In the case of blocks with comparison operators in which bracketed expressions are used, the CLOSE BRACKET operator placed before the binary allocations is stored by the translating program in the intermediate code as a binary CLOSE BRACKET operator. This binary CLOSE BRACKET operator is corrected by the program preparation procedure and converted to a CLOSE WORD BRACKET. PA computes the destination addresses for the branch modules and run-time blocks and the historical values to be skipped. From PLC version 6.0 onwards, the PA command is called automatically each time the program is started (G command).

Output of the user program as a hex file

Command:



Function:

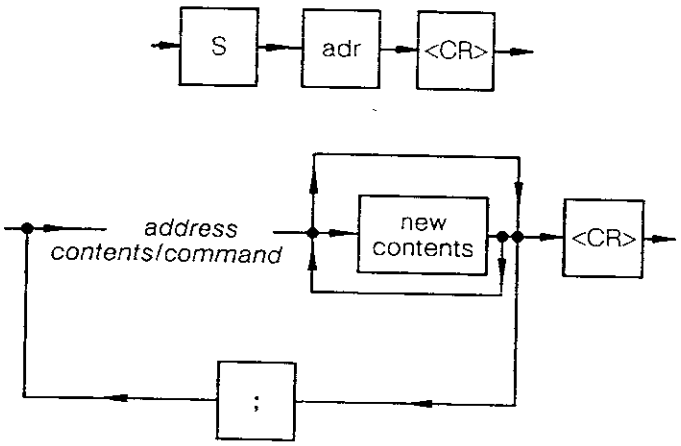
The user programs or comments can be output as a hex file through the bottom serial interface 1 of the PLC. The user can therefore load the user programs or comments into a EPROM programmer and program EPROMs. This is done in a dialog between the user and the PLC. The procedure can be aborted by pressing <CTRL> C. See Chapter "Serial interfaces" for details of the format of the hex file and the data of the interface.

Note:

If a user program includes the module DRUCK or EMAS, which use the same interface as the PU command, both user programs have to be aborted *before* executing the PU command.

Enter/edit (substitute) user program

Command:



adr: Program memory address from which the program is to be entered or edited

address: The program memory address whose contents are to be altered is displayed by the PLC

contents: Applies only to block calls. The back-translated contents of the program memory address are displayed

command: Applies to blocks and the module header (number and type). The back-translated command or module header is displayed, always as a complete command, i.e. operand and operator or module call and module type. If an address is entered which does not point to a command start or a block call, the software corrects it to an address pointing to the command start.

new contents: New contents of the user program

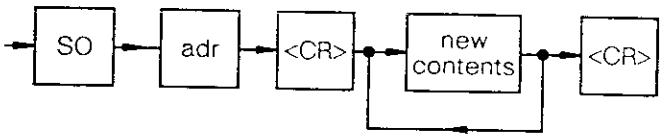
:: After input of a semicolon, the subsequent program memory address and its contents are displayed and can be altered if required. If no new CONTENTS are entered before the semicolon, the old contents of the displayed program memory address remain unchanged.

Function:

A program memory word is selected and is displayed as instruction or operand. The display contents can then be overwritten.

Enter/edit user program without echo

Command:



adr: Program memory address from which the program is to be entered or edited.

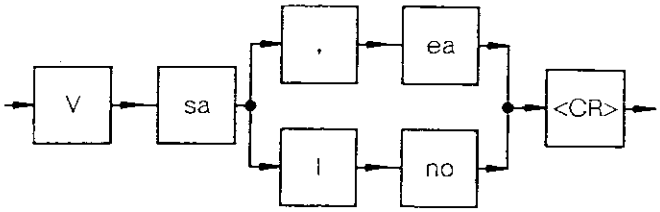
new contents: New contents of the user program

Function:

The program memory address is preset from which the program is to be entered. The program can then be entered consecutively. The PLC does *not* return an echo of the entered program. In the event of an error, however, the PLC returns an error message (e.g. invalid input).

Move user program

Command:



sa: Start address of the program part to be moved

ea: End address

l: Length (key word)

no: Number of program memory words by which the program part is to be moved

Function:

The program is moved from address 'sa' to address 'ea' or from address 'sa' by the specified number of program memory words. The resulting gap is filled up with NOPs. New program parts can be inserted in this gap. Moving a program part is possible only if the necessary space is still available at the end of the user program. This is, however, checked automatically.

Examples:

V 0,10 <CR>

The program is moved from address 0 to address 10. NOPs are inserted from address 0 up to address 9.

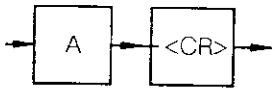
V 10 L 20 <CR>

The program is moved by 20 program words from address 10 to address 30 and 20 NOPs are inserted.

8.2 Commands for testing the user program

Abort user program

Command:

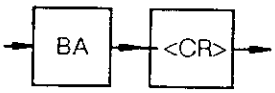


Function:

The user program is aborted. All outputs (binary and word) are set to zero. The user program can be restarted by entering "G".
Started timers continue to run in the operating system regardless of the program status: They are aborted only by a warm start/reset.

Display breakpoint

Command:

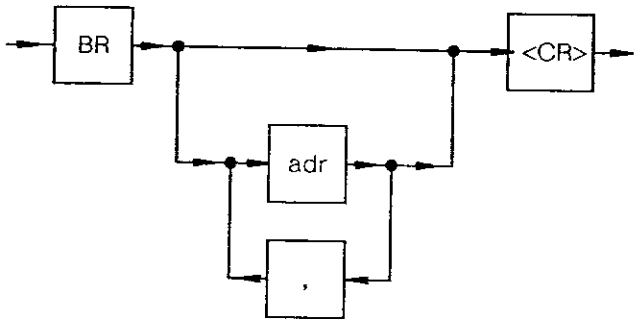


Function:

All breakpoints of the program are displayed. In a command, however, it is not the breakpoint address which is displayed but the address where the command starts and its contents.

Reset breakpoints

Command:



adr: Address of the breakpoint to be deleted

,: If only specific breakpoints are deleted, the individual addresses must be separated during input by a comma.

Function:

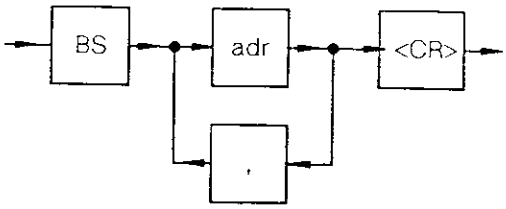
The breakpoints can be deleted individually. The command of

BR <CR>

is used to delete all breakpoints of the program.

Set breakpoint

Command:



adr: Address of the breakpoint.

,: If several breakpoints are set, the addresses must be separated by a comma on input.

Breakpoints can be set:

- to the address of the operand *following* an allocation character
- to the address of a CLOSE BRACKET
- to the address of the last parameter of a block
- to the address of the program end

Function:

After it has been started, the program stops at the first breakpoint. Breakpoints can also be entered while the program is running. Up to 15 breakpoints may be preset.

Switching to the next breakpoint: by input of a *semicolon* the program runs after expiry of the cycle time or after the next event up to the next breakpoint and displays the program address and the command located there. If the next breakpoint is *not* reached, owing to a long cycle time or failure of the next event to occur, display can be aborted, if required, by entering <CTRL>C.

Note:

If the TURBO mode is configured, the PLC switches automatically to NORMAL mode when a breakpoint is set and the cycle time is multiplied by four. This is done because breakpoints can only be processed in NORMAL mode. When the breakpoint is deleted, the PLC automatically switches back to TURBO mode and the cycle time is again set to the value of KD 0,0. This internal PLC measure normally does *not* manifest itself externally to the user.
Exception: The breakpoint is set at a point in the program which, as the result of a jump, for instance, is *not* processed. In this case, the program continues to run but with four times the cycle time, which may detrimentally affect the functions.

Switch-over between
operator control functions <-> monitor

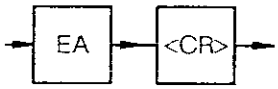
Command:

Function:

You can switch into the PLC's monitor program by simultaneously pressing the <CTRL> and W keys. Here, a few basic functions are placed at the user's disposal at the monitor level. When you are in the monitor, you can return to the PLC's operator control program by entering <CTRL> and W. For details, see chapter entitled "Monitor functions".

I/O test

Command:



Function:

In this mode of operation, the user can check the wiring of his I/O signals from the PLC up to the process for correctness.

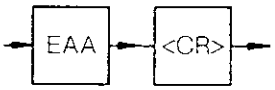
The user program is *not* processed after it has been started. Only the I/O signals planned in the program are handled. That is to say, the input signals are read in and the output signals are output.

By actuation of limit switches etc., it is possible to check whether the signals arrive in the PLC under the agreed designations. By specifically setting outputs, it is possible to check whether the signals arrive at the correct locations in the process. The required I/O variables can be displayed in the PLC by means of the commands Z or ZD.

The command "EA" can also be entered while the program is running. Then, this mode of operation does not become effective until the next program cycle begins.

Terminate I/O test

Command:

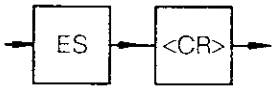


Function:

The I/O test mode is deactivated with this command, i.e. the user program continues to run as normally. It is expedient to abort the program before deactivating the I/O test.

Single step ON

Command:



Function:

After starting the program, only one sentence or block is executed and the program stops after the each allocation, CLOSE BRACKET or at the end of the block.

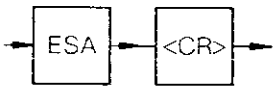
Variable values can be displayed by means of the Z command.

The "ES" command can also be entered while the program is running. Then, this mode of operation does not become effective until the next program cycle begins.

Switching to the next step:
By input of a *semicolon*, the program runs after expiry of the cycle time or after the next event up to the next stop point and displays the program address and the command located there. If the next stop point is not reached, owing to a long cycle time or failure of the next event to occur, display can be aborted, if required, by entering <CTRL> C.

Single step OFF

Command:

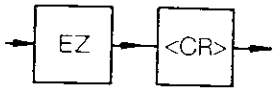


Function:

Single step mode is terminated, i.e. the user program continues to run as normally as from the current stop point.

Single cycle ON

Command:



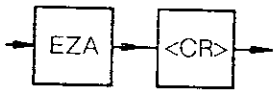
Function:

When the program is started, the program stops at its end. The "EZ" command can also be entered while the program is running. In this case, this mode of operation does not become effective until the next program cycle begins.

Switching to the next program cycle:
By input of a *semicolon*, the program runs after expiry of the cycle time or after the next event *once* and displays the program address and the command (!PE) located there. If the next stop point is *not* reached, owing to a long cycle time or failure of the next event to occur, display can be aborted, if required, by entering <CTRL> C.

Single cycle OFF

Command:

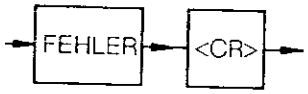


Function:

The single cycle mode is terminated, i.e. the program is again executed as normally.

Display of the error register's contents

Command:



Function:

The contents of the error register (1000H:6) are read out. See Chapter "Self diagnosis and reaction to errors" for details of error numbers.

Forcing I/O signals

With the PLC the user can "force" input and output signals. In doing so, values are preset by the user for I/O signals. Instead of using the actual input signals, the PLC then operates with the force values. The PLC does not issue the output signals computed in the PLC program to the output modules, but again, the force values.

These force values apply until forcing is cancelled for individual I/O signals or for all I/O signals. Both the values supplied by input modules and also the values assigned to outputs in the PLC program therefore do not have *any* influence during forcing. Forcing can be applied both to binary input/output signals and also to word input I/O signals.

Maximum number of I/O signals to be forced:

- Binary inputs: 64
- Word inputs: 16
- Binary outputs: 64
- Word outputs: 16

Forcing is carried out in the following way:

Forcing inputs

At the start of the program cycle, the PLC creates an image of the input signals planned in the PLC program. If inputs are to be forced, their actual values are substituted by user-specified force values after they have been read in. During the program cycle, the PLC operates exclusively with the modified input image and this means signal changes in the input module play no role whatever during the program cycle.

Forcing outputs

At the end of the program cycle, the PLC transfers the output image of the output signals planned in the PLC program to the output modules. If outputs are to be forced, their real values are substituted by the force values *before* they are output in the output image.

Response to power failure, RESET or warm start

After a power failure, RESET or warm start, the PLC will have "forgotten" the force order. The list of I/O signals to be forced which has been entered *before* the power failure, RESET or warm start does, however, still exist in the PLC and can also be displayed by entering the command FORC A <CR>. By entering one single signal to be forced, the *complete* force list is reactivated and forcing again becomes active.

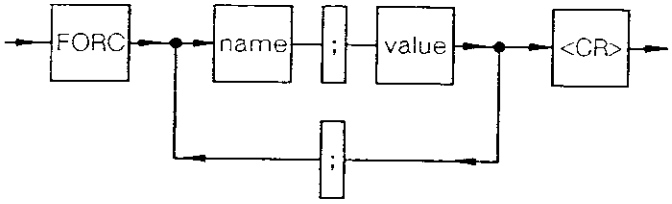
The following commands are available for forcing I/O signals:

- FORC: Enter force value
- FORC A: Display force value
- FORC R: Delete forcing

Enter force value

The name of the I/O signal to be forced and the force value are entered by means of the FORC command.

Command: FORC enter force value



name: Name of the input or output to be forced

value: Force value for the input or output

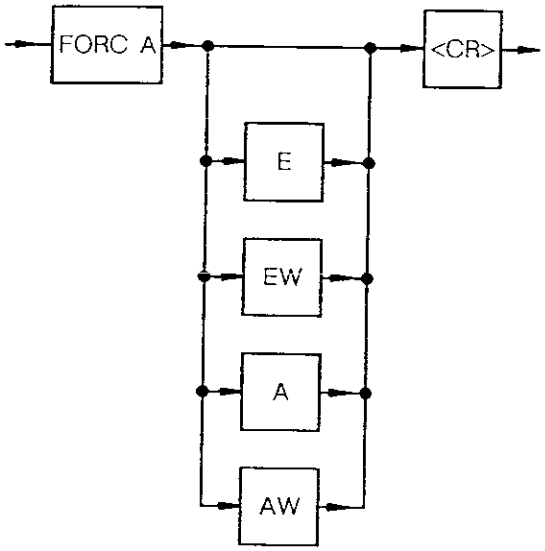
;: A semicolon is placed between the name and the force value as a delimiter. If several inputs/outputs are to be forced, these must also be delimited by a semicolon.

Function:

Input of the I/O signals to be forced and their values. The information concerning which inputs/outputs are to be forced is stored in a power fail-safe manner in the PLC's operand memory.

Display force value

Command:

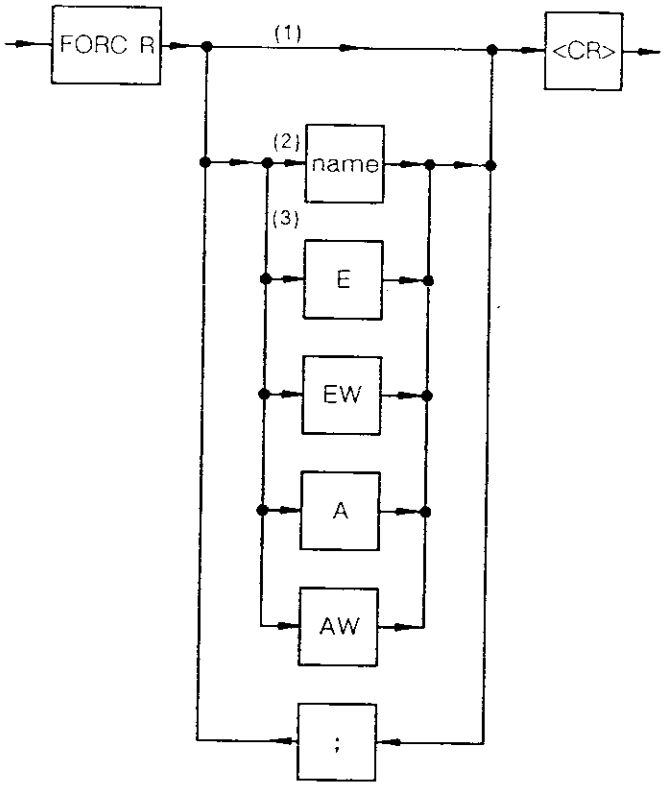


Function:

- Display of all inputs and outputs to be forced
- Display of all inputs and outputs to be forced of a specific group of inputs/outputs.

Delete forcing

Command:



name: Name of the inputs/outputs for which forcing is to be terminated

;: If forcing is terminated for specific inputs/outputs only, the individual names must be separated by a semicolon during input.

Function:

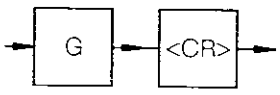
(1) Terminates forcing for all I/O signals

(2) Terminates forcing for *individual* I/O signals

(3) Terminates forcing for *one* specific group of I/O signals

Start user program

Command:



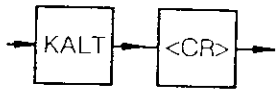
Function:

The user program is started.

If the INI-OPS mode is configured, the program's operand area is initialized with "zero" (see also OPS_INI).

Carrying out a cold start

Command:

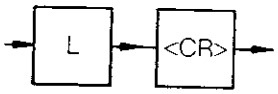


Function:

A cold start is carried out for the PLC. This command must not be issued together with the $ function through the MPST bus, but only through the serial interface of the PLC for which the cold start is to be carried out. After a cold start, the program is in its basic state, i.e. all memories are initialized and the default configuration of the modes and the data block directory is set. A cold start is also realized automatically on power-on for the first time (see also chapter entitled "Starting behaviour").

Continue user program

Command:

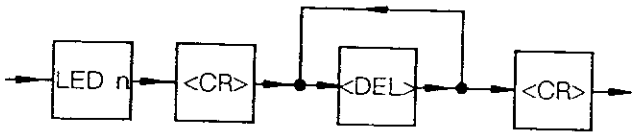


Function:

The user program is continued after a previous stop ("W"). On continuation, the flags have the same value as when the program was stopped.
Started timers continue to run in the operating system regardless of the program status. They are aborted only by a warm start/reset.

LED ON/OFF on the front panel of the PLC

Command:



n: 1 ... 3, number of the LED on the front panel of the PLC (LED 1 is the top LED)

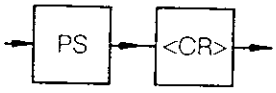
DEL: ASCII character DEL = $7F_H$, (on PCs, <CTRL> + delete key)

Function:

On the PLC's front panel, the selected LED is switched ON/OFF. The first time the <DEL> key is pressed, the selected LED is switched off and the state of the LED is toggled each time the key is pressed again.

Display program status

Command:

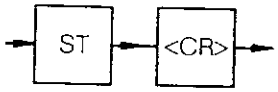


Function:

The program status (program at breakpoint, program aborted, program stopped, program running) of the respective user program is displayed.

Display PLC status

Command: (Available from version V7)



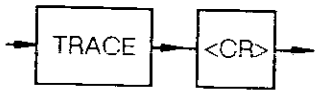
Function:

The complete PLC status is displayed as follows:

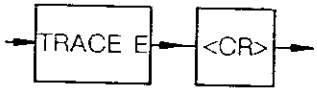
Current program No.
Program identification
Cycle time
Program status
Active test functions
Configuration of modes
TRACE register
Error register
Capacity utilization

TRACE mode

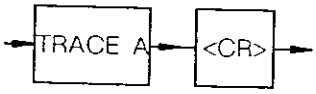
Command: Display TRACE memory



Command: Activate TRACE mode



Command: Deactivate TRACE mode

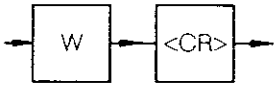


Function:

In TRACE mode, the PLC stores the address of the block last executed or of the instruction last executed. Therefore, after a system crash you have information about the extent to which the user program was executed. The TRACE memory contents are retained in the event of RESET.

Stop user program

Command:

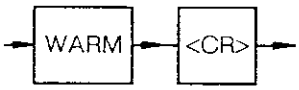


Function:

The user program is stopped. The values of outputs and flags are retained.
Started timers continue to run in the operating system regardless of the program status. They are aborted only by a warm start/reset.

Carrying out a warm start

Command:



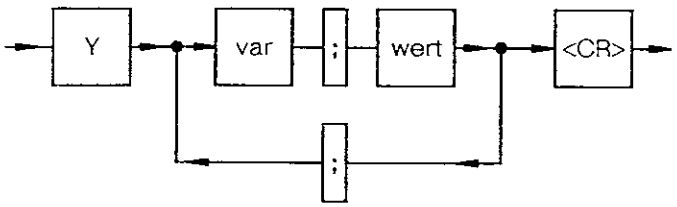
Function:

A warm start is carried out for the PLC. This command must not be issued together with the $ function through the MPST bus, but only through the serial interface of the PLC for which the warm start is to be executed.

When a warm start is executed, the program memory's contents and the configuration of the modes and the data block directory remain unchanged. A warm start is also executed automatically when "power ON" or after a hardware reset. See also Chapter entitled "Starting behavior".

Overwrite value of a variable with a value to be preset

Command:



var: Name of the variable or indirect constant

value: New value to be assigned to the variable

;: A semicolon must be placed between the name and value of the variable. If several variables are to be overwritten, these must also be separated by semicolons.

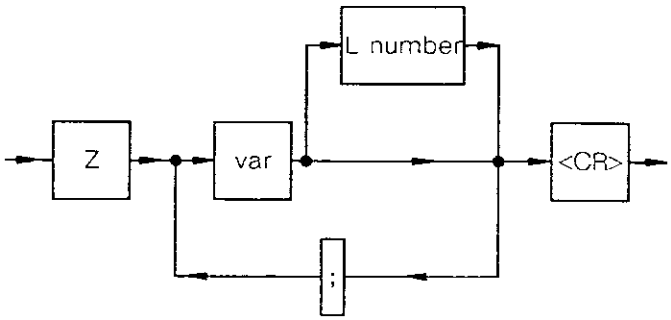
Note:

If the variable is a step variable, it can only be set, but not reset. When setting the step variable, all other steps of the chain are reset automatically.

If this command is used to alter an indirect constant, this change is realized in the operand memory only and not in the program memory. That is to say, this value is again overwritten by the value from the program memory after the next RESET or program start.

Display status of variables

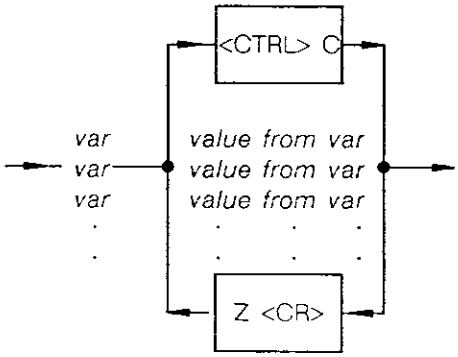
Command:



var: Variable (flag, step, input, output, indirect constant) to be displayed.

;: The individual variables must be separated by semicolons.

L number: Number of consecutively numbered variables from the variable var which are to be displayed. Example: M 0,0 L 3 displayed: M 0,0 M 0,1 M 0,2



Z: The values of the variables (max. 22) are each updated when the character Z <CR> is entered.

Function:

The variable names preset by the user are displayed. The values of these variables are updated after each input of the character Z <CR>. The displayed variable values always originate from the same program cycle and represent the current status at the end of the cycle.

The number of variables to be displayed is limited to 22 with this command because no more screen lines are available.

Computer connection instead of terminal

If a computer is connected for evaluation of the status values instead of the terminal, if required, the following commands can also be used instead of Z (the syntax diagram is the same as for the Z command):

ZO: The maximum number of variables is 120, otherwise see command Z.

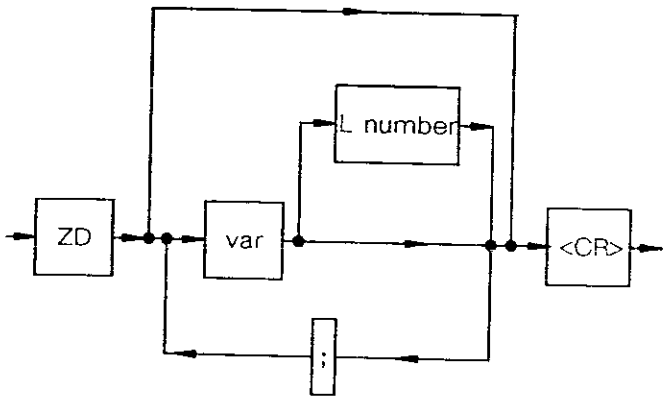
Screen control: With the commands of Z, ZO and ZD, the PLC uses the following control characters for screen control:

Carriage return: <CR>
Line feed: <LF>
Delete screen: <ESC>[2J
Position cursor: <ESC>[<line>;<column>H

ZZ: Maximum number of possible variables is 120. The PLC sends no ESC sequences for screen control but only the variable values, each followed by a <CR>. The variable values have the same sequence as the preset variable list. Otherwise it is the same as the Z command.

Display status of variables and update continuously

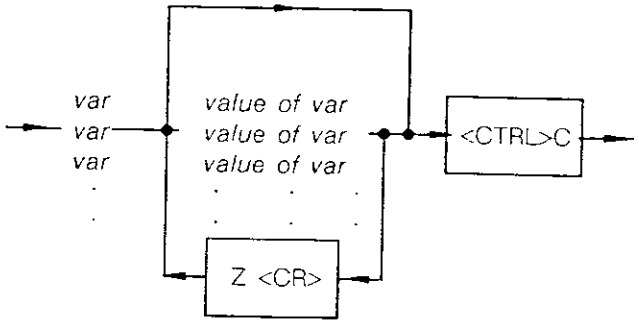
Command:



var: Variable (flag, input, output, indirect constant) to be displayed

;: The individual variables must be separated by a semicolon

L number: Number of consecutively numbered variables from the var variables, which are to be displayed. Example: M 0,0 L 3. Displayed: M 0,0 M 0,1 M 0,2



Function:

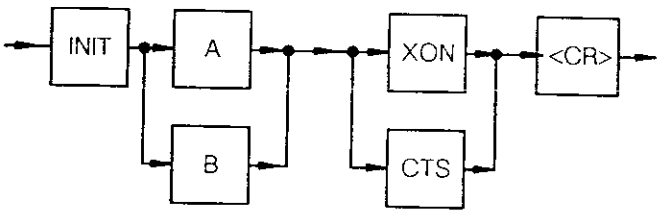
The variable names preset by the user are displayed. The corresponding variable values are updated automatically. The displayed variable values always originate from the same program cycle and represent the current status at the end of the cycle.

The maximum number is 22. The command is terminated by entering <CTRL>C.

When the character of Z <CR> or ZD <CR> is then entered, the status display for the previously entered variables is re-activated.

Initialization of the serial interfaces

Command:



A: Top operating serial interface of the PLC

B: Serial interface 1 (bottom interface of the PLC)

XON: Synchronization with the characters of XON or XOFF

CTS: Synchronization with the hardware signals of CTS (Clear To Send) and RTS (Ready To Send) and additionally with the characters of XON/XOFF.

CR: Carriage Return

Function:

The command applies only to the PLC and not for an additional module 35 DS 91. If the block of DRUCK or EMAS is connected to the bottom interface 1, this command is forbidden for this interface. The interface for the block of DRUCK or EMAS must always be initialized with the SINIT block. If a further 35 DS 91 unit is connected to the PLC, its interfaces are *not* influenced by this command.

Synchronization of the two interfaces and the connected module takes place as follows:

- by way of the hardware signals CTS and RTS, in which case the RTS line of the connected module is linked to the CTS input of the PLC. The RTS line of the PLC is linked to the CTS input of the connected module.
- additionally via XON/XOFF

Note:

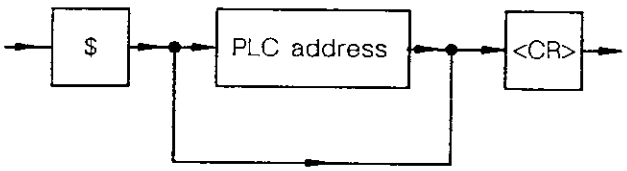
The PLC itself does not send XON/XOFF, but reacts to these characters when they are sent by the connected module (e.g. printer, terminal etc.).

As from version 5.0, basic initialization of the interface is set to RTS/CTS (see also Chapter serial interfaces). On re-initialization to XON/XOFF, the PLC *no longer* reacts to the RTS signal from the connected module. It does, however, continue to generate its own RTS signal.

See also Chapter "Serial interfaces"

Operation and test of several controllers through o n e serial interface

Command:



PLC address: Address of the PLC with which the user wishes to communicate

CR: Carriage Return

Function:

See also the chapter "Operating the PLC via the MPST bus".

This command is used to establish or abort a communication path from a PLC A to a PLC B.

The terminal is connected to any chosen PLC in the subrack. After the address of the destination PLC has been entered, all further entered commands are sent through the MPST bus to the selected destination PLC and their replies are fetched through the MPST bus and output through the serial interface. This, however, applies only if this PLC also exists and is free. The destination PLC is considered to be free when its current communication interface is the serial interface. If, however, it is currently being operated through the MPST bus, it is considered to be occupied and the communication path is not established.

When establishing the communication path, the communication interface of the destination PLC is switched over from the serial interface to the SENTENCE structure in its transfer RAM.

If the command is entered *without* specifying a module address, any previously established communication path is cleared again. This takes place by virtue of the fact that operation of the destination PLC is again switched over to its serial interface. All commands entered from this time on now again refer to the PLC to which the terminal is connected.

Important:

The commands WARM (warm start) and KALT (cold start) are forbidden when using the $ function. When using these commands, the transfer RAM of the destination PLC is initialized and the communication path as well as the RAM test running in the destination PLC for the transfer memory is disturbed. Then the destination PLC indicates a RAM error.

8.3 Commands for configurating the PLC

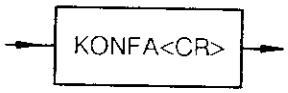
Configuration of the PLC by the user comprises the two abilities:

- setting modes
- editing the data block directory

Configure modes

Display of the configured modes

Command:



Function:

The configured operating modes are displayed.

After a cold start, the following modes are set on the PLC by way of default:

- GERMAN
- MASTER PLC
- AUTO PLC
- PTG MODE
- COPY AWP (user program)
- 120 KB
- CHANGE ABORT
- NOSENSOR
- NORMAL
- TIME CYCLE
- INI-OPS
- PRIO HIGH for program 1
- PRIO LOW for program 2

Display/change modes

Command:

Refer to the next but one page for an illustration

Function:

- Both PLC programs must be in the *aborted* state.
- After input of the KONFS <CR> command the configured mode of the first mode couple is displayed on the screen. By pressing the <DELETE> key (on PCs <Ctrl> and delete key) the identifier for the complementary mode is entered into the organization directory and this mode is displayed. By pressing the semicolon key the configured mode of the next modifiable mode couple is displayed. The command is terminated when the <CR> key is pressed.

Note:

With PCs often the DELETE key is missing. In this case, the key code (7F) of the DELETE key can be generated by pressing two keys. Regularly those are the <Ctrl> key and the delete key.

Mutually exclusive modes

The modes shown in *one* box in the illustration are mutually exclusive, i.e. either one or the other mode can be configured. The user need not concern himself with this exclusivity principle because this is guaranteed automatically by the configuration command.

Coupled modes

Some PLC modes are coupled to each other, i.e they can*not* be set independently of each other. This manifests itself as follows:

- When a mode is set to which others are coupled, the PLC automatically sets it to the correct parameters and these can also no longer be changed by the user.
  Example: when CHANGE RUN is configured, the PLC automatically sets the NO SENSOR and TIME CYCLE modes.
- In order to set specific modes, it is necessary for another one to have been set to a specific parameter beforehand.
  Example: if CHANGE RUN is to be configured, this can only be done if the large program memory (120 kbytes) has been set beforehand.

The *next modifiable* mode is always selected by entering a semicolon when setting the KONFS command. When this is done, coupled modes which must not be modified are displayed, but are skipped by the cursor and are therefore not modifiable.

- The following modes are coupled to each other:

  - SENSOR
    ---> NORMAL for program 2 and
    ---> EVENT for program 2

  - 56 kbytes
    ---> CHANGE ABORT

  - CHANGE RUN
    ---> NO SENSOR and
    ---> TIME CYCLE and
    ---> 120 kbytes

Effectiveness of mode changes

The following mode changes take effect immediately:

- GERMAN <--> ENGLISH
- 120 KB <--> 56 KB
- CHANGE ABORT ---> CHANGE RUN

The following mode changes take effect when PLC programs are started:

- NO SENSOR <---> SENSOR
- NORMAL <---> TURBO
- TIME CYCLE <---> EVENT
- INI OPS <---> NOINI OPS
- PRIO HIGH <---> PRIO LOW

The following mode changes do not take effect until after a warm start, i.e., WARM command or reset or power OFF/ON:

- AUTO PLC <---> PLC WITH ZST

  When PLC WITH ZST *and* PLC MODE are both set, a warm start cannot be executed by means of the warm start command because, in this case, the PLC is started up by the central control unit (ZST). The PLC waits for initialization by the ZST if a warm start command is entered inadvertently. This waiting state can be aborted by entering the <Ctrl>C command.

- PTG MODE <---> PLC MODE
- COPY AWP <---> NOCOPY AWP

The following modes do not take effect until after the DBV RES command (resetting the data block directory) followed by a warm start/reset:

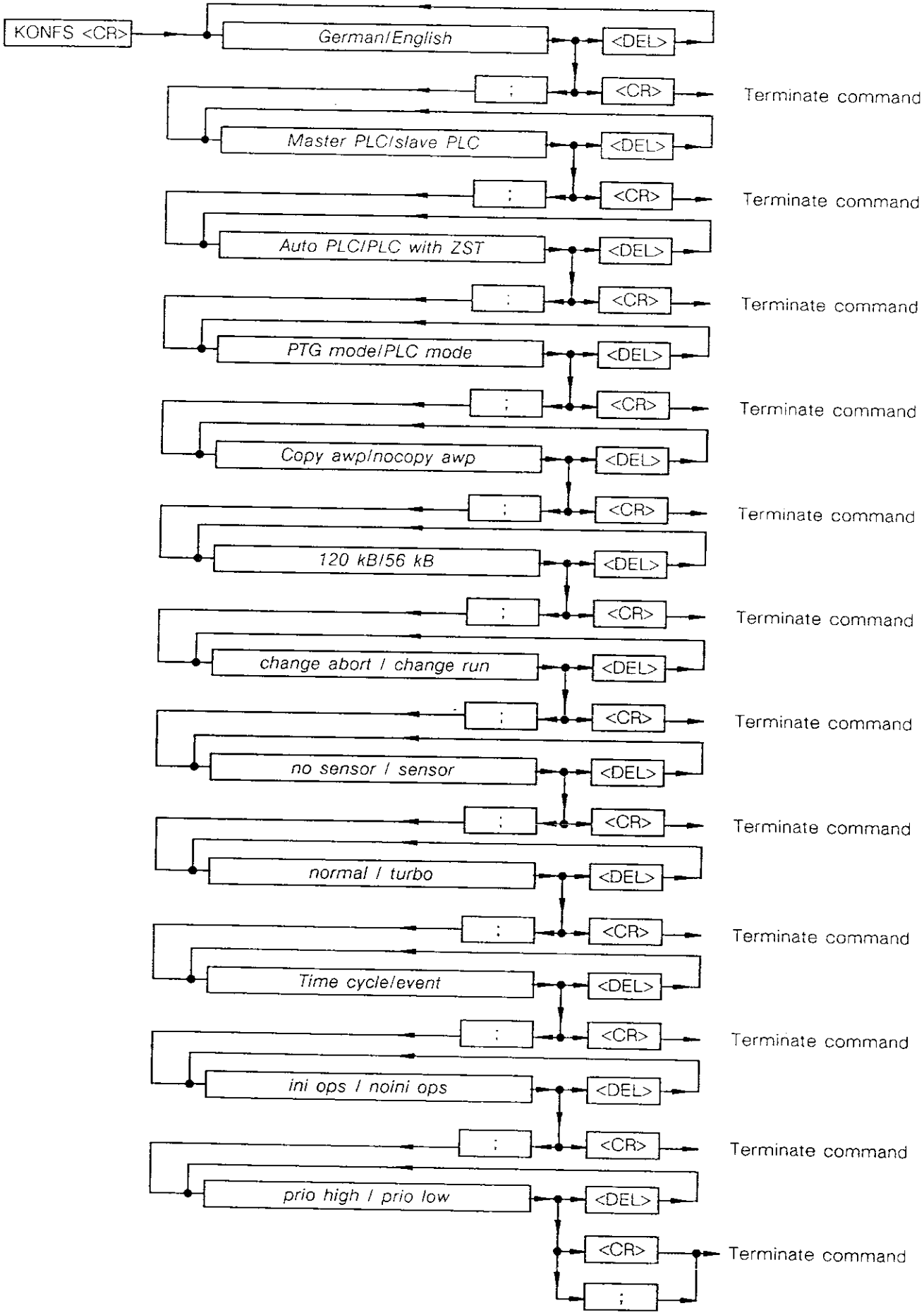
- MASTER PLC <---> SLAVE PLC

The following mode change can only be achieved by a cold start:

- CHANGE RUN ---> CHANGE ABORT

Displaying/modifying modes

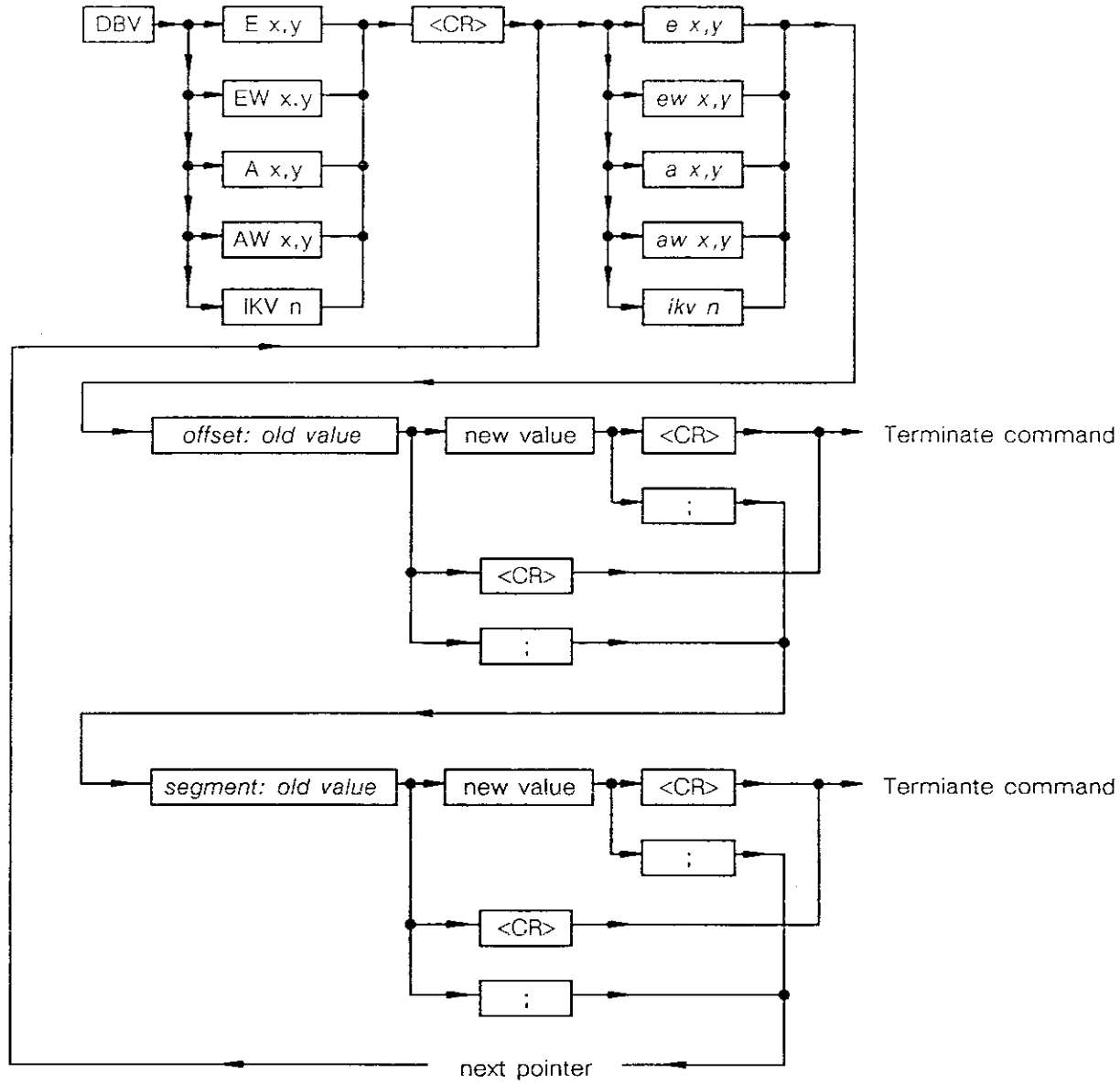
Command:



Configurating the data block directory

Displaying/changing the data block directory

Command:



User inputs:

- Key words: UPPER CASE LETTERS
- Addresses, values: lower case letters
- System outputs: *lower case letters in italics*

Function:

The user has the possibility of allocating, in the software, his I/O signals planned in the user program to any chosen physical address. For binary I/O signals, this statement applies in each case to *one* I/O group, e.g. E 0, 0 ... E 0,15. This allocation takes place in a data block directory original, which is stored in the **user program memory** in a manner preventing destruction in the event of a power failure (refer also to the chapter entitled 'Data block directory').

Important:

Both reconfiguration and resetting of the DBV (data block directory) to the default configuration is not activated until a warm start is executed (by means of the WARM command or a reset or power OFF/ON). The warm start transfers the original of the data block directory from the program memory to the transfer RAM.

- In the PLC WITH ZST mode, the Central Control Unit (ZST) can modify the data block directory in the **transfer RAM** when starting up the PLC. To do this, the ZST either finds the default DBV or a DBV altered by the user as the basis for the modification. Such a modification of the DBV in the transfer RAM is rendered ineffective again the next time a warm start of the PLC takes place.

Resetting the data block directory

Command:

Function:

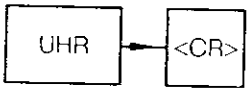
The original of the data block directory in the program memory is reset to the default configuration (see also chapter entitled 'Data block directory').

Important:

Both reconfiguration and resetting of the DBV to the default are not activated until a warm start is executed (WARM command or reset or power OFF/ON). The warm start transfers the original data block directory from the program memory to the transfer RAM.

Time and date display

Command:



Function:

The time and date are displayed as follows:

SYSTEM TIME: HH:MM:SS
SYSTEM DATE: DAY OF WEEK DD.MM.YY

where:
HH: hours
MM: minutes
SS: seconds

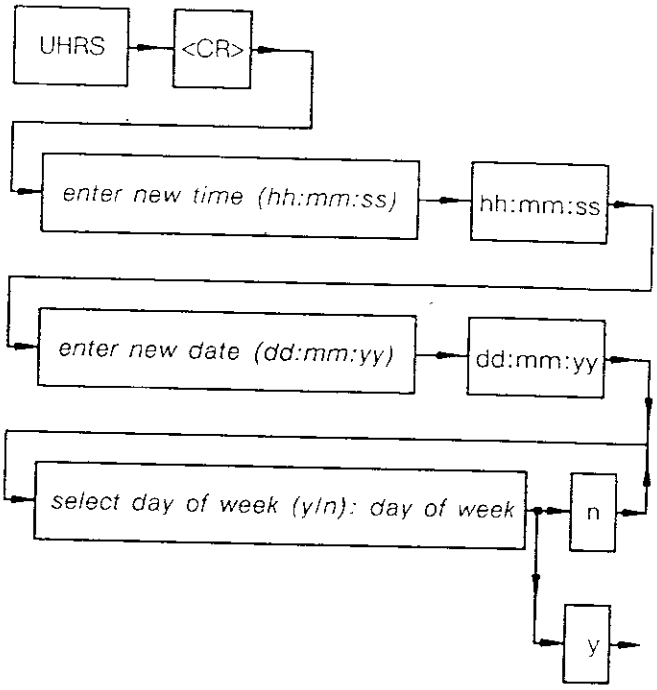
DAY OF WEEK: name of the day of the week
DD: day
MM: month
YY: year

Note:

The clock module on the PLC is only active if jumper 5...6 is fitted on the X17 connector panel.

Setting the time and date

Command:



Function:

To set the date and time: the clock internally manages a number between 1 and 7 for the day of the week. When converting the number to the name, it assumes that Monday is the first day of the week (number 1 ---> Monday). If the timer is set with the block CLOCK (see also block catalog), a different number can be assigned to Monday. In this case, the displayed day of the week no longer agrees with the CLOCK <CR> command because the display function fundamentally assumes that the number 1 is assigned to Monday.

hh or hh: hours
mm or mm: minutes
ss or ss: seconds
dd or dd: day
mm or mm: month
yy or yy: year
day of week: name of the weekday
n: input for 'no'
y: input for 'yes'

Note:

The clock module on the PLC is only active when the jumper 5...6 is fitted on the X17 connector panel.

9 Data block directory (DBV)

9.1 Functions of the data block directory

As specified in the MPST standard DIN 66264, the operands with the operand identifiers E and A (inputs/outputs) are assigned physical devices or memory locations by the data block directory. Therefore, the data block directory is a marshalling list between the I/O operands of the PLC program and the allocated physical addresses. It is located in the PLC's transfer memory and is therefore also accessible from the MPST bus.

9.1.1 Default data block directory

Each PLC has a *default data block directory* which is allocated to all I/O signals of the PLC addresses of modules or memory locations. This default marshalling for the I/O modules, the global values and the superglobal values is described in the following chapters:

- "Communication with I/O modules and preprocessors"
- "Communication between the two PLC programs"
- "Multi-processor operation".

The station address at which an I/O module is to be addressed must be set by means of the address switches on the I/O modules.
The default data block directory is stored battery-backed in the user program RAM and is copied to the transfer memory when the PLC is switched on.

9.1.2 Modification of the default data block directory

The majority of users will always work with the default data block directory and will not modify it.
If required, however, the default data block directory can be modified very easily by the user if, in a multiprocessor system, for instance, more I/O signals are needed than are available on a PLC. This change is made with the PLC command DBV (see chapter entitled "Operating and test functions") or by means of the ABB Procontic programming system.

The user has the possibility of allocating, in the software, his I/O signals planned in the user program to any chosen physical address. For *binary* I/O signals, this applies to *one I/O group* in each case. A binary I/O group consists of 16 binary signals, e.g. E 0,0 ... E 0,15 or A 3,0 ... A 3,15.

Therefore, if required, the I/O signals can be remarshalled to:

- any I/O modules
- any Passive Stations
- any Active Stations (transfer memory)
- any internal memories of the PLC

The address area on the MPST bus is subdivided into an area for

- active bus stations and
- passive bus stations.

From the PLC, these two areas are addressed as follows:

- Area for active bus stations:
  3000_H:XXXX_H
  XXXX_H is the active station's address
- Area for the passive bus stations:
  2000_H:XXXX_H
  XXXX_H is the passive station's address.

The command DBV <CR> acts on the default data block directory stored in the PLC's battery-backed program memory. This data block directory, which the user can modify if required, is also stored in the user EPROM.

Every time a warm start is executed (power OFF/ON, reset or the PLC command WARM), the modifiable data block directory is copied out of the user program RAM into the transfer memory and therefore takes effect.

9.1.3 Resetting the data block directory to the default configuration

The modified data block directory can be returned to the default configuration by means of the PLC command DBV RES <CR>.

Important:

Both the new configuration and also reset to the default configuration do not take effect until a warm start is executed (power OFF/ON, reset or the PLC command WARM).

9.2 Location and structure of the data block directory in the transfer memory

The data block directory in the *transfer memory* can be altered through the MPST bus by another active station, e.g., a central control unit (ZST), another PLC or an industrial computer.

A change to the data block directory in the *transfer memory* takes effect immediately. However, it is rendered inactive again the next time a warm start (power OFF/ON, reset or the PLC command of WARM) takes place.

The location and structure of the data block directory in the *transfer memory* are given below:

The data block directory consists of a list of pointers to physical devices or memory locations which are allocated to the I/O operands of the PLC. Each pointer consists of two words of 16 bits each:

- Segment address (16 bit value) and
- Offset address (16 bit value)

Therefore, in the data block directory each of these pointers occupies two words.

9.2.1 Start of the data block directory in the transfer memory

In the transfer RAM, the data block directory starts at address 50_H.

- When viewed from the PLC, this is the address:

  $1000_H:50_H$ (segment and offset).

- When viewed from the MPST bus in the area of the active bus stations, this is the address:

  $3000_H:XXXX_H + 50_H$
  XXXX_H is the PLC station address

9.2.2 Pointers to physical devices

The first pointer that points to an input module is located at the start of the data block directory, i.e., at address $1000_H:50_H$. The binary inputs of E 0,0 ... E 0,15 are allocated to it.

The second pointer that points to an input module is located to two words higher in the data block directory, i.e., at address $1000_H:54_H$ and the binary inputs of E 1,0 ... E 1,15 are allocated to it.

At the DBV address	the 32-bit pointer stands for	
Binary inputs		
1000_H:0050_H	Binary inputs	E 00,00...E 00,15
1000_H:0054_H	Binary inputs	E 01,00...E 01,15
.	.	.
.	.	.
1000_H:$014C_H$	Binary inputs	E 63,00...E 63,15
Binary global values		
1000_H:0150_H	Bin. global values	E 64,00...E 64,15
.	.	.
.	.	.
1000_H:$016C_H$	Bin. global values	E 71,00...E 71,15
Binary superglobal values		
1000_H:0170_H	Binary super-global values	E 72,00...E 72,15
.	.	.
.	.	.
1000_H:$01D0_H$	Binary super global values	E 96,00...E 96,15
Word inputs		
1000_H:$01D4_H$	Word input	EW 00,00
1000_H:$01D8_H$	Word input	EW 00,01
.	.	.
.	.	.
1000_H:0550_H	Word input	EW 13,15
Read axis boards		
1000_H:0554_H	Axis board 1, axis 1 Read status	EW 14,00
1000_H:0558_H	Axis board 1, axis 1 Read actual value	EW 14,01
1000_H:$055C_H$	Axis board 1, axis 1 Read measured value	EW 14,02
1000_H:0560_H	Axis board 1, axis 2 Read status	EW 14,03
1000_H:0564_H	Axis board 1, axis 2 Read actual value	EW 14,04
1000_H:0568_H	Axis board 1, axis 2 Read measured value	EW 14,05
1000_H:$056C_H$	Axis board 1, axis 3 Read status	EW 14,06
1000_H:0570_H	Axis board 1, axis 3 Read actual value	EW 14,07
1000_H:0574_H	Axis board 1, axis 3 Read measured value	EW 14,08
1000_H:0578_H	Axis board 1, axis 4 Read status	EW 14,09
1000_H:$057C_H$	Axis board 1, axis 4 Read actual value	EW 14,10
1000_H:0580_H	Axis board 1, axis 4 Read measured value	EW 14,11
1000_H:0594_H	Axis board 2, axis 1 Read status	EW 15,00
1000_H:0598_H	Axis board 2, axis 1 Read actual value	EW 15,01
1000_H:$059C_H$	Axis board 2, axis 1 Read measured value	EW 15,02
1000_H:$05A0_H$	Axis board 2, axis 2 Read status	EW 15,03
1000_H:$05A4_H$	Axis board 2, axis 2 Read actual value	EW 15,04
1000_H:$05A8_H$	Axis board 2, axis 2 Read measured value	EW 15,05
1000_H:$05AC_H$	Axis board 2, axis 3 Read status	EW 15,06
1000_H:$05B0_H$	Axis board 2, axis 3 Read actual value	EW 15,07
1000_H:$05B4_H$	Axis board 2, axis 3 Read measured value	EW 15,08
1000_H:$05B8_H$	Axis board 2, axis 4 Read status	EW 15,09
1000_H:$05BC_H$	Axis board 2, axis 4 Read actual value	EW 15,10
1000_H:$05C0_H$	Axis board 2, axis 4 Read measured value	EW 15,11
Word global values		
1000_H:$05D4_H$	Word global value	EW 16,00
.	.	.
.	.	.
1000_H:0650_H	Word global value	EW 17,15
Word superglobal values		
1000_H:0654_H	Word superglobal value	EW 18,00
.	.	.
.	.	.
1000_H:0792_H	Word superglobal value	EW 22,15
Binary outputs		
1000_H:0794_H	Binary outputs	A 00,00...A 00,15
1000_H:0798_H	Binary outputs	A 01,00...A 01,15
.	.	.
.	.	.
1000_H:0890_H	Binary outputs	A 63,00...A 63,15
Bin. global values		
1000_H:0894_H	Bin. global values	A 64,00...A 64,15
.	.	.
.	.	.
1000_H:$08B0_H$	Bin. global values	A 71,00...A 71,15
Binary superglobal values		
1000_H:$08B4_H$	Binary super-global values	A 72,00...A 72,15
.	.	.
.	.	.
1000_H:0914_H	Binary super-global values	A 96,00...A 96,15

At the DBV address	the 32-bit pointer stands for	
Word outputs		
1000_H:0918_H	Word output	AW 00,00
1000_H:$091C_H$	Word output	AW 00,01
.		.
.		.
1000_H:$0C94_H$	Word output	AW 13,15
Write axis boards		
1000_H:$0C98_H$	Axis board 1, axis 1 Write status	AW 14,00
1000_H:$0C9C_H$	Axis board 1, axis 1 Write setpoint value	AW 14,01
1000_H:$0CA0_H$	Axis board 1, axis 1 Write measured value	AW 14,02
1000_H:$0CA4_H$	Axis board 1, axis 2 Write status	AW 14,03
1000_H:$0CA8_H$	Axis board 1, axis 2 Write setpoint value	AW 14,04
1000_H:$0CAC_H$	Axis board 1, axis 2 Write measured value	AW 14,05
1000_H:$0CB0_H$	Axis board 1, axis 3 Write status	AW 14,06
1000_H:$0CB4_H$	Axis board 1, axis 3 Write setpoint value	AW 14,07
1000_H:$0CB8_H$	Axis board 1, axis 3 Write measured value	AW 14,08
1000_H:$0CBC_H$	Axis board 1, axis 4 Write status	AW 14,09
1000_H:$0CC0_H$	Axis board 1, axis 4 Write setpoint value	AW 14,10
1000_H:$0CC4_H$	Axis board 1, axis 4 Write measured value	AW 14,11
1000_H:$0CD8_H$	Axis board 2, axis 1 Write status	AW 15,00
1000_H:$0CDC_H$	Axis board 2, axis 1 Write setpoint value	AW 15,01
1000_H:$0CE0_H$	Axis board 2, axis 1 Write measured value	AW 15,02
1000_H:$0CE4_H$	Axis board 2, axis 2 Write status	AW 15,03
1000_H:$0CE8_H$	Axis board 2, axis 2 Write setpoint value	AW 15,04
1000_H:$0CEC_H$	Axis board 2, axis 2 Write measured value	AW 15,05
1000_H:$0CF0_H$	Axis board 2, axis 3 Write status	AW 15,06
1000_H:$0CF4_H$	Axis board 2, axis 3 Write setpoint value	AW 15,07
1000_H:$0CF8_H$	Axis board 2, axis 3 Write measured value	AW 15,08
1000_H:$0CFC_H$	Axis board 2, axis 4 Write status	AW 15,09
1000_H:$0D00_H$	Axis board 2, axis 4 Write setpoint value	AW 15,10
1000_H:$0D04_H$	Axis board 2, axis 4 Write measured value	AW 15,11
Word global values		
1000_H:$0D18_H$	Word global value	AW 16,00
.		.
.		.
1000_H:$0D94_H$	Word global value	AW 17,15
Word superglobal values		
1000_H:$0D98_H$	Word superglobal value	AW 18,00
.		.
.		.
1000_H:$0ED4_H$	Word superglobal value	AW 22,15
Data structure SENTENCE		
1000_H:$0ED8_H$	Data structure "Sentence" where the characters arriving from the MPST-bus are expected from the PLC. ($-function)	
1000_H:$0EDC_H$	Data structure "Sentence" where the PLC provides the characters for the operating commands via the MPST bus. ($-function)	
Iconic image processor (video-sensor OMS-F)		
1000_H:$0EE0_H$	IKV 1 (1. unit 35 IV 90, Iconic image processor)	
1000_H:$0EE4_H$	IKV 2 (2. unit 35 IV 90, Iconic image processor)	
1000_H:$0EE8_H$	IKV 3 (3. unit 35 IV 90, Iconic image processor)	
1000_H:$0EEC_H$	IKV 4 (4. unit 35 IV 90, Iconic image processor)	
1000_H:$0EF0_H$	IKV 5 (5. unit 35 IV 90, Iconic image processor)	
Reserve pointers		
1000_H:$0EF4_H$	Reserved	
.		
.		
1000_H:$0F10_H$	Reserved	

10 Communication with I/O modules and preprocessors

The ABB Procontic T300 system provides an extensive range of I/O modules and preprocessors with which the PLC can communicate.

10.1 Communication with I/O modules

The PLC processes the I/O signals via the process diagram, i.e. it does not access the assigned I/O modules during the program function.

Input signals

Before each program cycle, all input signals (operand identifiers E and EW) occurring in the program are written into the operand memory as an image.
This implies the advantage that consistent values of the input signals (actual values) can be used during the processing.

Output signals

During the program cycle, all output signals (operand identifiers A and AW) occurring in the program are written into a program-specific output image in the operand memory. The PLC outputs the output image at the the end of each program cycle. In multi-processor mode, *several* PLC processors must *not* write to the *same* binary output group because the different PLCs will otherwise mutually overwrite each others' output values on the output device (see also chapter entitled "Multi-processor operation"). Different programs of *one* PLC, however, may write to one binary output group. In this case, mutual overwriting is prevented by the PLC's operating system.
The above statements also apply to superglobal values.

Marshalling

All I/O operands are assigned to physical addresses (module addresses) by way of a marshalling list. In conformity with DIN 66264, this marshalling list is called the data block directory (DBV). Each PLC possesses a default data block directory which, if required, can be reconfigured very easily by the user (see also Chapter "Data block directory").

Enabling the I/Os

All I/O operands occurring in the user program are enabled for input or output by means of the PA command. I/O operands which are not planned in the user program are *ignored* by the PLC during input and output. As from PLC version V 6.0, the PA command is executed automatically with each program start.

I/O test mode

In this mode of operation, only the I/O signals are processed and the user program is not executed (see also "Operating and test functions").

Alarm inputs

The 35 EB 92 binary input module has 8 alarm inputs which may trigger a group alarm in the station which supplies the MPST bus clock. If this is a PLC, every alarm causes an event-controlled PLC program to run through precisely one program cycle. In this PLC program, the inputs occurring in it are then read in and can be evaluated.

Alarm vector

When an alarm is issued, the input module sends an alarm vector to the PLC. This alarm vector's value is set on the input module by means of the DIL switch S3.

Either program 1 or 2 is run in the PLC supplying the bus clock depending on the alarm vector's value.

Alarm vector	Reaction of the PLC
0...7 (0...7_H)	forbidden on 35 EB 92; this vector is allocated to the 35 IV 90 iconic image processor
8...23 (8_H...17_H)	Execution of a program cycle in PLC program 2
24...39 (18_H...27_H)	Execution of a program cycle in PLC program 1

After reception of the alarm vector by the PLC, it takes around 0.85 ms before a new cycle begins in the allocated cycle.

10.2 Default jumpering of the I/O signals and I/O modules

The data block directory (DBV) defines the I/O operands' assignments to the physical addresses. If required, the user can easily modify these assignments, but this is only necessary in rare cases.

With the default jumpering, physical addresses are:

- Addresses of I/O modules
- Addresses of global value memories
- Addresses of the super global value memory

Each I/O module address must be set on a DIL switch.

The following default marshalling applies if the user does not alter the data block directory:

Binary inputs

E 00,00...E 00,15	is jumpered to module address $E000_H$
E 01,00...E 01,15	is jumpered to module address $E002_H$
.	.
.	.
E 63,00...E 63,15	is jumpered to module address $E07E_H$

Word inputs

EW 00,00	is jumpered to module address $E100_H$
EW 00,01	is jumpered to module address $E102_H$
.	.
.	.
EW 13,15	is jumpered to module address $E2BE_H$

Binary outputs

A 00,00...A 00,15	is jumpered to module address $E080_H$
A 01,00...A 01,15	is jumpered to module address $E082_H$
.	.
.	.
A 63,00...A 63,15	is jumpered to module address $E0FE_H$

Word outputs

AW 00,00	is jumpered to module address $E300_H$
AW 00,01	is jumpered to module address $E302_H$
.	.
.	.
AW 13,15	is jumpered to module address $E4BE_H$

Binary input modules

The module addresses of all binary input modules (35 EB 90, 35 EB 91, 35 EB92) can be adjusted without interruption to this default jumpering of the PLC by means of their address switches.
The process image is registered automatically at the start of the program cycle.

Principle of functioning of the binary input modules: (Refer to the respective module description)

Binary output modules

The module addresses of all binary output modules (35 AB 90, 35 AB 94, 35 AB 95, 35 AB 96, 35 AB 97) can be adjusted without interruption to this default jumpering of the PLC by means of their address switches.
The process image is output automatically at the end of the program cycle. In multi-processor mode *several* PLC processors must *not* write to *the same* binary output group because the different PLCs will otherwise mutually overwrite each other's output values on the output device (see also the chapter entitled "Multi-processor operation"). Different programs *of one* PLC, however, may write to *one* binary output group. In this case, mutual overwriting is prevented by the PLC operating system.
The above statements also apply to super global values.

Principle of functioning of the binary output modules: Refer to the respective module description.

Analog input module of 35 EA 90 with 35 TP 90

Note:

The 35 EA 90 analog input module is **not** registered when creating the input image automatically. This module is served with the module of ANAEIN (see the block catalog).

The 35 EA 90 module is plugged onto the 35 TP 90 carrier board. 8 analog signals ranging from -10V...+10V or -5V...+5V can be connected. The 35 TP 90 carrier board can be used in any free address area within the input/output channels ($E000_H$...$FFFF_H$). On the MPST bus it occupies an address area of 256 bytes.

Principle of function of the analog input module: see the module description of 35 EA 90.

Analog input module of 35 EA 91 with 35 TP 90

The 35 EA 91 module is plugged onto the 35 TP 90 carrier board. 8 analog signals ranging from 4...20 mA can be connected. The 35 TP 90 carrier board can be used in any free address area within the input/output channels ($E000_H$...$FFFF_H$). On the MPST bus it occupies an address area of 256 bytes. The process image is registered **automatically** at the start of the program cycle (contrary to 35 EA 90). Addressing gaps occur.

However, it is possible to address the module by continuous EW operands by modifying the data block directory (DBV). For details of the principle of functioning of the analog input module: see the module description of 35 EA 91.

Pt 100 input module of 35 EA 92 with 35 TP 90

The 35 EA 92 module is plugged onto the 35 TP 90 carrier board. 4 Pt 100 analog signals can be connected. The 35 TP 90 carrier board can be used in any free address area within the input/output channels ($E000_H$...$FFFF_H$). On the MPST bus it occupies an address area of 256 bytes. The process image is registered **automatically** at the start of the program cycle. Addressing gaps occur. However, it is possible to address the module by continuous EW operands by modifying the data block directory (DBV). For details of the principle of functioning of the Pt 100 input module: see the module description of 35 EA 92.

Pt 100 input module of 35 EA 94

8 Pt 100 analog signals can be connected to the Pt 100 input module.
By means of address switches, the module address can be set without interruption to the default marshalling of the word inputs (E100H...E2BEH) of the PLC.

The process image is registered **automatically** at the start of the program cycle.
For details of the principle of functioning of the Pt 100 input module: see the module description of 35 EA 94.

Analog output module of 35 AA 92

The 35 AA 92 analog output module has four output channels.
The process image is output automatically at the end of the program cycle.

For details of the principle of functioning of the analog output module: see module description of 35 AA 92.

The 35 AA 92 analog output module can be used as follows:

- in the address area of the axis boards

The following applies when the module address of $E500_H$ is set:

AW 14,01 corresponds to analog output 1
AW 14,04 corresponds to analog output 2
AW 14,07 corresponds to analog output 3
AW 14,10 corresponds to analog output 4

The following applies when the module address of $E520_H$ is set:

AW 15,01 corresponds to analog output 1
AW 15,04 corresponds to analog output 2
AW 15,07 corresponds to analog output 3
AW 15,10 corresponds to analog output 4

- in the address area of the word outputs (E300H ... E4BEH)

The analog output module occupies an address area of 20_H bytes on the MPST bus. It is not possible to address the module by continuous output operands. Therefore, with the operand numbers gaps occur:

Example: If module address $E300_H$ is set:

AW 00,02 corresponds to analog output 1
AW 00,06 corresponds to analog output 2
AW 00,10 corresponds to analog output 3
AW 00,14 corresponds to analog output 4

Example: If the module address $E320_H$ is set:

AW 01,02 corresponds to analog output 1
AW 01,06 corresponds to analog output 2
AW 01,10 corresponds to analog output 3
AW 01,14 corresponds to analog output 4

Axis board of 35 AE 92

In the default configuration of the data block directory, the PLC can control two axis boards. Each 35 AE 92 axis board can operate up to four axes.

From the PLC, the user can directly handle the following signals of the axis board via I/O operands:

- Status
- Setpoint value
- Actual value
- Measured value

For details of the principle of functioning of the axis board: see the module description of 35 AE 92.

Default assignment for the axis board of 35 AE 92

Module address:

- Axis board 1: $E500_H$
- Axis board 2: $E520_H$

Axis board 1

EW 14,00 Axis 1: Read status register
EW 14,01 Axis 1: Read actual value
EW 14,02 Axis 1: Read measured value

EW 14,03 Axis 2: Read status register
EW 14,04 Axis 2: Read actual value
EW 14,05 Axis 2: Read measured value

EW 14,06 Axis 3: Read status register
EW 14,07 Axis 3: Read actual value
EW 14,08 Axis 3: Read measured value

EW 14,09 Axis 4: Read status register
EW 14,10 Axis 4: Read actual value
EW 14,11 Axis 4: Read measured value

AW 14,00 Axis 1: Write status register
AW 14,01 Axis 1: Write setpoint value
AW 14,02 Axis 1: Write measured value

AW 14,03 Axis 2: Write status register
AW 14,04 Axis 2: Write setpoint value
AW 14,05 Axis 2: Write measured value

AW 14,06	Axis 3:	Write status register
AW 14,07	Axis 3:	Write setpoint value
AW 14,08	Axis 3:	Write measured value
AW 14,09	Axis 4:	Write status register
AW 14,10	Axis 4:	Write setpoint value
AW 14,11	Axis 4:	Write measured value

Axis board 2

EW 15,00	Axis 1:	Read status register
EW 15,01	Axis 1:	Read actual value
EW 15,02	Axis 1:	Read measured value
EW 15,03	Axis 2:	Read status register
EW 15,04	Axis 2:	Read actual value
EW 15,05	Axis 2:	Read measured value
EW 15,06	Axis 3:	Read status register
EW 15,07	Axis 3:	Read actual value
EW 15,08	Axis 3:	Read measured value
EW 15,09	Axis 4:	Read status register
EW 15,10	Axis 4:	Read actual value
EW 15,11	Axis 4:	Read measured value
AW 15,00	Axis 1:	Write status register
AW 15,01	Axis 1:	Write setpoint value
AW 15,02	Axis 1:	Write measured value
AW 15,03	Axis 2:	Write status register
AW 15,04	Axis 2:	Write setpoint value
AW 15,05	Axis 2:	Write measured value
AW 15,06	Axis 3:	Write status register
AW 15,07	Axis 3:	Write setpoint value
AW 15,08	Axis 3:	Write measured value
AW 15,09	Axis 4:	Write status register
AW 15,10	Axis 4:	Write setpoint value
AW 15,11	Axis 4:	Write measured value

10.3 Preprocessors

The PLC is capable of communicating with preprocessors. These preprocessors can be connected as follows:

- to a serial interface
- to a binary output module
- to the MPST bus

Overview of the preprocessors

- Modules for process display and control
- 35 IR 93 industrial computer
- 35 KP 91 Communication processor for the process control system Master
- 35 KP 92 Communication processor for VERITRON PAD power converters
- Positioning modules for axes
- 35 ZB 91 Coupler for ABB Procontic field bus ZB 10
- 35 IV 90 Iconic image processor and 35 KI 90 affiliated camera interface as the sensor system (OMS-F)

This section is provided for general information only. Please refer to the respective manuals for the current data and the description of functions.

10.3.1 Process display and control

Today, machines, installations and industrial processes are increasingly being automated by means of programmable logic controllers (PLCs) and process computers. Process control and operating modules must be available to enable the operator to control such automated machines. This modules are linking elements enabling man/machine communication (MMC) between the operator and the machine. For process display and control, ABB offers a staggered assortment of modules, software and tools. The user can choose an optimum solution for his application from this assortment in terms in function and costs.

Overview of the preprocessors for process display and control

35 TA 10 Text display,
with 1 line x 40 character fluorescent display.
Connection via process outputs.

35 BS 40 Operating station,
with 2 lines x 40 character fluorescent display and keyboard for process display and control.
Connection via serial interface

07 PM 11 R2 Process graphics module,
for process display and control with a monochrome or colour monitor.
Connection via serial interface

35 BS 94 R1 Operating station,
with 12" colour monitor and keyboard for connection to 07 PM 11.
Connection via serial interface

35 BS 93 R3 Operating station,
with 12" colour monitor, keyboard and graphics module for process display and control.
Connection via serial interface

935 BK 70 Picture configurator,
for planning 07 PM 11 and 35 BS 93.

935 PM 71 Software package,
Process display and control for ABB Procontic field bus ZB 10.

35 BS 95 Operating station,
with 12" colour monitor and IBM/AT compatible computer.
Connection via serial interface

935 PM 72 Software package,
Process display and control for
ABB Procontic field bus ZB 10.

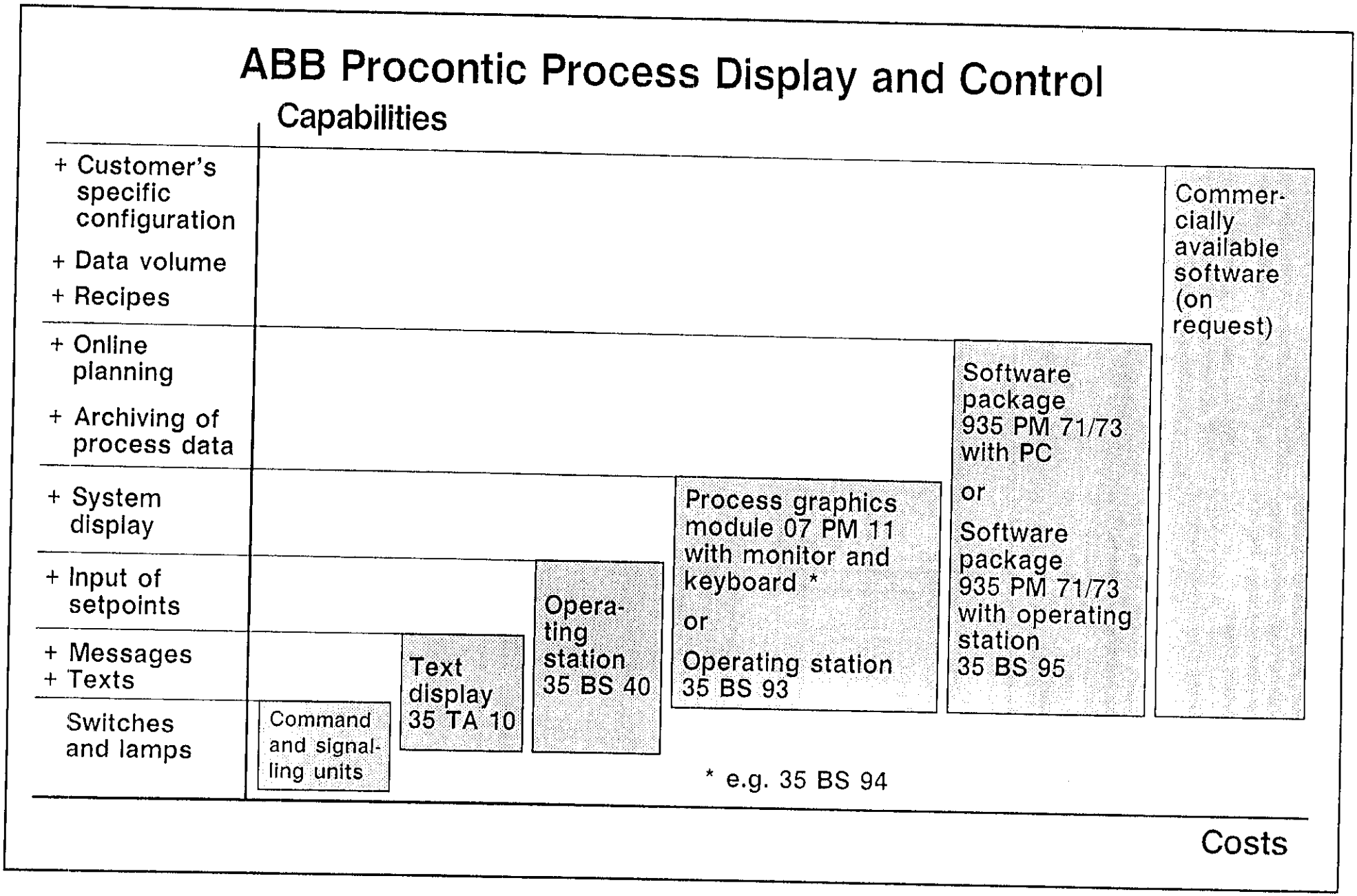


ABB Procontic Process Display and Control
Capabilities
+ Customer's specific configuration
+ Data volume
+ Recipes
+ Online planning
+ Archiving of process data
+ System display
+ Input of setpoints
+ Messages
+ Texts
Switches and lamps
Command and signalling units
Text display 35 TA 10
Operating station 35 BS 40
Process graphics module 07 PM 11 with monitor and keyboard *
or
Operating station 35 BS 93
Software package 935 PM 71/73 with PC
or
Software package 935 PM 71/73 with operating station 35 BS 95
Commercially available software (on request)
* e.g. 35 BS 94
Costs

10.3.1.1 Text display 35 TA 10

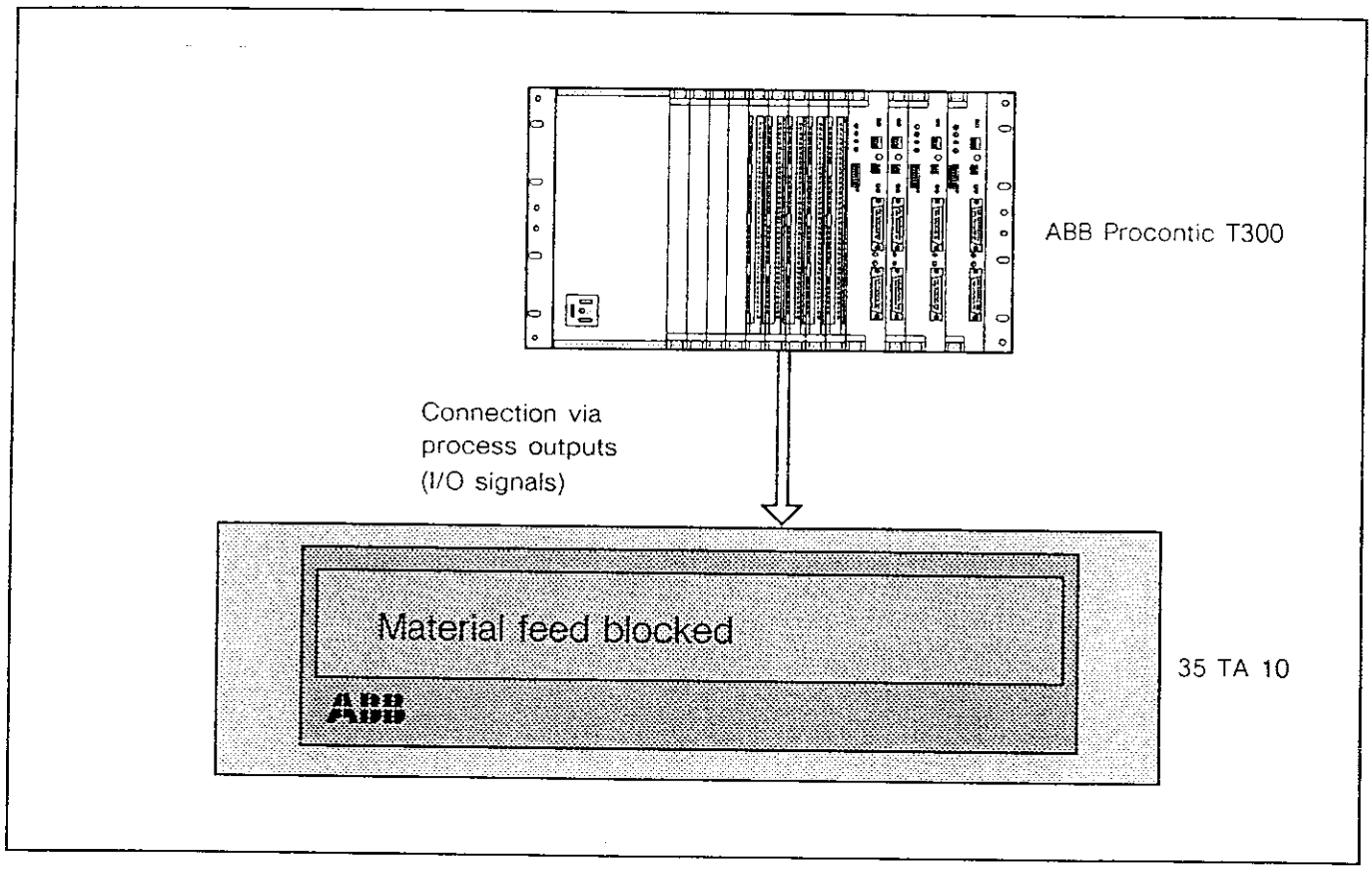
The 35 TA 10 text display shows information about machine states and messages concerning machine operation or maintenance, etc., in plain language. The operator is informed and instructed clearly.

It is connected by way of process outputs belonging to the PLC.

Texts are shown on a fluorescent display, thus ensuring good legibility even in poor lighting conditions. The 35 TA 10 text display is capable of storing up to 400 texts in EPROMs without the risk of destruction in the event of a power failure occurring.

The 35 TA 10 text display can be operated in various modes:

- Direct message: A message is displayed for as long as the message number is present at the inputs.
- First message: The first message is displayed. Up to 19 stored messages are displayed in the sequence of their occurrence.
- Cyclical message: Up to 19 stored messages are displayed in a scrolled form.
- External variable: Texts can be combined with variable values from the ABB Procontic. Variables can occur in any quantity and position in the text.



Technical data

Display	fluorescent display, 1 line x 40 characters
• Character height, representation	7 mm, matrix 5 x 7 dots
• Character sets	Latin and Cyrillic
Text memory	max. 400 texts, 16 kbyte EPROM (2 x EPROM 2764)
Control	via process outputs (binary or BCD coded)
Supply voltage	24 V DC (19...33 V including ripple), ripple < 3 Vp-p
• Current consumption	typ. 250 mA at 24 V
• Power dissipation	approx. 8 W
Type of protection of the front panel	IP65 with seal
Ambient temperature	0...55 °C
Storage temperature	- 25...70 °C
Dimensions of the module (width x height x depth)	254 x 74 x 58 mm
Weight	approx. 800 g

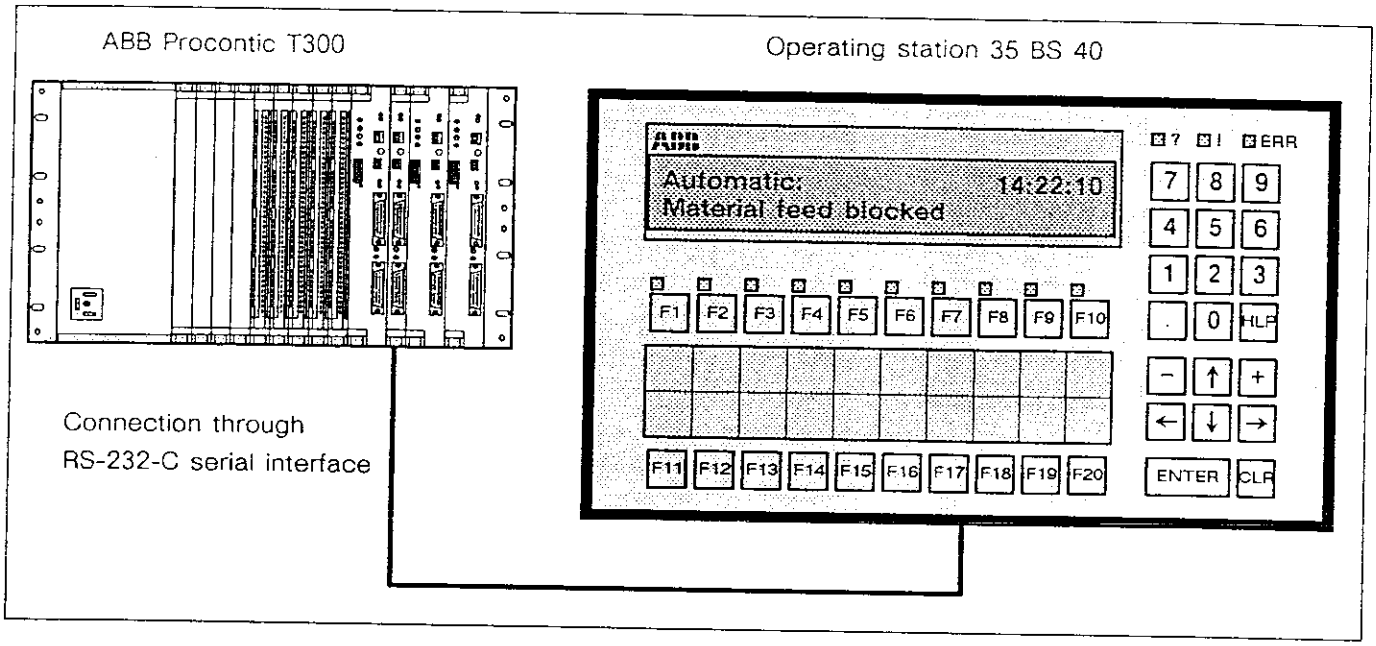
10.3.1.2 Operating station 35 BS 40

The 35 BS 40 operating station shows information about machine states and messages concerning machine operation or maintenance, etc., in plain language. Texts are shown on a fluorescent display, thus ensuring a good legibility in poor lighting conditions.

The operator can also influence the machine sequence by means of the 35 BS 40 operating station. By means of the keys, for example, it is possible to control machine parts or to modify modes and setpoints.

Planning

The 35 BS 40 operating station is programmed using the 935 BS 40 editor on the 07 PC 31/32 programming unit or on a PC. The 935 BS 40 editor is used to program the texts for the display including variable values (time of day, actual values and setpoints from the ABB Procontic, etc.) and the meanings of the function keys.
The PLC program for control of the operating station of 35 BS 40 is created using the programming software of 907 PC 32.
Ready-made cables are used to connect the station to the ABB Procontic through the serial interface.

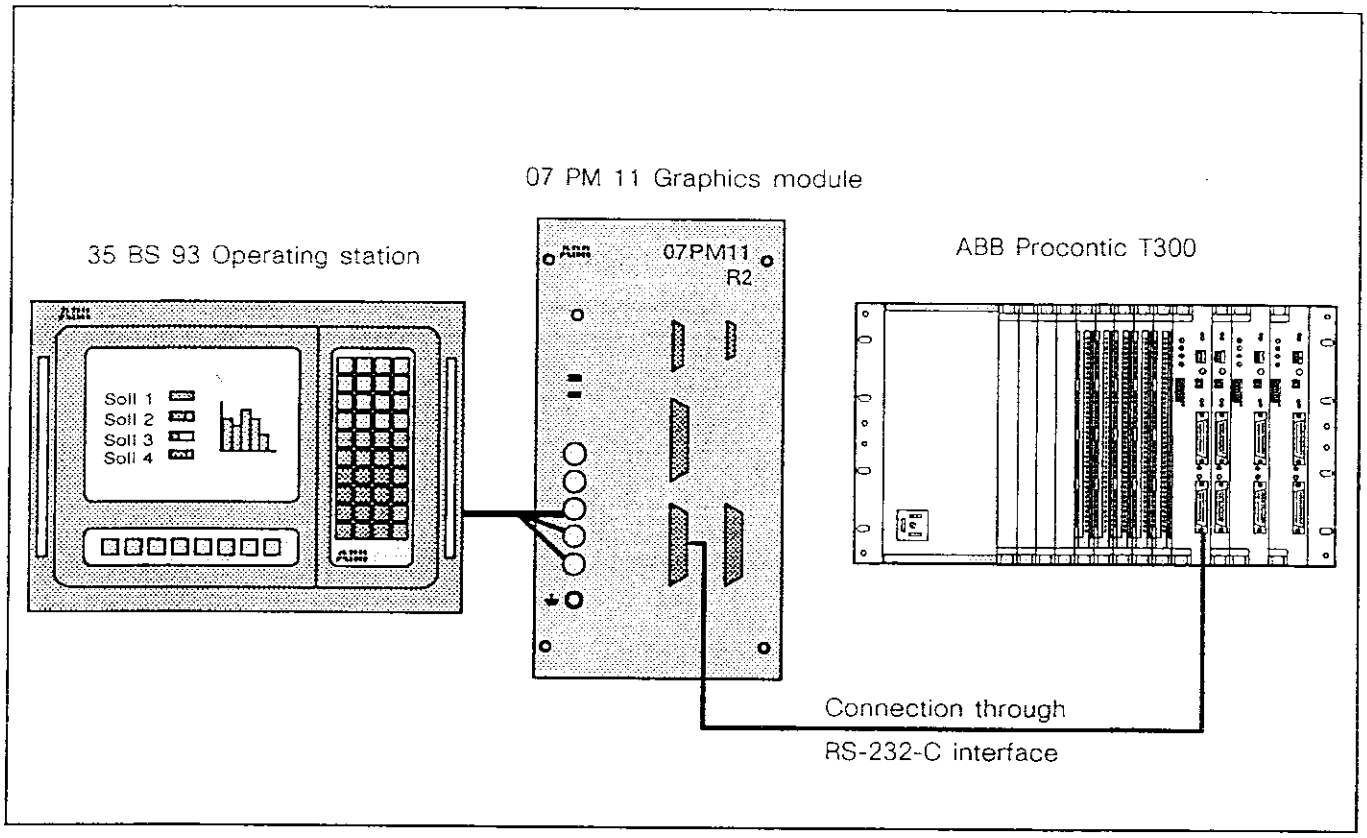


Technical data

Display	fluorescent, 2 lines x 40 characters
• Character height, representation	7 mm, matrix 5 x 7 dots
• Character sets	ASCII, German, Swedish, Spanish
Entries	20 Function keys, 8 control keys, decimal keys, 18 x 18 mm key area, keys with tactile touch, function and control keys configurable by planning (switch-on and switch-off codes with up to 16 ASCII characters), function keys of F01...F10 with LEDs, the display of the LEDs (off, on, brief flashing, long flashing) can be controlled.
Text memory	max. 999 Texts, 32 kbyte battery-buffered RAM, optional program memories of 35 PR 41 and 35 PR 42
Interface	serial interface in accordance with EIA RS-232 or 20 mA (TTY) data format and baud rate (300...19200) are adjustable, data flow control via RTS/CTS or XON/XOFF, communication via ASCII protocol
Supply voltage	24 V DC (19...33 V including ripple), ripple < 3 Vp-p
Power dissipation	8 W
Type of protection of the front panel	IP65
Ambient temperature	0...55 °C (storage temperature: -25...70 °C)
Interference resistance	In accordance with IEC 801-4 and IEC 801-1
Dimensions of the module	width 325 mm, height 190 mm, depth 43.5 mm
Weight	Approx. 1400 g

10.3.1.3 Process graphics module 07 PM 11

Together with a monochrome or colour monitor and a keyboard, the process graphics module of 07 PM 11 enables realization of an operating station, with which processes can be visualized and operated.



The 07 PM 11 process graphics module relieves the ABB Procontic of display-specific tasks, e.g., images, values and colours can be modified without the PLC having to have information about the shape and colour of picture elements.

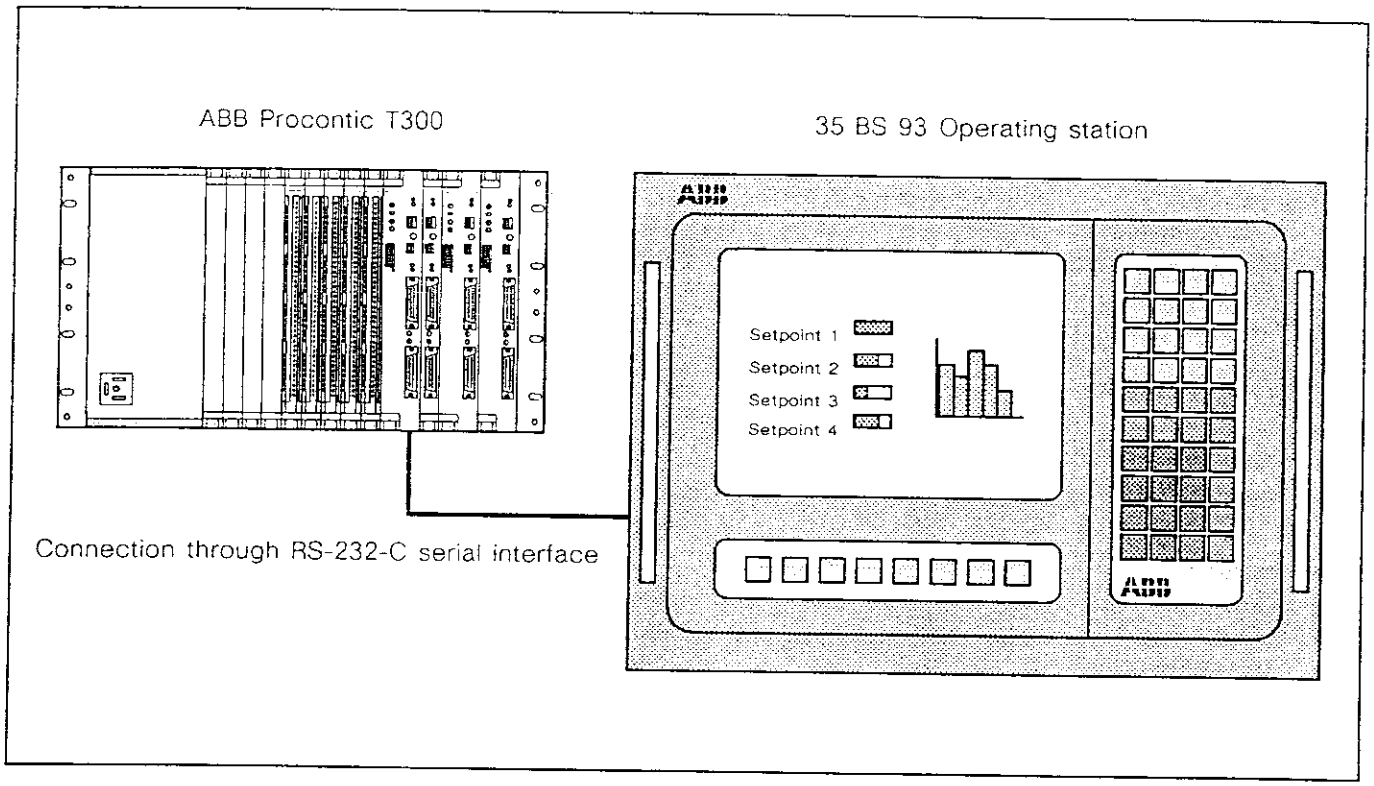
- Active mode:
  The communication is realized actively by the 07 PM 11 graphics module in the active mode. It reads the data for the process display out of the ABB Procontic. Output data are transferred directly to flags of the ABB Procontic.
- Text output:
  Output of texts (logs, malfunction messages, etc.) through a serial interface to a printer or terminal. The texts may contain variable values.
- Data and image variables specific to communication are stored in EPROMs on the 07 PM 11 graphics module. The PLC memory is therefore not burdened by these. The 07 PM 11 process graphics module of is capable of storing data for approx. 20...71 images.
- 100 image variables per image.
- Various image variable types for representation of
  . values (numbers, bars) and/or
  . states (colours, flashing, image parts).
- Simple updating of image variables regardless of the image structure.
  . Passive mode: with logical commands
  . Active mode: by independently reading the values from the ABB Procontic.
- Monitor interface (R, G, B; TTL)
- 3 serial interfaces for the connection of
  . ABB Procontic
  . keyboard (TTL and RS-232-C)
  . printer

Technical data

Power supply	
• Supply voltage	+ 24 V ± 30 %
• Current consumption	< 1.5 A
Power dissipation	36 W
Operating modes	
• Terminal (similar to VT100)	
• Passive mode (PAN mode)	
• Active mode	
Semi-graphics	
• Character format	6 x 6 to 16 x 16 pixels
• Number of characters	2 x 256
• Attributes	256
• Colours	8 colours (background and foreground of the characters can be chosen separately
Type of flashing	1
Display	
• Frequency	50 Hz without interlacing
• Resolution	17...47 lines x 32...85 characters depending on the character format
Serial interfaces	
• Quantity	3
• EIA standard	RS-232-C / RS-423-A
• Transmission speed	150...19200 baud
Monitor control, optionally	
• CCIR standard	R, G(S), B, BAS, SYN
• Standard TTL level	R, G, B, HDR, VDR
Mechanical design	
• Double Eurocard PC board format housing For installation in subracks	6U x 238 mm, 7 modular spacings (28 R) conforming to DIN 41494 or for direct wall mounting

10.3.1.4 Operating station 35 BS 93

Processes can be visualized and operated by means of the 35 BS 93 operating station. The 35 BS 93 operating station is suitable both for installation in 19" racks and also for installation in switch panels. The front panel has the IP 65 type of protection.



The 35 BS 93 operating station contains a 12" colour monitor, a machine keyboard with 40 keys and 8 soft keys. The keyboard is designed as membrane keyboard and the keys have tactile touch.

The 35 BS 93 operating station relieves the ABB Procontic of display-specific tasks, i.e., images, values and colours can be modified without the PLC having to have information about the shape and colour of picture elements.

- Active mode:
  The communication is realized actively by the 35 BS 93 operating station in the active mode. It reads the data for process display out of the ABB Procontic. Output data are transferred directly to flags of the ABB Procontic.
- Data and image variables specific to communication are stored in EPROMs on the 35 BS 93 operating station. The PLC memory is therefore not burdened by these. The 35 BS 93 operating station is capable of storing data for approx. 20...71 images.
- 100 image variables per image.
- Various image variable types for representation of
  . values (numbers, bars) and/or
  . states (colours, flashing, image parts).
- Simple updating of the image variables regardless of the image structure.
  . Passive mode: with logical commands
  . Active mode: by independently reading the values from the ABB Procontic.
- Serial interface for ABB Procontic connection.

Planning tools

The 935 BK 70 image configurator is a complete package containing all planning tools ranging from the image editor to automatic programming of image EPROMs. The 935 BK 70 comprises software which is capable of running on an IBM compatible personal computer. All capabilities of the 35 BS 93 R3 operating station are supported from version **935 BK 70 R402**.

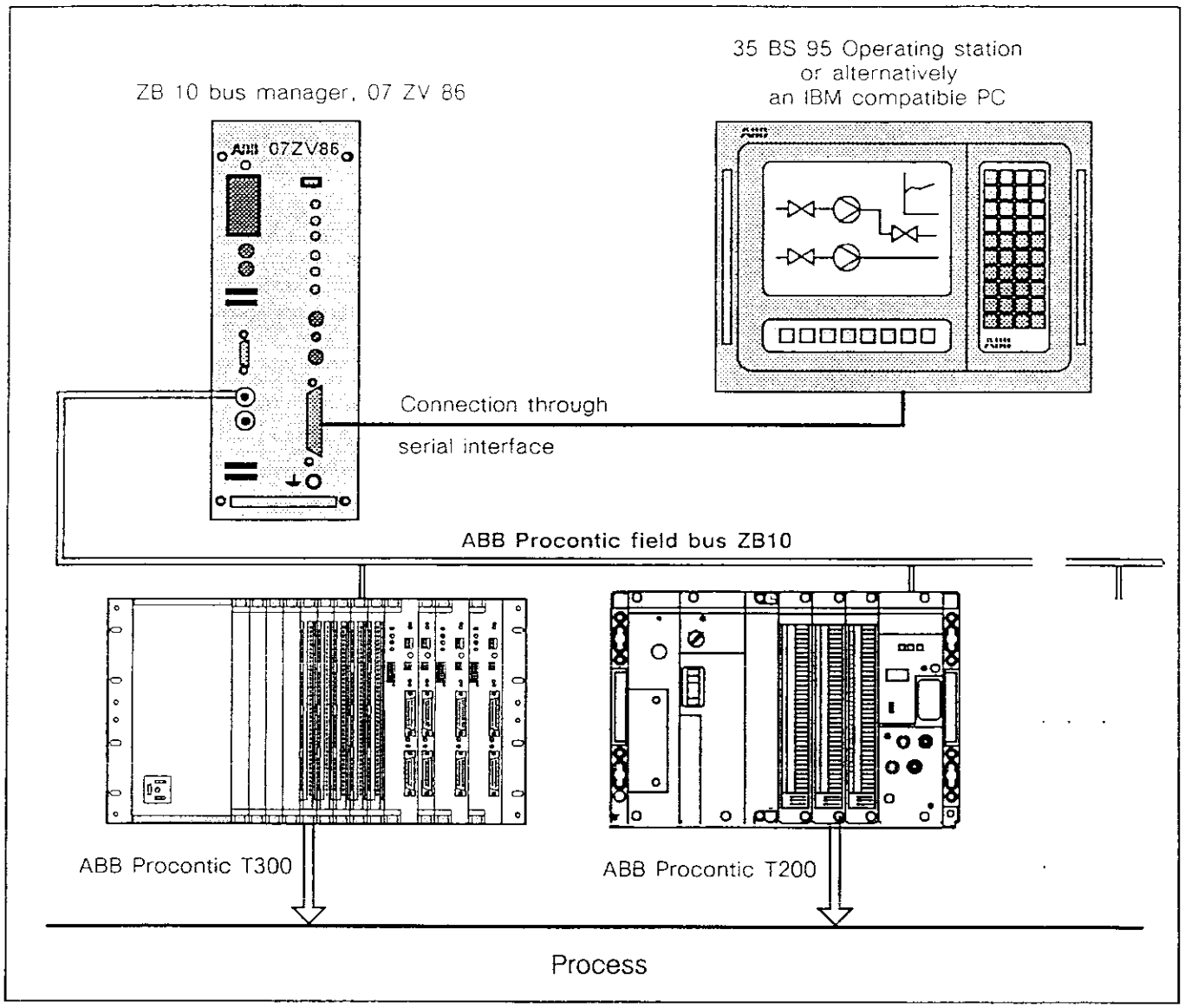
- Editor for allocation list: Information is given here about the individual image variables, about where the data can be found (in which PLC, in which flags) and how the data is to be interpreted.
- Text editor
  The user can select an editor of his choice from the 935 BK 70 image configurator.
- Test functions: The created project can be tested using these functions without the need for a PLC to be connected.

Technical data

Operating modes	
• Terminal (similar to VT100)	
• Passive mode (PAN mode)	
• Active mode	
Semi-graphics	
• Character format	6 x 6 to 16 x 16 pixels
• Number of characters	2 x 256
• Attributes	256
• Colours	8 colours (Background and foreground of the characters can be chosen separately)
• Type of flashing	1
Display	
• Frequency	50 Hz non-interlacing
• Resolution	17...47 lines x 32...85 characters depending on the character format
Serial interfaces	
• Number	3
• EIA standard	RS-232-C / RS-423-A
• Transmission speed	150...19200 baud
Key design	Mechanical short-stroke keys underneath a membrane Labelling of keys F1...F16 can be varied
Protective monitor cover	Plexiglass GS
Regulations	
• Hardware design	VDE 0110, VDE 0160
• Type of protection of the front panel	IP 65
• Type of protection in total	IP 10
• Interference suppression	VDE 0871, limit category B
• Interference resistance	IEC 801
Supply voltage	220 V AC -15 % + 10 %
Power dissipation	120 W
Ambient temperature	0...40 °C
Storage temperature	-20...70 °C
External ventilation	Conditionally necessary
Dimensions of the module	19" x 8 height modules x 405 mm
Weight	20 kg

10.3.1.5 Software package 935 PM 71

The software for the process display and operation, 935 PM 71, enables the realization of an operating station on an IBM compatible PC, with which processes can be visualized and operated. The ABB Procontic PLCs are coupled via the ABB Procontic field bus ZB 10. The PLCs are connected to the operating station through the 07 ZV 86 bus manager.



The ABB Procontic PLCs are relieved of display-specific tasks, i.e., images, values and colours can be modified without the PLC having to be informed about colour and shape of picture elements.

The 935 PM 71 is a menu-assisted system. The required functions are selected by function keys or the cursor and the user does not need to learn any commands by heart! Every function is explained by additional help texts.

Capabilities:

- Process monitoring with:
  . fully graphic flow charts
  . tables
  . trends
- Process operation by setting analog and binary outputs via menu selections and from flow charts.
- Alarm detection and processing
  . Print-outs
  . Display of unacknowledged and unnoticed alarms
  . Plain language messages and request for inputs
- Simple updating of process variables regardless of the image structure by independent reading of values out of the process image of the 07 ZV 86 bus manager.
- Archiving and logging. 16 different archive and log formats can be defined.
- Event control: A specific event is followed by one or several actions.

- o Processing of process values via virtual variables. Mathematical, logical and statistical functions are available for selection.
- o ONLINE planning
- o Symbol library: The scope of delivery of the 935 PM 71 includes a symbol library. The symbols are based on DIN 30 600 and are arranged according to subject groups. This substantially facilitates the creation of flow charts, e.g., of process engineering installations.
- o Planning example: An included planning example explains the procedure to be followed when planning a process display and control system with the 935 PM 71 and the field bus ZB 10.

Operating station:

The 935 PM 71 can be run on commercially available personal computers of the IBM PC/XT/AT type. The following hardware is necessary:

- 640 kbyte RAM
- Hard disc with a capacity of at least 20 Mbytes
- MS DOS as from version 3.3
- Math. co-processor
- EGA graphics with a 256 kbyte RAM
- Hardware clock
- RS-232-C serial interface (COM1) for the 07 ZV 86
- Parallel interface for a printer
- 5.25"/360 kbyte or 3.5"/720 kbyte floppy disc drive for the installation of the 935 PM 71

ABB offers the IBM/AT compatible industrial computer of **35 BS 95** as an operating station.

Processing stations:

The following systems from the ABB Procontic family are available as processing stations:

- o ABB Procontic b
- o ABB Procontic T200
- o ABB Procontic T300

These systems are networked through the field bus ZB10. Here, 250 telegrams with 32 bits each are provided.

Communication:

The communication between the operating station with the 935 PM 71 and the processing stations is handled through the 07 ZV 86 bus manager. The bus manager ensures the proper sequence of data communications. Here, 250 telegrams with 32 bits each are provided.

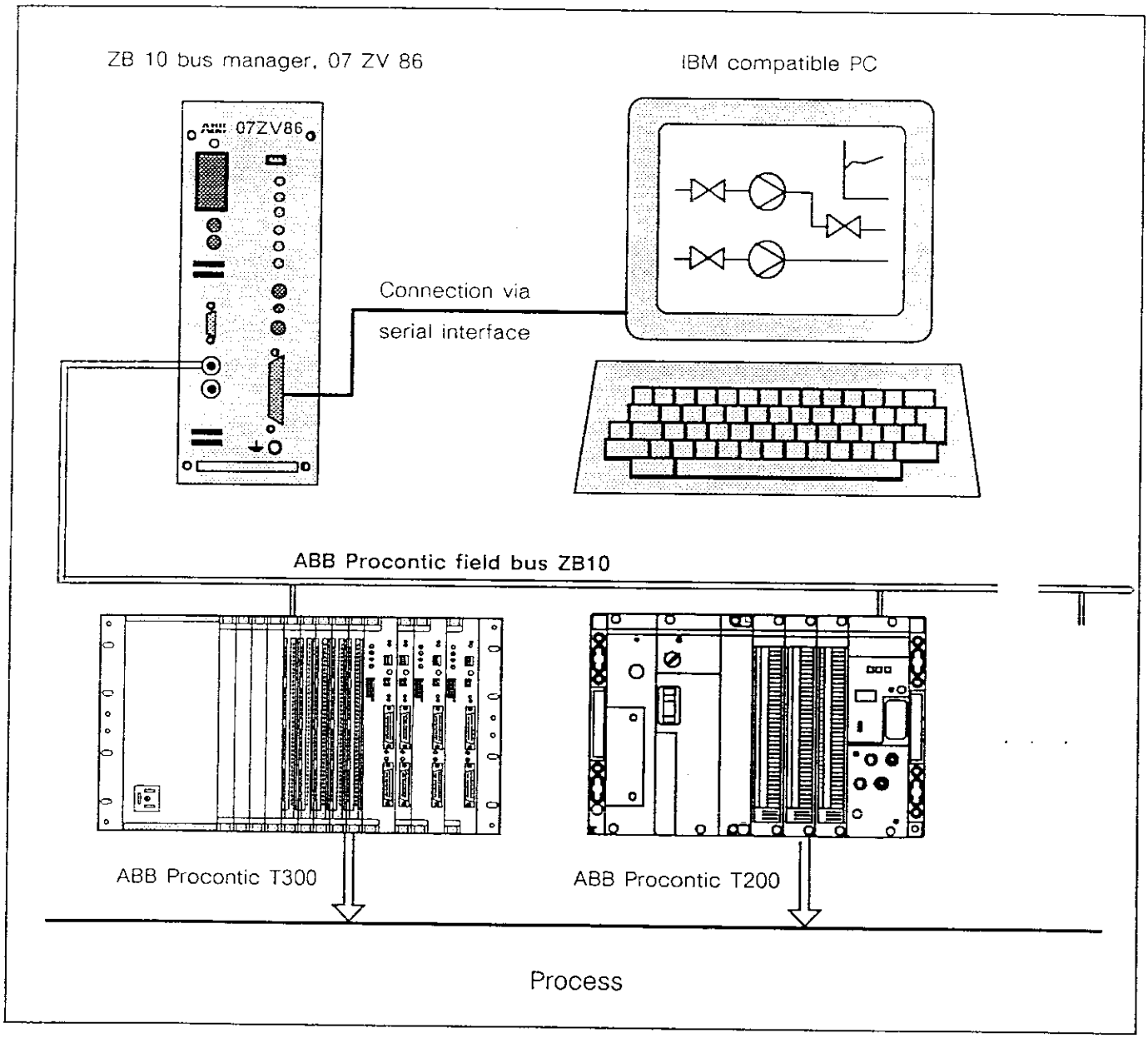
Software:

935 PM 71 contains an integrated graphics editor for image configuration. The process variable definition is menu-assisted. Actions are simply selected with function keys for "programming" the event control.

The ABB Procontic programming and test system, 907 PC 32, is available for programming the ABB Procontic system family. The field bus ZB 10 is programmed with the 930 PC 30 programming and diagnostic software.

10.3.1.6 Software package 935 PM 72

Together with a personal computer, the software for the process display and control, 935 PM 72, enables the realization of an operating station, with which processes can be visualized and operated. It is the linking element for the man–machine communication (MMC) between the operator and the ABB Procontic field bus ZB10. The ABB Procontic programmable controllers (PLCs) also communicate through the ABB Procontic field bus ZB 10.



Capabilities:

- o Semi–graphic display representation
- o Screen selection by means of selection markers or entering the screen name
- o Mouse control possible
- o Re–adjustment, disabling, enabling, marking process variables
- o Acknowledgement of malfunctions
- o Output of commands
- o Output of setpoints
- o Printer operation
- o Hard copy (also in colour)
- o Operation assisted by menus
- o Pre–allocation of function keys
- o Practically unlimited number of images
- o Screen activation time: less than 1 second
- o Up to 1000 process variables in each image

Options:

- Online image editor
- Online trend
- Log generator
- Alarm overview
- Virtual keyboard
- Long-term and balance archives

Communication:

The communication between the operating station with the 935 PM 72 and the processing stations is handled through the 07 ZV 86 bus manager of the ABB Procontic field bus ZB 10. The bus manager ensures the proper sequence of data communication. Access to the data of the field bus is possible through the operation station with the 935 PM 72.

10.3.2 Industrial computer 35 IR 93

The 35 IR 93 is an industrial computer integrated in the ABB Procontic T300 which can be programmed in a high-level language. It has unrestricted access to all automation components in the system, e.g., to the PLC, CNC, communication processors, binary I/O, analog I/O, image processing, ...

The communication between the PLC and the industrial computer takes place through the MPST bus, two communication channels being possible:

1. Communication via I/O signals

Any chosen I/O signal and every global value can be pointed easily from the PLC to the industrial computer.

2. Communication via PLC operating commands

Each operating command of the PLC can be sent both via the serial interface and via the MPST bus to the PLC. The industrial computer is thus able to obtain all PLC statuses, variable statuses and error statuses capable of interrogation (see also chapter entitled "Operation of the PLC via the MPST bus").

Handling the industrial computer is documented in a separate user manual.

Capabilities

Hardware:	Industrially suitable standard ABB Procontic T300 components, mass storage without mobile parts (RAM/EPROM discs), winchester and floppy discs optional via SCSI
Access:	The 35 IR 93 can be programmed in: - PASCAL - C - FORTRAN - PL/M - Assembler
Software:	iRMX86 real-time operating system from INTEL; software packages such as editors, compilers, linkers, debuggers are available

Programming and testing are performed on the same hardware as used later.

Several industrial computers can be installed in one subrack, thus allowing later performance increases and further expansion.

Typical areas of application

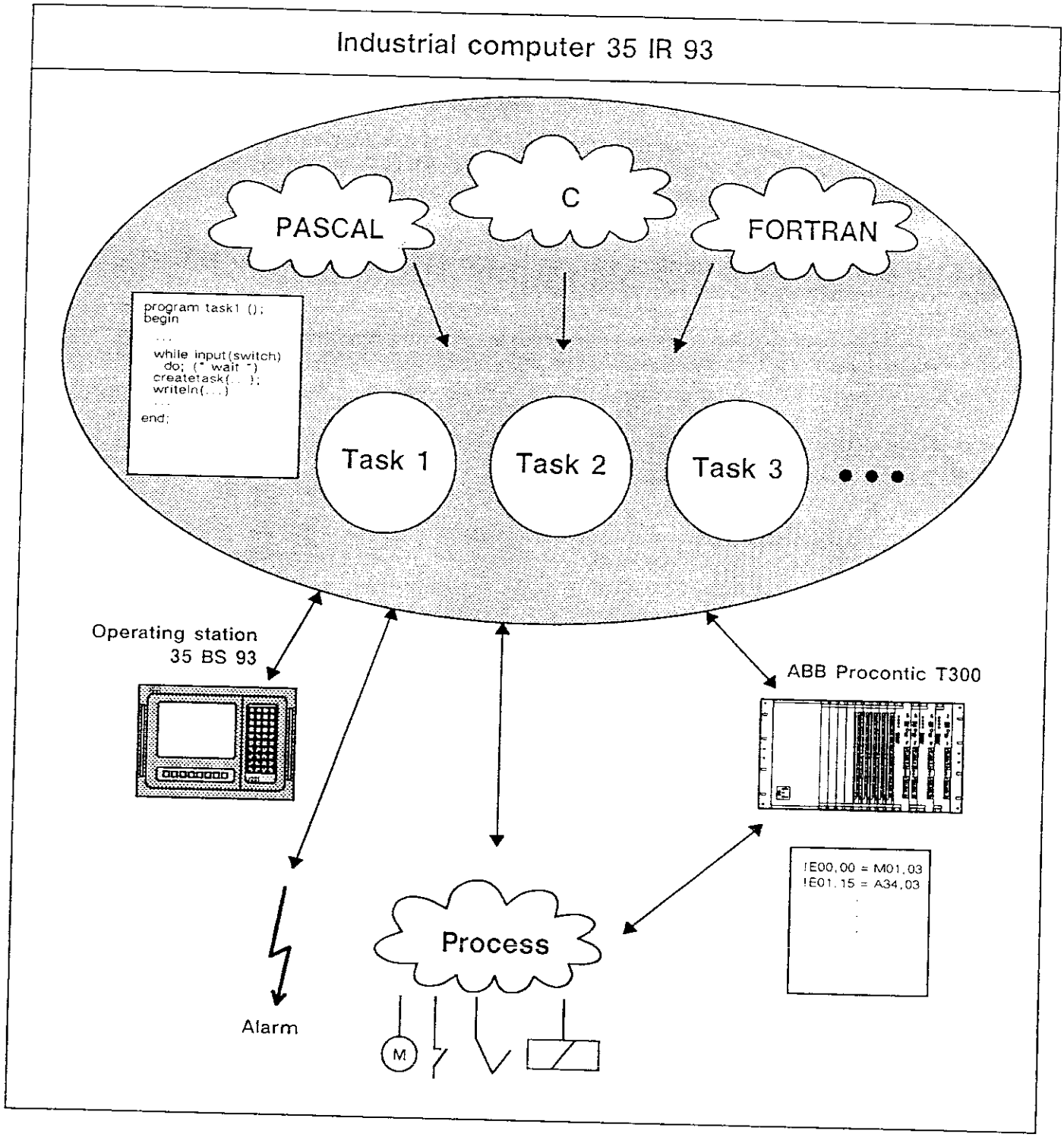
- Communication
- Process control
- Logging of events
- Arithmetic functions
- Sorting (e.g., of data)
- Statistics (e.g., of measured values)
- Optimization tasks
- Diagnosis
- Error handling

Scope of delivery

Besides the complete 35 IR 93 module, the scope of delivery also includes a manual and discs containing program development libraries.

Technical data

Discs:	EPROM disc, RAM discs, winchester and floppy discs are optional
Clock:	Battery-backed hardware clock
Interfaces:	2 or 4 serial interfaces (RS 232 C, V24)
Operating system:	iRMX86 (INTEL)
Tasks:	Large number of user tasks


Industrial computer 35 IR 93
PASCAL
C
FORTRAN
program task1 ();
begin
...
while input(switch)
do; (* wait *)
createtask(...);
writeln(...)
...
end;
Task 1
Task 2
Task 3
Operating station
35 BS 93
ABB Procontic T300
!E00,00 = M01,03
!E01,15 = A34,03
Process
Alarm
M

10.3.3 Coupling: ABB Procontic T300 to the ABB Master process control system. Communication processor 35 KP 91

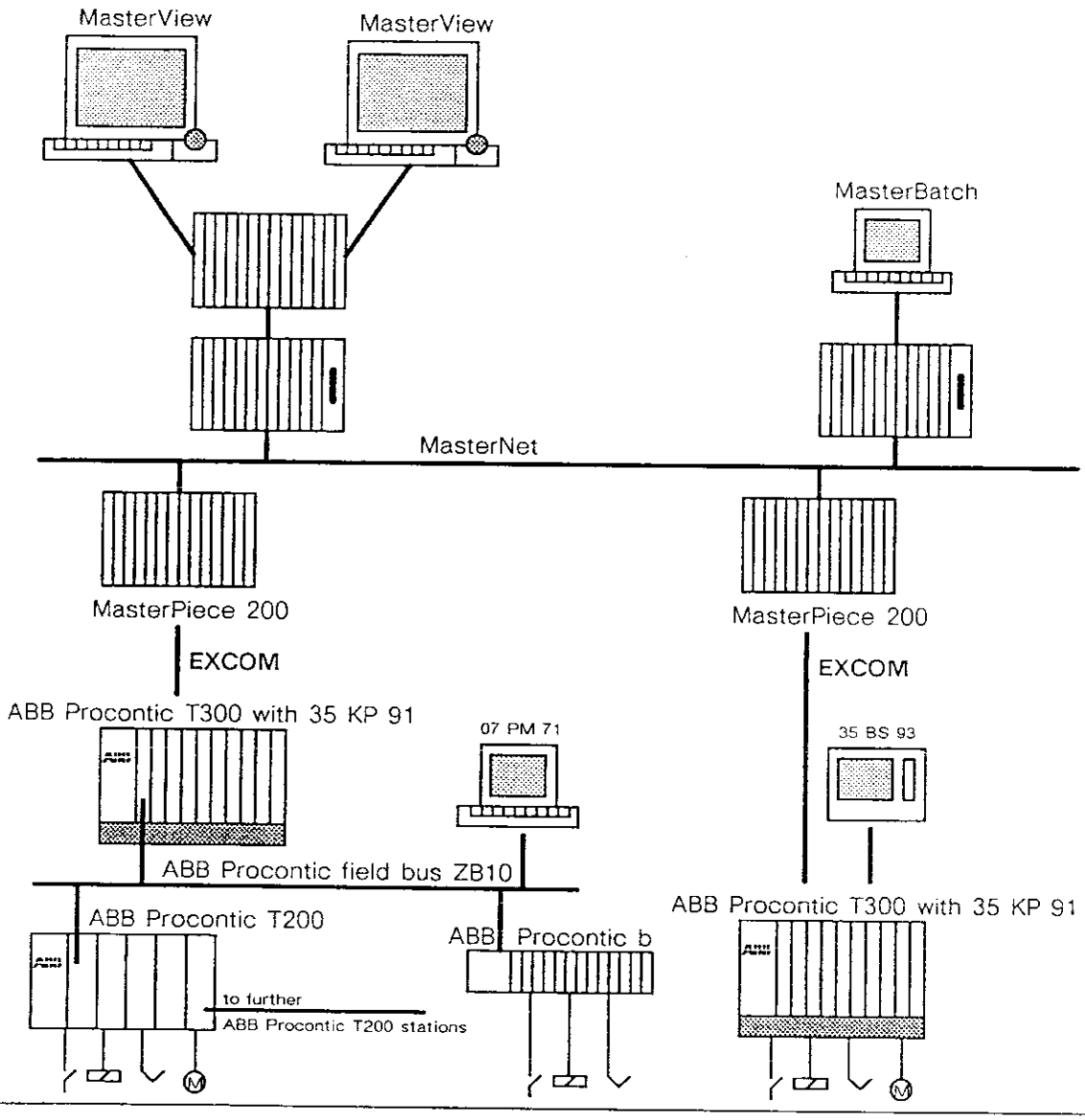
The ABB Procontic T300 PLC system is coupled to the ABB Master control system by way of the *35 KP 91* communication processor and the serial asynchronous interface of the *MasterPiece 200* using the *Asea Master EXCOM* transmission protocol.

Required ABB Procontic components

- 35 KP 91 EXCOM communication processor with 935 KP 91 EXCOM communiction processor software.

Coupling the Master process control system to the ABB Procontic PLC system with the communication processor of 35 KP 91

System structure



Data exchange

The data in the *MasterPiece 200* are organized in the data base as files. The communication realized allows the access to the *DAT* file (data exchange file; see the figure below).

Technical data

Protocol	Protocol conforming to the EXCOM description (ASEA Master EXCOM) with check sum
Baud rate	110, 300, 600, 1200, 2400, 4800, 9600
Commands	Translate symbol Read file Write file
Data types	32 booleans 16 bit integers 32 bit integers
Number of inputs	0..7904 Binary 0..383 16 bit integers 0..63 32 bit integers
Number of outputs	0..8160 Binary 0..383 16 bit integers 0..63 32 bit integers

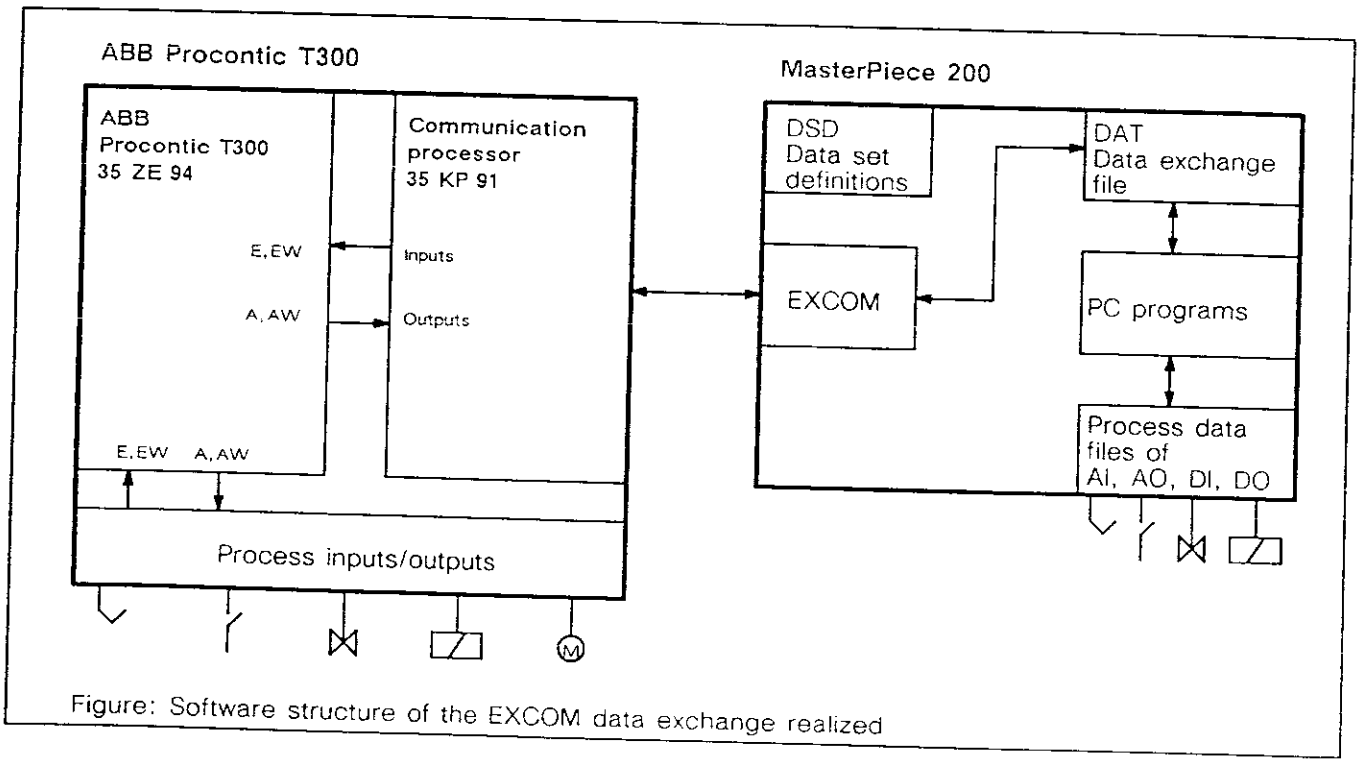


Figure: Software structure of the EXCOM data exchange realized

10.3.4 Coupling: ABB Procontic T300 to VERITRON PAD power converters. Communication processor 35 KP 92

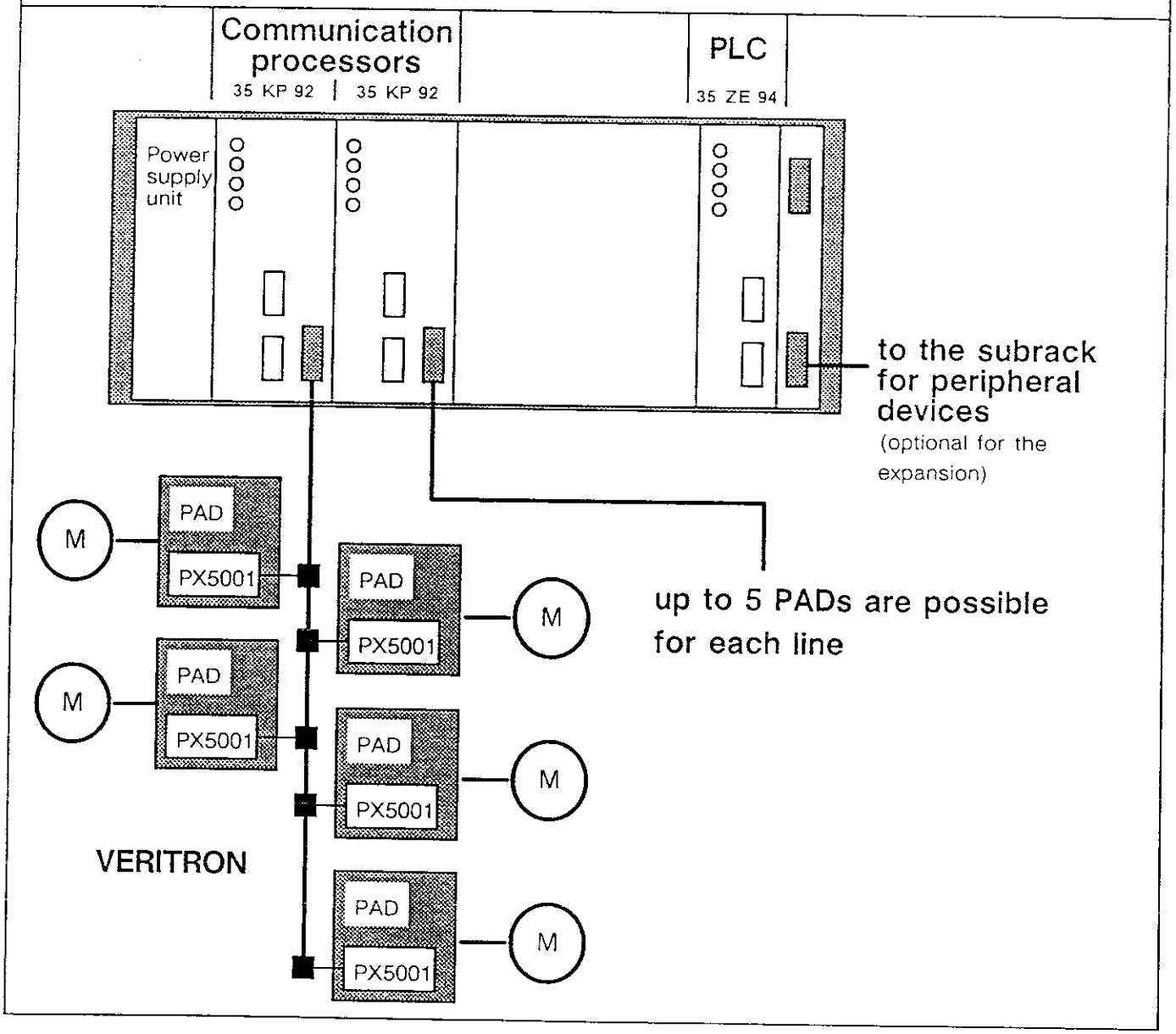
Brief description

The 35 KP 92 communication processor allows access by the ABB Procontic T300 to up to 5 VERITRON PAD power converters through a digital coupling link. Several communication processors can be installed in one subrack.

Scope of delivery

Besides the complete 35 KP 92 hardware, the scope of delivery also includes the manual describing the communication processor and a floppy disc containing a 907 PC 31 PLC example program.

Coupling of VERITRON PAD drive controllers through digital coupling to the ABB Procontic T300 using the communication processor of 35 KP 92



Data exchange

The processor transfers the setpoints and control information for the VERITRON type drive amplifiers to the communication processor 35 KP 92, which continuously provides the current actual values and status information of the drive amplifiers for the processor. Transfer of the actual values and setpoints through the PAD bus is executed cyclically, and without any intervention by the processor, by the communication processor 35 KP 92, which also monitors transfer on the PAD bus (see figure below).

Technical data

Number of PADs:	1...5
No. of fast data words (setpoints):	4
No. of slow data words (setpoints):	0...10
No. of fast data words (actual values):	4
No. of slow data words (actual values):	0...10
PAD bus cycle time:	3 ms per PAD
PAD bus baud rate:	375 kbit/s
PAD bus transmission level:	RS-485
PAD bus transmission protocol:	HDLC/ SDLC
Maximum PAD bus cable length:	200 m

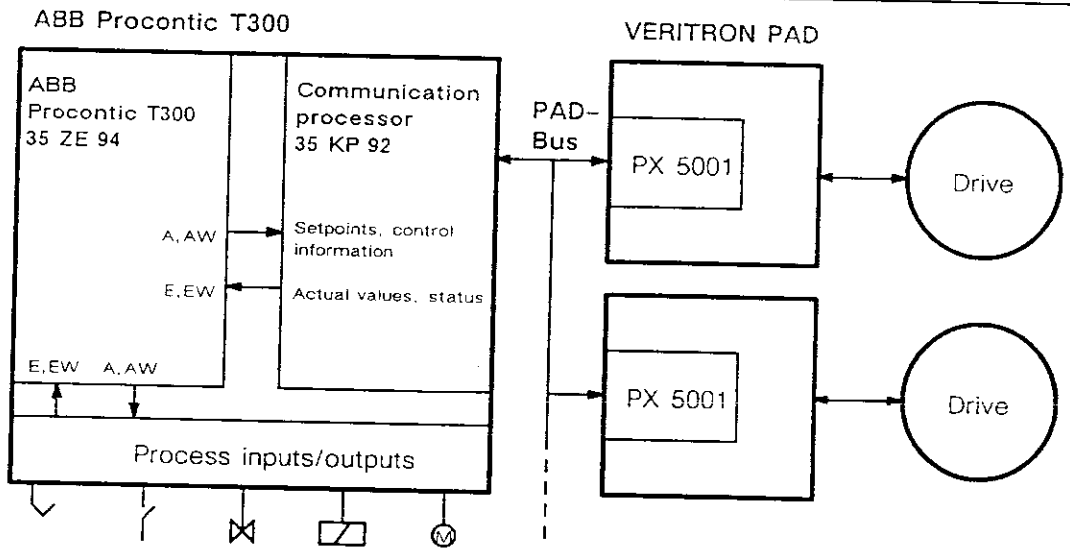


Figure: Data exchange between ABB Procontic T300 and VERITRON PAD

10.3.5 Positioning modules for axes

10.3.5.1 Single-axis positioning module 35 PO 90

The module is a passive station of the MPST bus. It serves for independent positioning of a servo drive within the ABB Procontic T300 system.

The control of the 35 PO 90 user program processing is executed via the ABB Procontic T300 bus. The PLC modules

- POKANF
- POKO and
- POKEND

are available for handling binary I/O traffic (see also module catalogue).

As a quick response to the signals

- Enable
- MNI
- End+
- End-

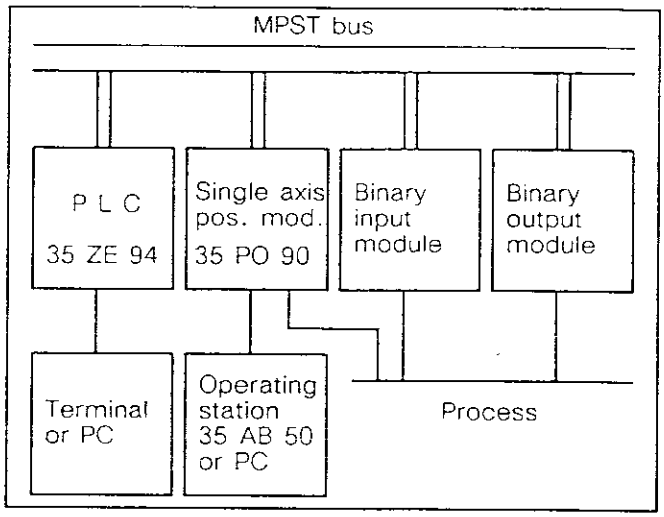
the single-axis positioning module disposes of binary inputs. At these binary inputs the initiators for the signals are connected.

Handling of the 35 PO 90 single-axis positioning module is documented in a separate description.

The controller 35 PO 90, which is controlled by a micro-controller, processes user programs consisting of NC sentences, which it stores and manages itself. The 35 AB 50 operating module serves the purpose of programming and display.

User program archiving on an IBM PC-compatible Compaq PC, and user program input and transfer from the PC to the 35 PO 90 are supported by the PC software 935 AM 50. Up to 256 35 PO 90 modules can operate simultaneously in the same ABB Procontic T300 system.

Example of system configuration using the 35 PO 90:



10.3.5.2 Positioning geometry processor 35 GV 93

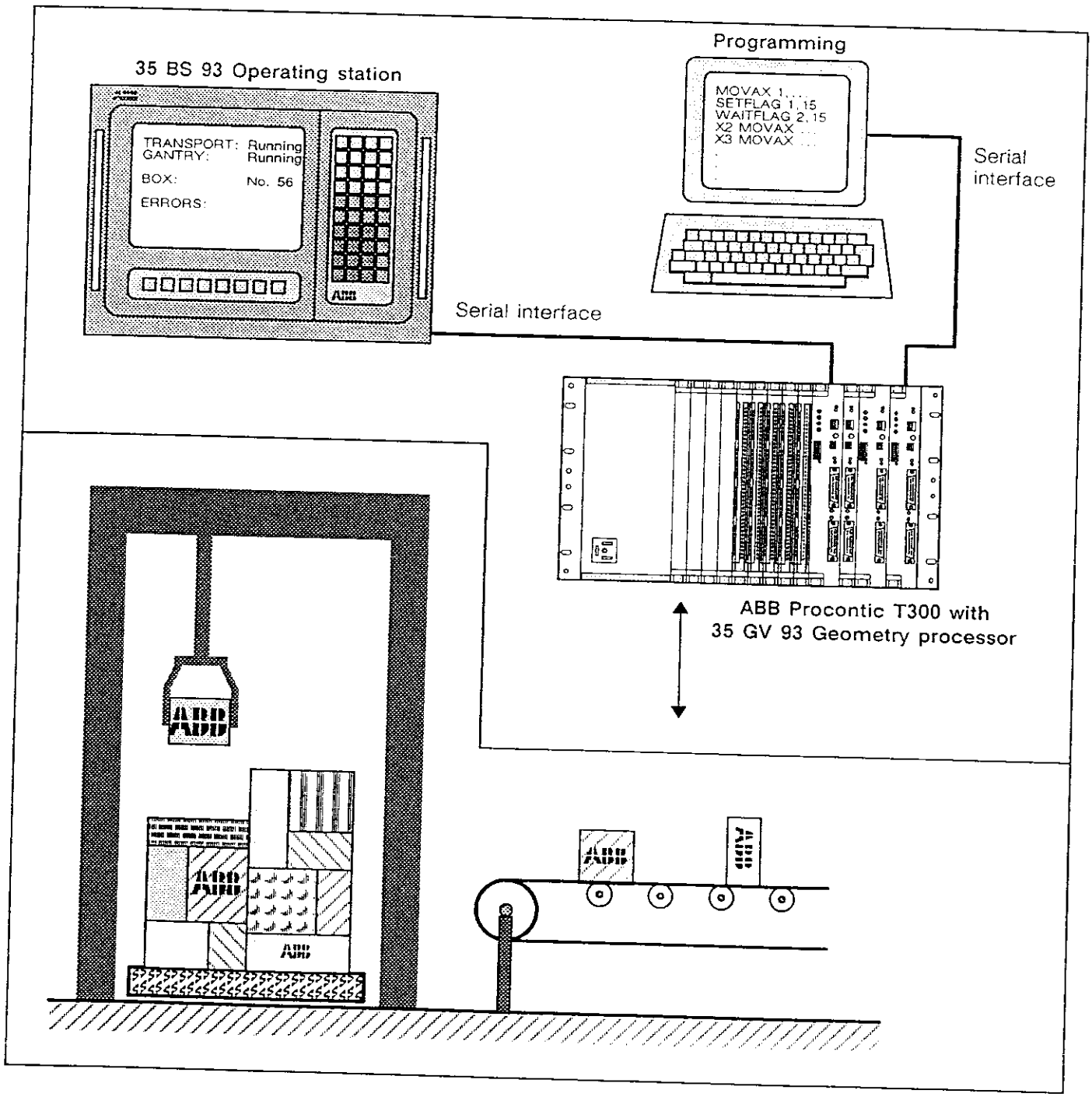
The 35 GV 93 is a positioning geometry processor for 1...12 axes. All axes can be moved independently of each other. Movements can be specified from the PLC by specifying a position or by starting a program.

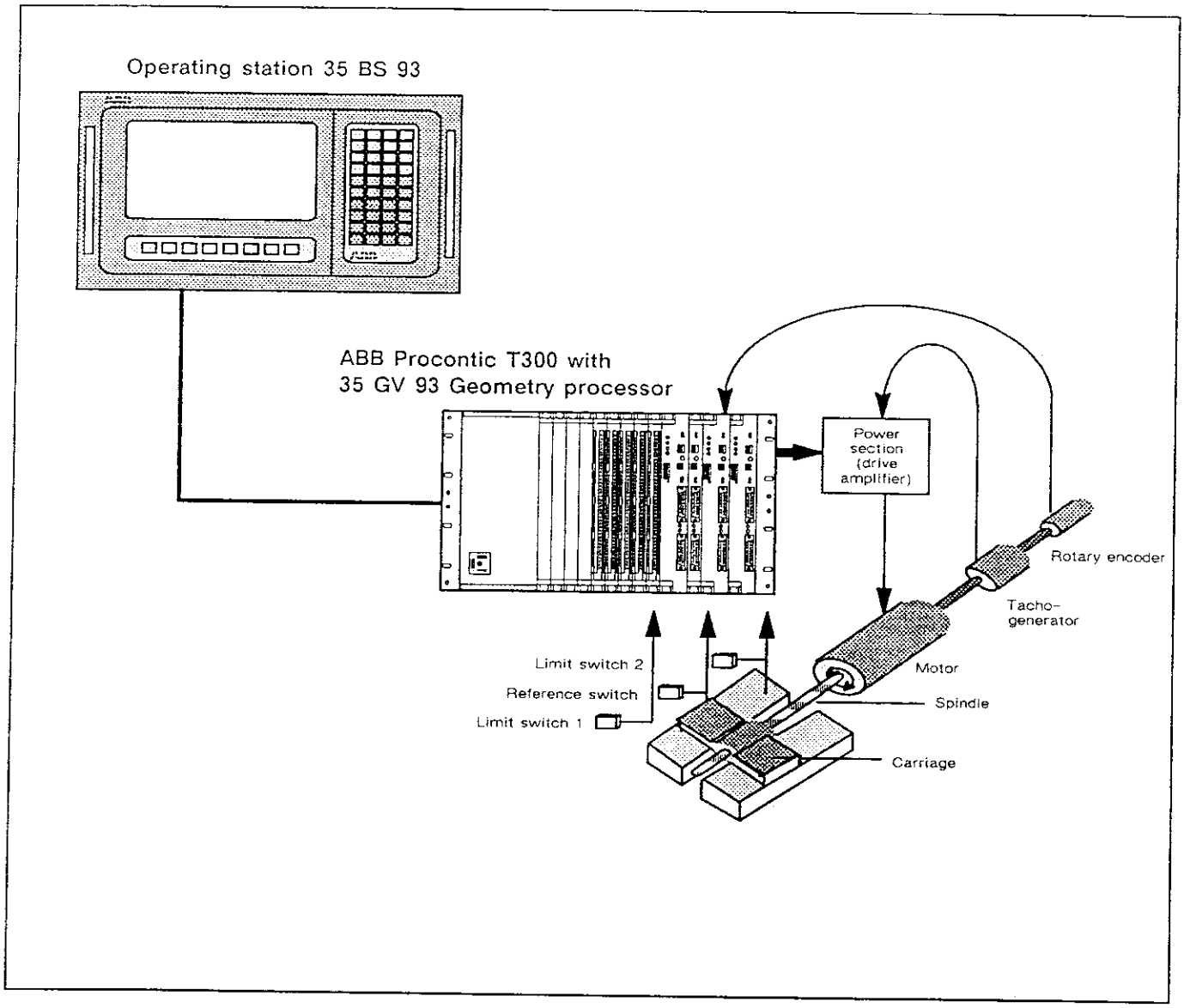
Capabilities

- ☞ Adjustable ramps per axis in both directions of movement (linear, $\sin^2$)
- ☞ Programmable using the affiliated programming, test and commissioning software which can be run on an IBM PC
- ☞ Measured point movement
- ☞ Electronic gearing
- ☞ Industrially compatible standard ABB Procontic T300 components
- ☞ 1...12 axes
- ☞ High speed of movement (> 64 m/min)

Typical areas of application

All areas, in which axes have to be controlled by a PLC, e.g., palletizers, loading gantries, handling robots.




Operating station 35 BS 93
ABB Procontic T300 with
35 GV 93 Geometry processor
Power section (drive amplifier)
Rotary encoder
Tacho-generator
Motor
Spindle
Carriage
Limit switch 2
Reference switch
Limit switch 1

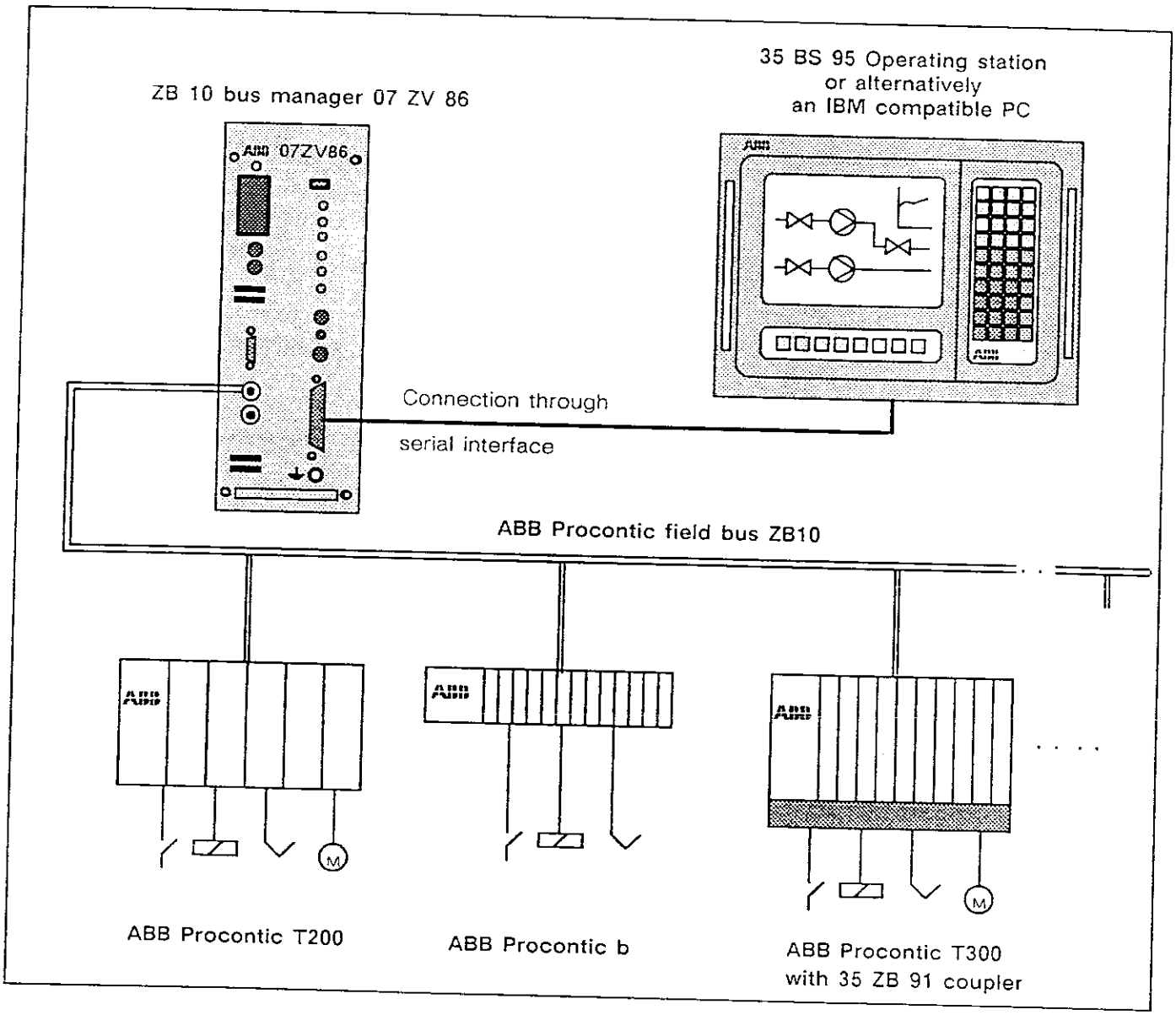
10.3.6 Coupling: ABB Procontic T300 to field bus ZB 10. Coupler 35 ZB 91

The ABB Procontic field bus ZB 10 serves to link up ABB Procontic systems in a network. It is high-speed serial bus (150 kbytes), whose data flow is controlled by the 07 ZV 86 bus manager.

The ZB 10 coupler, 35 ZB 91, serves to connect an ABB Procontic T300 system to the ABB Procontic field bus ZB 10. The coupler is a passive station in the ABB Procontic T300 system.

From the point of view of the PLC, the ZB 10 coupler must be addressed like an I/O module. It can therefore be used instead of an I/O module on the MPST bus. However, the coupler can also be operated outside the address range intended for the I/O modules. In this case, the PLC I/O signals allocated to the coupler must be repointed to the coupler by means of the PLC command of DBV <CR>.

Handling of the ZB10 coupler is documented in a separate description.



10.3.7 Video sensor system, OMS-F, modules 35 IV 90 and 35 KI 90

Together with

- the iconic image processor (35 IV 90)
- the camera interface (35 KI 90)
- the video camera and the monitor

the PLC represents an inexpensive video sensor system in which the image processing is executed in real-time mode by hardware.

The video sensor system serves to evaluate black/white images and it can be used, e.g. for quality control, sorting etc.

The images received via the video camera are sent to the iconic image processor (35 IV 90) via the camera interface (35 KI 90) for evaluation.

Handling is integrated completely in the PLC. That is to say, all operating and programming commands are in accordance with syntax and semantic of the PLC language and are entered in the PLC.

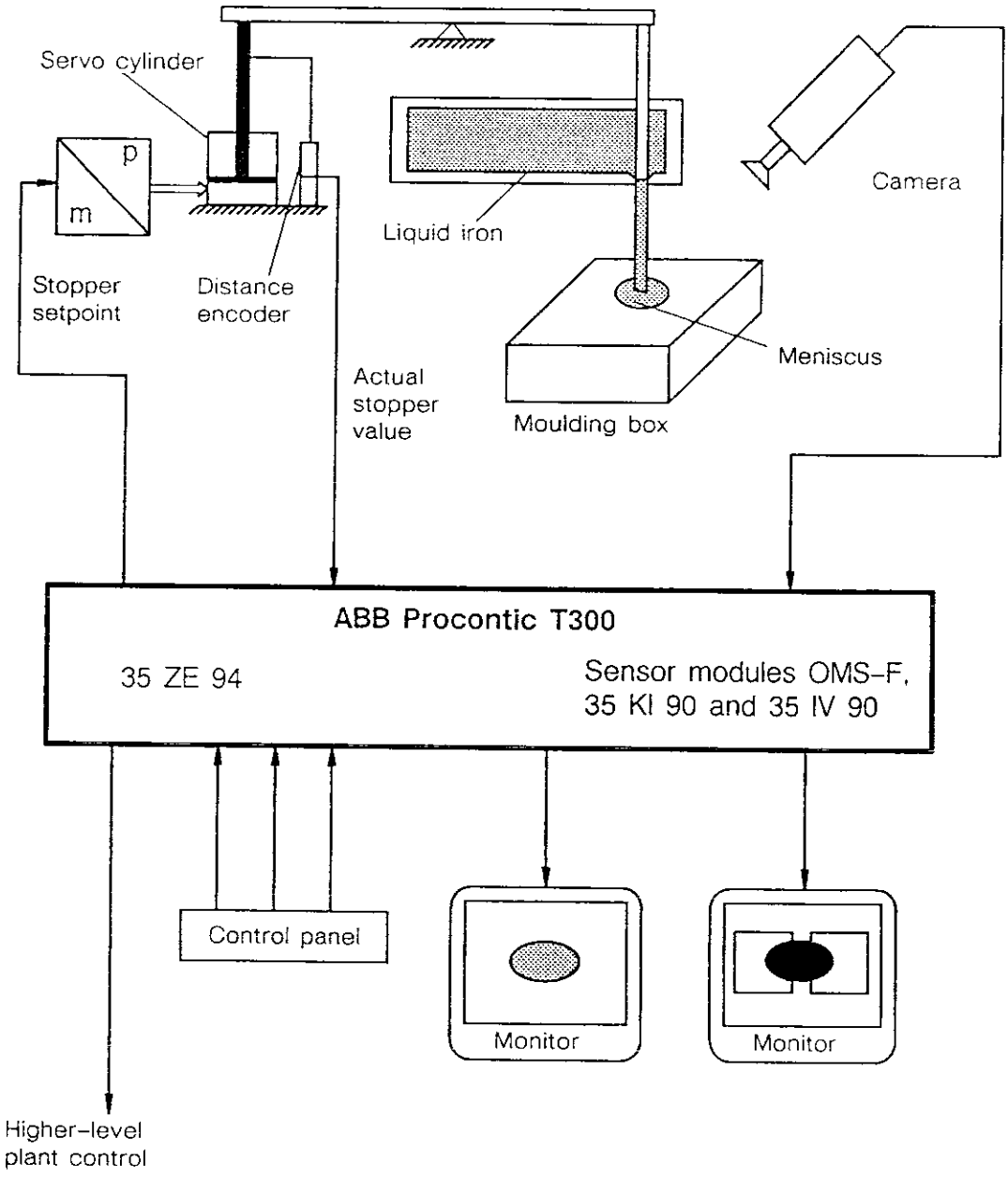


Figure: **Video sensor of OMS-F - Application example of meniscus control**

The module of 35 KI 90 has the following abilities:

- A/D conversion of the video signal with an optional resolution of 128 or 64 grey tone levels. This conversion is designed adaptively. The user can therefore optimally adapt the effective conversion range to the conditions of the scene to be evaluated, i.e., the user is independent of changes in the lighting environment.

- Conversion of the grey tone image into a binary image capable of inversion.

The module of 35 IV 90 has the following image evaluation abilities:

- 8 freely programmable windows

- Evaluation of the area.
For every shot the area can be evaluated simultaneously for all 8 windows.

- Determination of the area's center of gravity
During each shot the area's center of gravity in one window can be evaluated in addition to the 8 area ranges.
- Histogram analysis
For *each* shot the pixels over 8 relevant grey tone areas are analyzed. The allocation of the individual grey tone values to one range can be chosen completely freely. The histogram analysis is possible in one window per shot.

A PLC can communicate with several 35 IV 90 modules. The communication between the PLC and the iconic image processor takes place by interrupt control through the MPST bus. No interface problems are encountered by the user.

After each shot the PLC triggers an interrupt and, as a result of this, program 2 is run through *once*, whereby the data of the iconic image processor are read and prepared by the PLC. In this case, no TURBO mode is possible for program 2 because the TURBO memory (70000...77FFF) is needed for the SENSOR CONSTANTS.

10.3.7.1 Handling the video sensor

The exact handling of the video sensor is described in a separate manual. Only a summarizing overview of video sensor handling and of its integration into the PLC is given here.

The following is available for handling the video sensor:

- Operator control commands and
- function blocks

Setting the SENSOR operating mode

The operating modes are set with the configuration command: KONFS (see also the chapter entitled "Operating and test functions").

When setting the SENSOR operating mode, marginal conditions apply which, however, are obeyed automatically by the PLC:

- The SENSOR operating mode and the TURBO operating mode (user program 2) are mutually exclusive because the TURBO memory of user program 2 is needed for the sensor constants.

- When configurating the SENSOR operating mode, the EVENT operating mode is set automatically for user program 2. The EVENT operating mode for user program 2 cannot be deleted while the SENSOR operating mode is configured.

- When activating the SENSOR operating mode (transition from NO SENSOR to SENSOR) a RAM test is carried out for the RAM of the sensor constants. The RAM is then initialized with "0".

SRQ vector

The image processing module sends an SRQ vector to the PLC. The numerical value of the vector is identical to the station address (0 ... 7) of the iconic image processor (35 IV 90) which has provided its image evaluation data for the PLC. The numerical value of the vector is provided for the user program for further evaluation in the form of the word global value of EW 17,15.

Event operation mode

The PLC user programs are normally executed in the time-controlled mode, i.e., regularly at equal intervals.

The video sensor, however, requires event-orientated execution of the PLC program.

Whenever a shot is completed, user program 2 must be run through *once* for processing of the image. The event is transferred from the video sensor to the PLC, which supplies the bus clock, with the help of the SRQ communication. Therefore the PLC for the evaluation of the video sensor data must always supply the bus clock.

When the sensor ability is configured, program 2 is run through *once* after it has been started *without* an event being necessary for this purpose. This single program cycle is required to start image processing for the first time.

Loading/archiving the sensor constants

With the help of the ABB Procontic programming system 907 PC 32, the sensor constants can be achieved in the mass storage unit of a PC.

It is only possible to read and load the sensor constants when the SENSOR operating mode is set. Otherwise the handling of the sensor constants is the same as for the indirect word constants.

Normal indirect word constants are distinguished from the sensor constants by means of the numbering range:

KW 00,00...KW 39,15:	Indirect PLC word constants
KW 1000,00...KW 2041,15:	Sensor constants

If the RAM for the sensor constants is to be battery-backed, the jumper panel of X 22 must be wired as follows:

- Jumpers 1–2 and 5–6

Function blocks for SENSOR mode

The following sensor-specific function blocks are available in the PLC:

- SETMOD Setting the image processing modes
- CLWIN Deleting all window frames on the image processor
- PROWI Programming the window frames
- PROOFF Programing the offset values for the area counters
- READC Reading the result values of the area counters or histogram result
- COGRA Calculating the area's center of gravity
- PROHIS Programming the histogram stages
- SETBIN Setting/editing the binary image threshold
- MODWI Modifying and programming window frames
- UMIMA Converting pixels to real units of measurement
- GENWI Generation of a window frame

Operator control commands for SENSOR mode

- XW Setting up a window frame
- XB Setting a binary threshold
- XS Scaling the camera
- XO Programming the character offset string
- XM Setting the sensor modes

DBV configuration

A maximum of 5 iconic image processors (35 IV 90) is assigned to one PLC. Each 35 IV 90 module occupies an address area of 32 (20_H) bytes in the I/O area of the MPST bus. The address area for the 35 IV 90 begins at 2000_H:$E540_H$ and is thus located directly after the address area for the PLC axis boards. The pointers to these 35 IV 90 processors are located in the data block directory (DBV) and can be altered with the DBV <CR> command (see also the chapter entitled "Data block directory").

Pointer of the 1st 35 IV 90 :	2000_H:$E540_H$
Pointer of the 2nd 35 IV 90:	2000_H:$E560_H$
Pointer of the 3rd 35 IV 90:	2000_H:$E580_H$
Pointer of the 4th 35 IV 90:	2000_H:$E5A0_H$
Pointer of the 5th 35 IV 90:	2000_H:$E5C0_H$

The 5 accessible DBV pointers are selected for the 35 IV 90 processors as follows:

DBV IV1 <CR>
DBV IV2 <CR>
DBV IV3 <CR>
DBV IV4 <CR>
DBV IV5 <CR>

Initialization

The initialization of the video system by

- Initialization of the binary video threshold
- Deletion of all window frames
- Setting all counters to OFFSET 0

takes place in the initialization task (PCINIT).

RAM test for the sensor constants memory when configurating the SENSOR operating mode

When activating the SENSOR operating mode (transition from NO SENSOR to SENSOR), a RAM test is carried out for the RAM of the sensor constants. The RAM is then initialized with "0".

If an error is detected during the RAM test, the NO SENSOR operating mode is again set automatically and the following error message appears on the screen:

RAM ERROR Socket no.

The socket no. is the number of the faulty RAM socket.

The error number of FF 0F is entered in the error register (transfer RAM, control block 0, byte 6).

11 Communication between the two PLC programs

Communication between *the two user programs of a PLC* takes place by way of global values. The **global values** are I/O operands which are stored physically in the PLC's global value memory. A global value is written into the global value memory by means of an output operand and is read again by the input operand with the same operand number. Therefore, the PLC program treats the global values in exactly the same way as input and output signals, i.e. they are particularly read into the process image from the global value memory at the start of the program cycle and are written from the process image into the global value memory at the end of the program cycle. Both programs of a PLC may write *to the same* binary output group. This applies both when writing to an output module and also when writing to the global value memory.

Example:

- Write global value into global value memory
  !M00,00 = A 64,01
- Read this global value out of the global value memory
  ! E 64,01 = M 01,03

The following I/O operands are defined as global values:

Write global values

- Binary global values,128 values
  A 64,00...A 71,15
- Word global values, 32 values
  AW 16,00...AW 17,15

Read global values

- Binary global values
  E 64,00 ... E 71,15
- Word global values
  EW 16,00 ... EW 17,15

Marshalling

Physical addresses are assigned to the global values just as with the I/O operands. This assignment is carried out in the data block directory (DBV) (see also chapter 9).

Binary global values:

A 64,00...A 64,15	is marshalled to transfer RAM 1000:10CA$_H$
E 64,00...E 64,15	is marshalled to transfer RAM 1000:10CA$_H$
.	.
.	.
.	.
A 71,00...A 71,15	is marshalled to transfer RAM 1000:10D8$_H$
E 71,00...E 71,15	is marshalled to transfer RAM 1000:10D8$_H$

Word global values:

AW 16,00	is marshalled to transfer RAM 1000:10DA$_H$
EW 16,00	is marshalled to transfer RAM 1000:10DA$_H$
.	.
.	.
.	.
AW 17,15	is marshalled to transfer RAM 1000:1118$_H$
EW 17,15	is marshalled to transfer RAM 1000:1118$_H$

Addressing binary global value groups

Access to *one* binary global value group is allowed for both programs of a PLC. Each of the two PLC programs only specifically alters those bits of the group which occur in the respective program.

12 Multiprocessor operation

Up to eight active stations (processor boards) can be operated on the MPST bus. A different station address from the range 0...F_H must be set for each active station. The user need *not* bother about the bus management (arbitration) because bus accessing operations by the individual processors is controlled by the hardware.

If several controls are operated on the MPST bus, *one* PLC must be configured as the MASTER PLC and the others as SLAVE PLCs. The jumper for the bus clock may be fitted on only *one* PLC. If alarm processing (input module 35 EB 92 or video sensor OMS-F) is realized on one of the PLCs, the one PLC concerned must supply the bus clock.

Note:

Only one PLC may supply the bus clock.

Address 1 must be set at the station address for the MASTER PLC!

Any station addresses may be chosen for the SLAVE PLCs.

Data transfer between several controls on the MPST bus takes place via SUPERGLOBAL VALUES. These superglobal values can be read and written by all controls. The memory for the superglobal values is located physically in the transfer memory of the MASTER PLC.

12.1 Planning of superglobal values

The superglobal values are I/O operands which are stored physically in the MASTER PLC's superglobal value memory. A superglobal value is written into the superglobal value memory by an output operand and is read again by the input operand with the same operand number. Therefore, the PLC program treats the superglobal values in exactly the same way as input and output signals, i.e. they are particularly read into the process image from the superglobal value memory at the start of the program cycle and are written from the process image into the superglobal value memory at the end of the program cycle.

In multiprocessor mode, several PLC processors must not write to the same binary output group for superglobal values. The reason for this is that the write operation is not performed for each individual bit, but for the entire output group. When one single binary value is output in the PLC program, the remaining 15 bits belonging to the output group are also output. When several PLCs write to the same output group, they mutually overwrite each others' output values in the superglobal value memory.

Example:

- Write superglobal value into superglobal value memory by PLC n:

  PLC n:
  !M00,00 = A 72,01

- Read this superglobal value out of the superglobal value memory by PLC m:

  PLC m:
  ! E 72,01 = M 01,03

- **Forbidden:**

It is forbidden for PLC n and PLC m to write the same binary output group:

SPS n: !M00,00 = A00,01
SPS m: !M12,00 = A00,02

The following I/O operands are defined as superglobal values:

Writing superglobal values:

- Binary superglobal values A 72,00...A 96,15
  400 values
- Word superglobal values AW18,00...AW22,15
  80 values

Reading superglobal values:

- Binary superglobal values E 72,00...E 96,15
- Word superglobal values EW18,00...EW22,15

Example

It is intended to transfer the value of the binary flag M 0,0 from PLC m to PLC n.

Planning in PLC m:

! M 0,0 = A 72,00 "Write superglobal value (A 72,00)

Planning in PLC n:

! E 72,00 & M 3,5 = ... "Read superglobal value (E 72,00)

A superglobal value is addressed with the same I/O number during write and read operations.

Marshalling

Just like the I/O operands, the superglobal values are allocated physical addresses. These addresses point to the superglobal value memory and are allocated to the operands in the data block directory (DBV) (see also Chapter 9).

Marshalling MASTER PLC

The superglobal values are located physically in the MASTER PLC.

Binary superglobal values:

A 72,00...A 72,15	is marshalled to transfer RAM $1000{:}0F20_H$
E 72,00...E 72,15	is marshalled to transfer RAM $1000{:}0F20_H$
.	.
.	.
.	.
A 96,00...A 96,15	is marshalled to transfer RAM $1000{:}0F50_H$
E 96,00...E 96,15	is marshalled to transfer RAM $1000{:}0F50_H$

Word superglobal values:

AW 18,00	is marshalled to transfer RAM $1000{:}0F52_H$
EW 18,00	is marshalled to transfer RAM $1000{:}0F52_H$
.	.
.	.
.	.
AW 22,15	is marshalled to transfer RAM $1000{:}0FF0_H$
EW 22,15	is marshalled to transfer RAM $1000{:}0FF0_H$

Marshalling SLAVE PLC

The superglobal values are located physically in the MASTER PLC.

Binary superglobal values:

A 72,00...A 72,15	is marshalled to transfer RAM $3000{:}1F20_H$
E 72,00...E 72,15	is marshalled to transfer RAM $3000{:}1F20_H$
.	.
.	.
.	.
A 96,00... A 96,15	is marshalled to transfer RAM $3000{:}1F50_H$
E 96,00... E 96,15	is marshalled to transfer RAM $3000{:}1F50_H$

Word superglobal values:

AW 18,00	is marshalled to transfer RAM $3000{:}1F52_H$
EW 18,00	is marshalled to transfer RAM $3000{:}1F52_H$
.	.
.	.
.	.
AW 22,15	is marshalled to transfer RAM $3000{:}1FF0_H$
EW 22,15	is marshalled to transfer RAM $3000{:}1FF0_H$

12.2 Addressing binary superglobal groups

A binary output group (16 bits, A xx,00...A xx,15) is written in words. This is why only ever *one* PLC processor may write to *one* output group because several writing processors would otherwise mutually overwrite the output values. It is also irrelevant whether the binary output group is written to an output module or to the superglobal value memory.

However, access by both programs of *one* PLC is only allowed to *one* output group. The reason for this is that both PLC programs only specifically alter the bits that occur in the respective program.

- **Forbidden:**

It is forbidden for PLC n and PLC m to write to the **same** binary output group

PLC n: !M00,00 = A00,01
PLC m: !M12,00 = A00,02

13 TURBO operation

TURBO operation substantially shortens the program execution time for bit and word processing in comparison with the interpretatively operating NORMAL method.

13.1 TURBO concept

- User programs are stored in the PLC in an intermediate code which is executed interpretatively by the control.

- The intermediate code is the basis for
  - execution of the PLC program in NORMAL mode
  - feedback documentation of the PLC program directly out of the PLC
  - translation of the PLC program into the machine language required for TURBO mode

- If the user has configured TURBO mode for a PLC program, this PLC program is translated to machine language when the program is started. All necessary parameters such as flag addresses are incorporated into the machine language by the translating program. Before translation of the PLC program into the machine language, a RAM test is carried out for the TURBO RAM. If the PLC detects a RAM error, an error message is issued and the user program is not started.

- The following are translated:
  - Boolean instructions including brackets (bit processing)
  - Arithmetic instructions including brackets (word processing)
  - Instructions for word comparisons including brackets
  - Jump blocks
  - Run number blocks

13.2 Testing the PLC program

When the test functions

- single step and
- breakpoint

are called, the PLC switches automatically to NORMAL mode and, at the same time, multiplies the cycle time by four. This occurs because the breakpoint and single step test functions can only be processed in NORMAL mode. When these test functions are deactivated, the PLC then automatically returns to TURBO mode and the cycle time is again set to the original value (KD 0,0). This internal PLC measure normally does not manifest itself externally to the user.

Exception: a breakpoint is set at a program position which, owing to a program jump, for instance, is *not* processed. In this case, the program runs as normally, but with four times the cycle time, which may have a detrimental effect on the functions.

When leaving these test functions, the TURBO mode is again called for the PLC.

13.3 Advantages of this procedure

- Accelerated execution of the user program.

- No restriction in terms of the operator control and test functions, i.e. the user interface is retained completely.

- The user may run a program in NORMAL mode or in TURBO mode as required.

- Program changes and feedback documentation are carried out on the basis of the intermediate code.

- Any required mixing of:
  - bit processing
  - word processing
  - block calls

13.4 Execution time data

- Processor clock: 8 MHz

Processing mode		Normal	TURBO
Binary processing			
• Pure program computing time for 1 k instructions		approx. 16 ms	approx. 2 ms ... 2.3 ms
Word processing			
• Pure program computing time for 1 k instructions			
Sentence begin	(!)	16 ms	2 ms
Addition	(+)	16 ms	< 5 ms
Subtraction	(-)	16 ms	< 5 ms
Multiplication	(*)	28 ms	< 23 ms
Division	(:)	29 ms	< 24 ms
Allocation	(=)	16 ms	< 5 ms
Comparison	(<, >, =? etc.)	16 ms	< 5 ms
Management:			
• Operating system + DBV management per program cycle		approx. 0.75 ms	
• Read and output of process image per program cycle		Binary input:	0.05 ms per input group
		Binary output:	0.065 ms per output group
		Input/output word:	0.013 ms per word

14 Operation of the PLC via the MPST bus

Instead of using serial interface, the PLC can also be operated and tested through the MPST bus.
In this case, the commands for the operation and test functions are handled by way of two SENTENCE structures in the PLC's transfer RAM. The PLC no longer expects the operator control and test commands through the serial interface, but by way of the SENTENCE structure for input commands. The PLC responds to the received commands in the SENTENCE structure for output.

14.1 Communication path between the PLC processors

Each PLC A is able to establish a communication path to a PLC B. In doing this, all commands received through the serial interface of PLC A are forwarded to the SENTENCE structure for input of PLC B. All replies of PLC B are read out of its SENTENCE structure for output and are forwarded to the serial interface of PLC A (see also Chapter "Operator control and test functions, command: $).

During commissioning and test of a control system with several PLC processors, the terminal's or personal computer's interface cable therefore does not need to be reconnected.

Here, the commands WARM (warm start) and KALT (cold start) are forbidden because this results in initialization of the destination PLC's transfer RAM. This initialization is disturbed by virtue of the fact that the other PLC continuously writes into the transfer RAM while the RAM test is running there. The destination PLC is thus deceived into thinking a RAM error has occurred, and this is also reported.

14.2 Description of the logical interface in the transfer RAM

The following description is relevant only if a user wishes to operate the PLC through the logical interface from any intelligent BUS station (not PLC).

The logical interface in the transfer RAM is realized with two SENTENCE structures conforming to DIN 66264. Commands are entered into the PLC by way of *one* SENTENCE structure and output of the PLC reactions takes place through the *other* SENTENCE structure.

In the data block directory (DBV), the pointers are set to the two SENTENCE structures. That is to say, by changing the pointers in the DBV, the SENTENCE structures can also be placed in any chosen location within the entire address area. If the pointers in the DBV are not changed, the SENTENCE structures are located in the PLC's transfer RAM. The pointers to the SENTENCE structures are located as of address 1000_H:$0ED8_H$ in the DBV (see also chapter entitled "Data block directory").

Contents of the data block directory (DBV) at address:

1000_H:$0ED8_H$ The offset address of the SENTENCE structure for *input* of commands into the PLC is located here.

1000_H:$0EDA_H$ The segment address of the SENTENCE structure for *input* of commands into the PLC is located here.
Important: If the value 0 is located here, this means that the PLC is handling *input* and *output* through the serial interface.

1000_H:$0EDC_H$ The offset address of the SENTENCE structure for *output* of replies from the PLC is located here.

1000_H:$0EDE_H$ The segment address of the SENTENCE structure for output of replies from the PLC is located here if outputs are not taking place through the serial interface.

With operation, the PLC offers a great amount of flexibility. *Input* of commands and *output* of the replies can therefore take place in the following way:

- Input/output through the serial interface
- Input/output via the SENTENCE structures

Switching over the operator control interface between the serial interface and the MPST bus

The pointer in the data block directory to the sentence structure input defines whether the PLC is to be operated through the serial interface or via the MPST bus. By way of default, this pointer is set so that it is operated via the serial interface. Switchover to the MPST bus is achieved by means of a corresponding entry in the transfer RAM. The memory word with the address $0EDA_H$ is the crucial element here.
The following applies:
$0EDA_H$ = 0: ---> operation via serial interface
$0EDA_H$ = 1000_H: ---> operation via MPST bus

This memory word is addressed as follows:

- from the PLC:
  1000_H:$0EDA_H$

- from the MPST bus:
  active area and station address of the PLC:$0EDA_H$

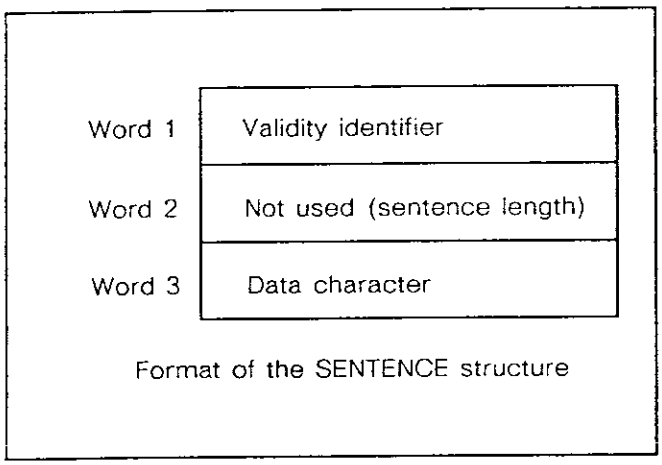
Formats of the SENTENCE structure

The SENTENCE structure comprises three words:

Word 1: Validity identifier
0 ---> Data character invalid
1 ---> Data character valid

Word 2: Not used (sentence length)

Word 3: Data character



Format of the SENTENCE structure

Handling communication

Communication is character-by-character.

Writing a character to the sentence structure for **character input:**

A data character may *only* be entered in word 3 if the validity character = 0. Validity character = 0 means that the character written previously has already been evaluated by the PLC.

After entry of the data character in word 3, the validity identifier of the one making the entry is set to 1. For the PLC, this means that a valid character is now awaiting evaluation.

Reading a character from the sentence structure for **character output:**

A data character may only be read out of word 3 if the validity identifier = 1. For the one reading out, this means that a valid character is now awaiting evaluation.

After readout of the data character from word 3, the validity identifier of the one reading out the character is set to 0. For the PLC, this means that it can provide a new character in word 3 for the one reading out.

15 Serial Interfaces

Interface specification: RS-423 and RS-232-C

15.1 Basic initialization: operator interface and interface 1

After activation of the PLC, the operator interface (top interface) and interface 1 (bottom interface) are set to the following characteristic data:

Data bits:	8
Stop bits:	1
Parity:	NONE
Synchronization:	RTS/CTS hardware signals

With the setting mentioned above, the module connected to the PLC can, however, also be operated with the following characteristic data:

Data bits:	7
Stop bits:	1
Parity:	SPACE

15.1.1 Additional 35 DS 91 module

An additional 35 DS 91 module can be connected if further interfaces are needed. The interfaces belonging to the additional 35 DS 91 module are *not* initialized automatically when the PLC is activated. Initialization of the interfaces as well as the activation of sending and receiving buffer software for the DRUCK and EMAS blocks (see also section 15.6) is realized exclusively with the SINIT module.

15.1.2 Interface initialization by the user

Interfaces 1...3 can also be initialized and operated with the blocks of IOW and IOR or the read/write port monitor command. In this case, the SINIT block must not be used, because the sending and receiving buffer software is also activated by it.

The basic initialization (see also chapter 15.1) of the required interface is carried out as follows by writing a string of initialization parameters into the status register:

1. Writing the status register with the value: 4
2. Writing the status register with the value: 44_H
3. Writing the status register with the value: 3
4. Writing the status register with the value: $0E1_H$
5. Writing the status register with the value: 5
6. Writing the status register with the value: $0EA_H$
7. Writing the status register with the value: 1
8. Writing the status register with the value: 12_H
9. Writing the status register with the value: 2
10. Writing the status register with the value: $0A_H$

15.1.3 PORT addresses of the serial interfaces

The interface designations are defined in section 15.7.

8-bit data are output from the PLC through the interface by writing into the send register.

8-bit data are read from the interface into the PLC by reading the receive register.

The status register serves to initialize the interface and provides the following information during operation about its current state:

Status register Bit 0 = 1: Interface has received a new character

Status register Bit 2 = 1: Interface is ready for output of the next character

Operator interface:
- Address of the send/receive register: 100_H
- Address of the status register: 104_H

Interface 1:
- Address of the send/receive register: 102_H
- Address of the status register: 106_H

Interface 2:
- Address of the send/receive register: 400_H
- Address of the status register: 404_H

Interface 3:
- Address of the send/receive register: 402_H
- Address of the status register: 406_H

15.2 Synchronization

The synchronization of the data flow takes place:

- always by way of the hardware signals of CTS and RTS, in which the RTS line of the connected module is linked to the CTS input of the PLC. The PLC's RTS line is linked to the CTS input of the connected module.
- additionally via XON / XOFF: this applies only to the operator interface and to interface 1 when a printer or EPROM programmer is connected.

The following must be observed with XON / XOFF:
If the module connected to the PLC is *not* operating the CTS/RTS lines, the following applies:

The PLC itself does not send XON/XOFF but reacts to these characters when they are sent by the connected module (e.g., printer, programmer, terminal, etc.).

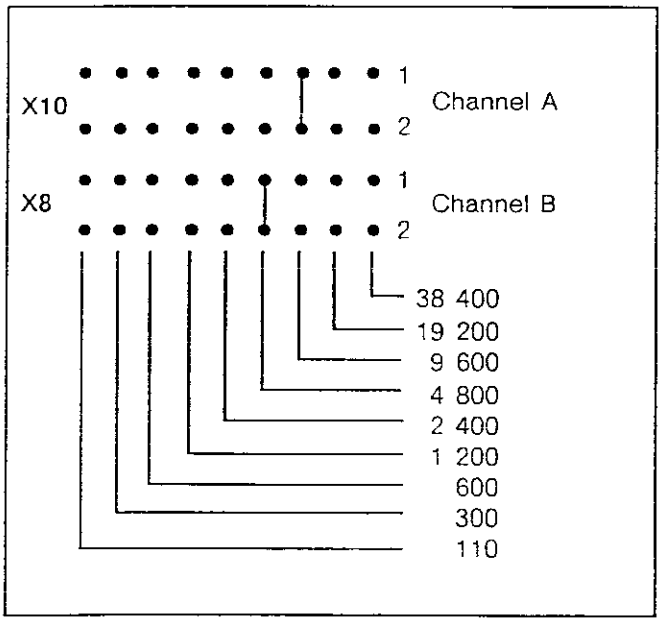
The following also applies with XON / XOFF:

- After sending an XOFF character, the connected module must only send the characters of XON or CTRL C (Abort). Other characters received after an XOFF character may be lost (error message: RECEIVE BUFFER OVERFLOW).

- During a PLC output through the operator interface, the PLC may receive and store a maximum of *three* characters from the connected module (keyboard, PC, etc.). All further characters (also XOFF, for instance) from the connected module are lost.

15.3 Setting the baud rates

The baud rates for the serial interfaces are set separately for channel A (top interface) and channel B (bottom interface) on connectors X8 and X10.



The following is set in the example:

Channel A: 9600 baud
Channel B: 4800 baud

15.4 Pin assignment

Pin	Signal name	Meaning
1	Screen	Protective Ground
2	TxD	Transmit Data
3	RxD	Receive Data
4	RTS	Request To Send
5	CTS	Clear To Send
7	GND	Ground
20	DTR	Data Terminal Ready

15.5 INTEL HEX file for creating the user program or comment EPROMs

The user program or comment from the RAM is transferred to the EPROM programmer through serial interface 1 (bottom interface) by means of the PU command. The transmission protocol has the INTEL HEX FILE format. In this case, the data are transferred in individual records. A record consists exclusively of ASCII characters and has the following structure:

Format of the INTEL-Hex files

:	L2 L1	A4 A3 A2 A1	TT	D0 D1 D2 ... Dn-1 Dn	S2 S1
(1)	(2)	(3)	(4)	(5)	(6)

(1) Record marker: The start of the record is marked by the ASCII character of :.

(2) Record length: Number of the data bytes "00" to "FF" corresponds to "30 30" to "46 46" in the ASCII code. The more significant hex digit (L2) is sent first.

(3) Start address: Start address from which the data to be transferred must be stored by the recipient. The more significant byte of the address is transferred first.

(4) Record type:
TT = 00: Data record
TT = 01: End record
TT = 02: Address extension record
TT = 03: Start address record

Only data and end records are generated.

(5) Data: In a word-by-word transfer, the more significant byte is sent first.

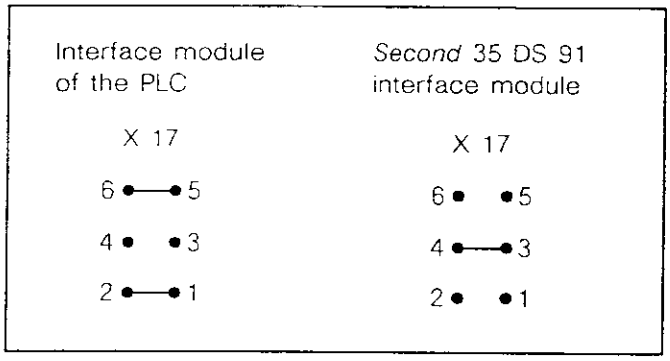
(6) Check sum: Two's complement of the sum of all bytes except the record marker, i.e., from the record length to the last data byte.

15.6 Connection of an additional 35 DS 91 interface module

Further serial interfaces may be necessary when using the EMAS and DRUCK modules. A second 35 DS 91 module is connected to the PLC in this case.

In this case, a setting is made on jumper panel X17 of the interface board so as to indicate that the module is an additionally connected 35 DS 91 module.

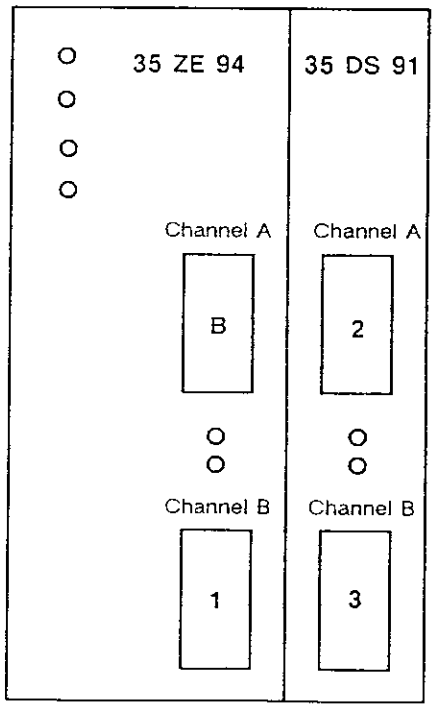
The following definitions apply:



If a second 35 DS 91 module is connected, its serial interfaces must be initialized by means of the SINIT block.

15.7 Operation of the interfaces by the PLC

Configuration



The PLC can operate the following serial interfaces:

- B: Operator interface B
- 1: Interface 1
- 2: Interface 2 (additional 35 DS 91 module)
- 3: Interface 3 (additional 35 DS 91 module)

Use of the interfaces

- Operator interface B:
  Connection of a PC or terminal for operation and test of the PLC.
- Interface 1:
  Either connection of a module which is capable of addressing by means of an operator control command (e.g., programmer, printer) or suitable for the blocks of DRUCK and EMAS.
- Interface 2:
  Suitable for the blocks of DRUCK, EMAS.
- Interface 3:
  Suitable for the blocks of DRUCK, EMAS.

Marginal conditions

- All interfaces are operated by the PLC software by means of "polling".
- All interfaces operate with hardware handshakes, (in certain circumstances, operator interface B and interface 1 may also additionally operate with XON/XOFF; see section 15.2).

Processing: serial interfaces 1...3 and operator interface

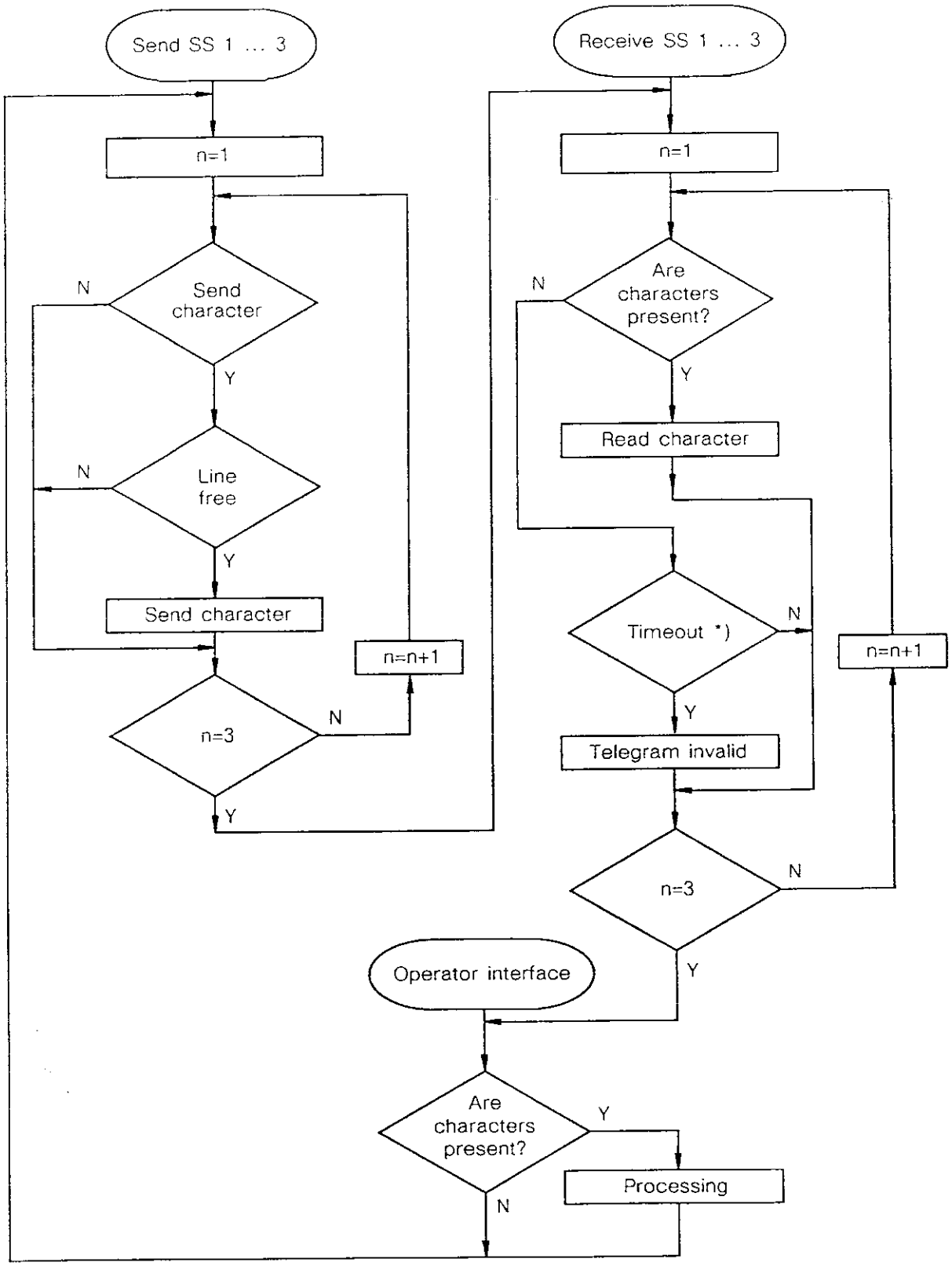
Operation with the serial interface has a lower priority than operation with the user programs. Therefore the operation of the serial interface has no influence on the program cycle times. Only that processor capacity, which is not occupied by the user programs, is available for the operation of the interfaces. If the PLC is little occupied by operating the user programs, the interfaces are more intensively used.

If interfaces 1...3 are used by the blocks of DRUCK / EMAS, the interfaces are processed sequentially by the PLC software in the following order:

1. Interface 1: Send direction
2. Interface 2: Send direction
3. Interface 3: Send direction
4. Interface 1: Receive direction
5. Interface 2: Receive direction
6. Interface 3: Receive direction
7. Operator interface B

Important:
If the period *between two characters* is more than 20 seconds when receiving the telegram by the EMAS block, the procedure is aborted by a "Timeout" and the telegram is lost.

The following flow chart shows the sequence of processing of the serial interfaces by the PLC software.



Processing: Serial interface n and operator interface, with: n = 1...3

*) The timeout applies only between two characters *within* one telegram.

15.8 Interface conversion from RS-232-C to RS-422

An interface conversion from RS-232-C to RS-422 allows a serial transfer over long distances.
The interface conversion described in the following allows the ABB Procontic T300 to be coupled with the programming system (907 PC 32), for example, over long distances (up to 2400 m for 19.2 kbaud).

Cables

The required connecting cables must be made by the user.
The signal assignment to make the cables was composed from the information in the data sheets from the manufacturer of the devices. The user is recommended to check whether the information is still up to date for his case of application.

Converters

Interface converters from Messrs Wiesemann und Theis GmbH are used.

Components:

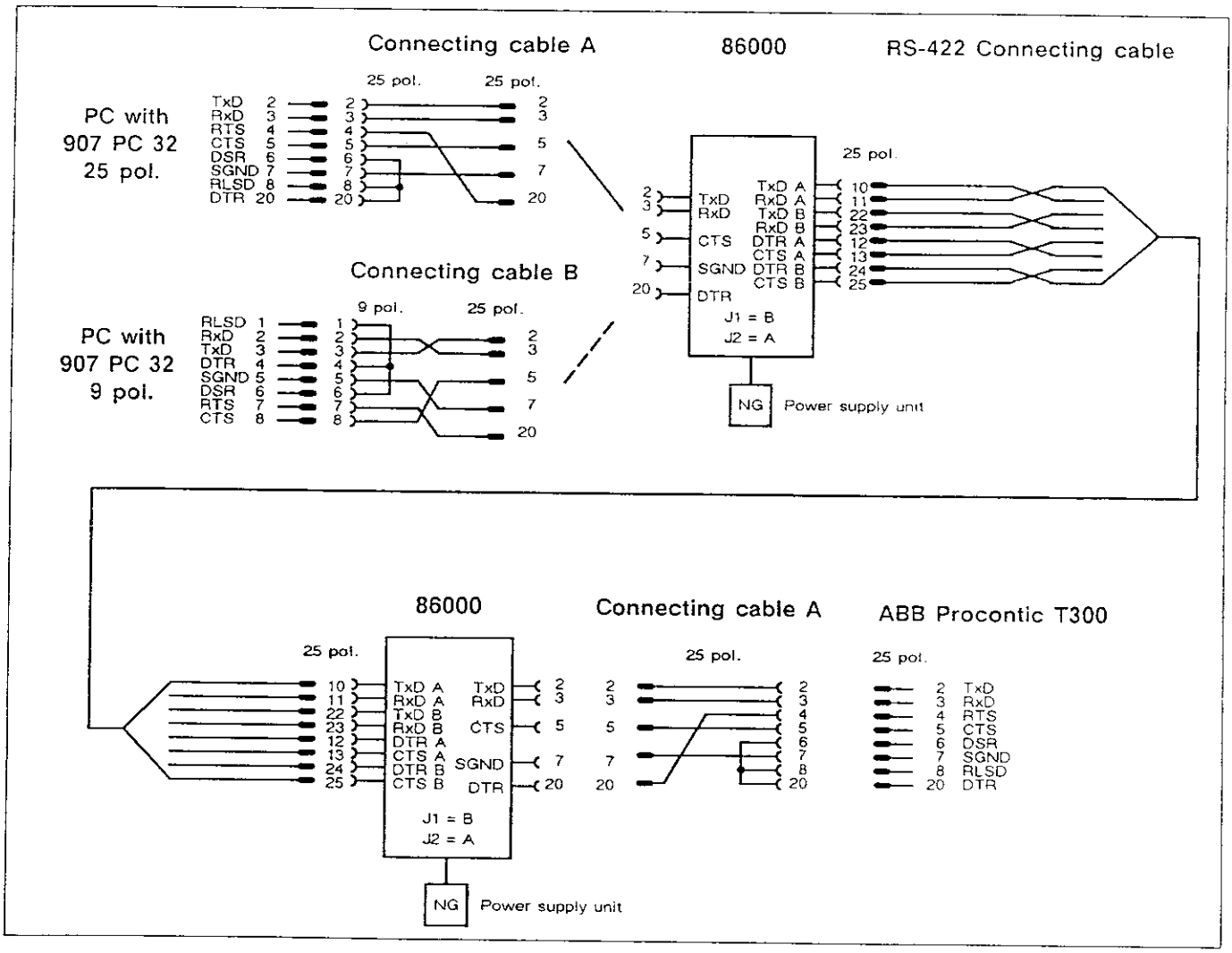
- 2 interfaces (RS-232-C/ RS-422 converters)
- 2 power supply units

Art. No.: 86000

Company address:

Wiesemann und Theis GmbH
Winchenbachstr. 3-5
W-5600 Wuppertal 2
Germany
Tel.: 0202/505077

Signal assignment when converting interfaces



RLSD	Receive Line Signal Detector
RxD	Receive Data
TxD	Transmit Data
DTR	Data Terminal Ready
DSR	Data Set Ready
RTS	Request To Send
CTS	Clear To Send
SGND	Signal Ground

NG	Power supply unit for interface converter
J1/J2	Jumpers on the interface converter

16 PLC controlled by a central control unit (ZST)

In the *PLC WITH ZST* operating mode, the PLC is initialized and the user programs are started under the control of a central control unit (ZST). The function of a central control unit can be assumed by another active station. For instance, this may be an industrial computer, another PLC or a special ZST station.

16.1 Initialization of the PLC

In this operation mode, the PLC is *not* initialized automatically after power-on or after a WARM START/RESET. It waits until it is guided through the initialization phase by the ZST in a handshake procedure.

Important:

If the ZST does not log in, the PLC's waiting state can be aborted by entering <CTRL>C via a connected terminal.

The PLC is controlled by the ZST for the following reasons:

- Controlled start-up of all active stations existing in the MPST system (no undefined states in the system owing to "racing" of individual active stations).
- The central control unit can send data to the PLC during initialization, thus influencing the initialization process. In this way, the ZST can define the marshalling between the PLC and I/O modules, for instance. Unless the user has altered it, the default data block directory (DBV) is the basis for remarshalling of the I/O modules by the ZST. If the user has altered the default DBV, the ZST finds this altered DBV in the transfer RAM as the basis.

The PLC is initialized by the ZST in two parts.

Part 1:

General initialization of a function block in accordance with DIN 66264. The initialization in part 1 is terminated when the PLC enters the identifier of 80_H in the control block 0, byte 4.

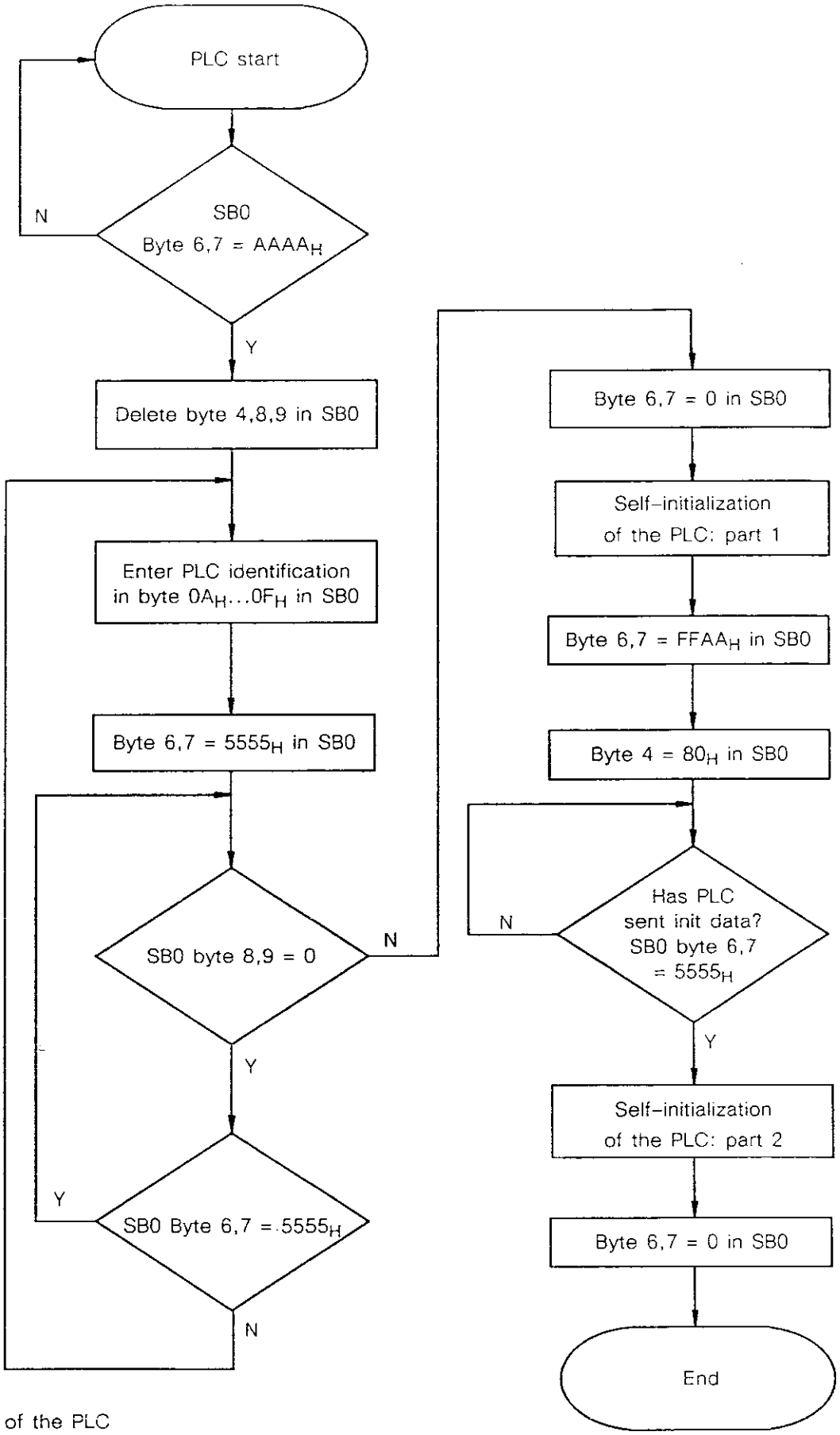
Part 2:

The general initialization in accordance with part 1 is followed by initialization specific to the function blocks. In this case, the PLC takes into account the initialization data sent by the ZST.

The complete initialization from the point of view of both the PLC and the ZST is shown in the following two figures. The initialization in accordance with DIN 66264 has been extended by the function of "Send init data to the PLC".

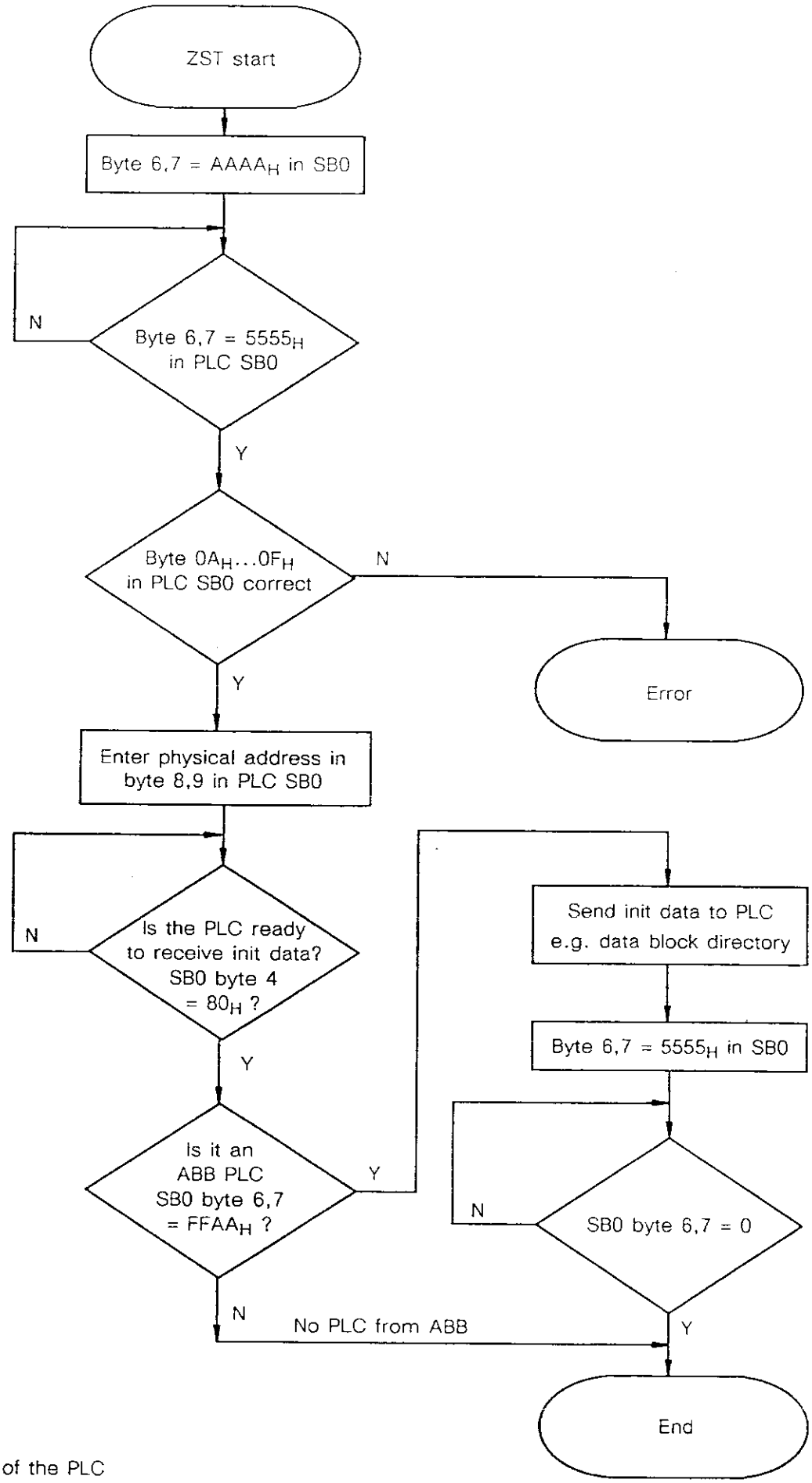
Starting the programs is defined in conformity with DIN 66264, part 2 (see also section 16.2).

Initialization from the PLC's point of view



SB0: Control block 0 of the PLC

Initialization from the ZST's point of view



SB0: Control block 0 of the PLC

16.2 Assigning jobs to programs

Within the meaning of DIN 66264, each of the two user programs is a *function that can be assigned to jobs* externally. That is to say that the programs can be
- started
- aborted
- continued and
- stopped

by the central control unit (ZST).
There are three *control blocks (SB0...SB2)* in the PLC transfer memory for realization of this *job assignment*.

By way of the control blocks, jobs are communicated to the PLC and acknowledgements are returned to the ZST. Each PLC has a control block 0 (SB 0) and one further control block (SB 1 and SB 2) for each of the two PLC programs.

16.2.1 Definition of the control blocks

Control block 0

Control block 0 begins from address 0 of the transfer memory and has the following structure:

Byte	Designation	Bit	Value	Meaning
0 1	SB pointer	0 : 7	1 or 2	Number of the PLC program to be assigned a job
2 3	not used			
4 5	PLC status Allowed program	7 8 : 15	1 0 or 1 or 2	PLC init. finished 0: both progr. permitted 1: Prog. 1 perm. 2: Prog. 2 perm.
6 7	Error message	0 : 7 8 : 15		Error number Number of the faulty PLC program or FF_H
8 9	not used			
A B C D E F	PLC identification		42 42 43 06 00 00	Manufacturer identifier

Control block pointer

The ZST enters the number of the PLC program to which a job is to be assigned (1 or 2) in byte 0, bits 0...7.
The entry (e.g., start program) made previously in the allocated in the control block (SB1 or SB2) is rendered valid by this entry.
The PLC acknowledges recognition of the job by reading and clearing the control block pointer.

PLC status

The PLC enters the value of 80_H in byte 4, bits 0...7 when it has initialized itself to such an extent so that it is ready to accept initialization data (e.g., data block directory) from the ZST.

Permitted program

From version V 8.0 onwards, the PLC allows changes to the running PLC program. In this case, when assigning jobs the ZST may only use the program number which the PLC has entered in byte 5.

The following applies:

Byte 5 = 0: The ZST may assign jobs to both programs 1 and 2 because the PLC is not in the *CHANGE RUN* operating mode.

Byte 5 = 1: The PLC is in the *CHANGE RUN* operating mode and the ZST may only assign jobs to program 1.
Only one program is possible on the PLC in the *CHANGE RUN* operating mode and this is currently in program memory 1.

Byte 5 = 2: The PLC is in the *CHANGE RUN* operating mode and the ZST may only assign jobs to program 2.
Only one program is possible on the PLC in the *CHANGE RUN* operating mode and this is currently in program memory 2.

Error message

Every error recognized by the PLC is assigned an error number. This error number is entered in byte 6 and remains there until a new error is diagnosed. In byte 7 the PLC enters the number of the program to which the error is related. If an error has occurred which is not related to a PLC program, the PLC enters the value of FF_H in byte 7 (see also the chapter entitled "Self-diagnosis and reactions to errors").

During the start-up of the PLC by the ZST, bytes 6 and 7 are also used when realizing a handshake between the PLC and the ZST. This overwrites any previous error message.

PLC identification

The identifier designating the PLC manufacturer is located in byte A_H to F_H. They contain the following entries:

Byte A_H: 42_H
Byte B_H: 42_H
Byte C_H: 43_H
Byte D_H: 06_H
Byte E_H: 00_H
Byte F_H: 00_H

Control block 1 and control block 2

When jobs are assigned to the PLC programs by the ZST, control block 1 is allocated to program 1 and control block 2 to program 2. The two control blocks follow control block 0 directly in ascending order in the transfer memory.

The control blocks 1 and 2 have the following structure:

Byte	Designation	Bit	Value	Meaning
0	Job identifier	0	1	Start
		1	1	Stop
		2	1	Continue
		3	1	Abort
1	not used	8 : 15		
2 3	not used			
4 5	DBV pointer	0 : 15		16 bit pointer to the data block directory in the transfer RAM
6	Permissible job	0	1	Start
		1	1	Stop
		2	1	Continue
		3	1	Abort
7	SPS status	8	1	Started
		9	1	Stopped
		10	1	Aborted
8 : B	For SB1: TRACE memory For SB2: not used			TRACE memory - Progr. No. - Progr. Addr.
C D	Cycle counters		0000 to FFFF	Counters for the program cycles
E	Init flag 1	7	0/1	invalid/valid
F	Init flag 2	7	0/1	Init type

Job identifier

The central control unit (ZST) enters the job identifier in order to achieve its desired status change in the affiliated PLC program.

The following are defined as jobs:

- START PROGRAM
- STOP PROGRAM
- CONTINUE PROGRAM
- ABORT PROGRAM

DBV pointer

The PLC enters the 16 bit address of the data block directory in bytes 4 and 5. The address refers to the start of the PLC transfer memory. Thus, the ZST is able to make changes to the data block directory.

Permissible job

The PLC enters in byte 6 which jobs are currently permissible. *Before* assignment of a job, the ZST must check whether the intended job is momentarily allowed by the PLC.

PLC status

The PLC enters the current status of the affiliated PLC program in byte 7.

The following statuses are defined:

- PROGRAM STARTED
- PROGRAM STOPPED
- PROGRAM ABORTED

The program status can also be indicated by the commands of PS <CR> and ST <CR> (see also the chapter entitled "Operator control and test functions").

TRACE memory

The PLC enters the number of the currently processed PLC program in byte 8 in the TRACE mode and the last executed program address in bytes A_H and B_H. The TRACE memory is only defined in control block 1. This memory locations are not used in control block 2. The TRACE memory can also be indicated by means of the commands of TRACE <CR> and ST <CR> (see also the chapter entitled "Operator control and test functions").

Cycle counter

The PLC increments the cycle counter with each program cycle executed in the affiliated PLC program. For example, the cycle counter indicates whether or not the affiliated PLC program is actually running.

Init flag 1

Init flag 1 specifies whether or not init flag 2 is valid, i.e., whether or not the configured mode of INI-OPS/ NOINI-OPS has been rendered inactive by init flag 2.

Init flag 1: 0 ---> Init flag 2 is invalid and the configured mode applies.
1 ---> Init flag 2 is valid and the initialization mode in init flag 2 applies.

Init flag 2

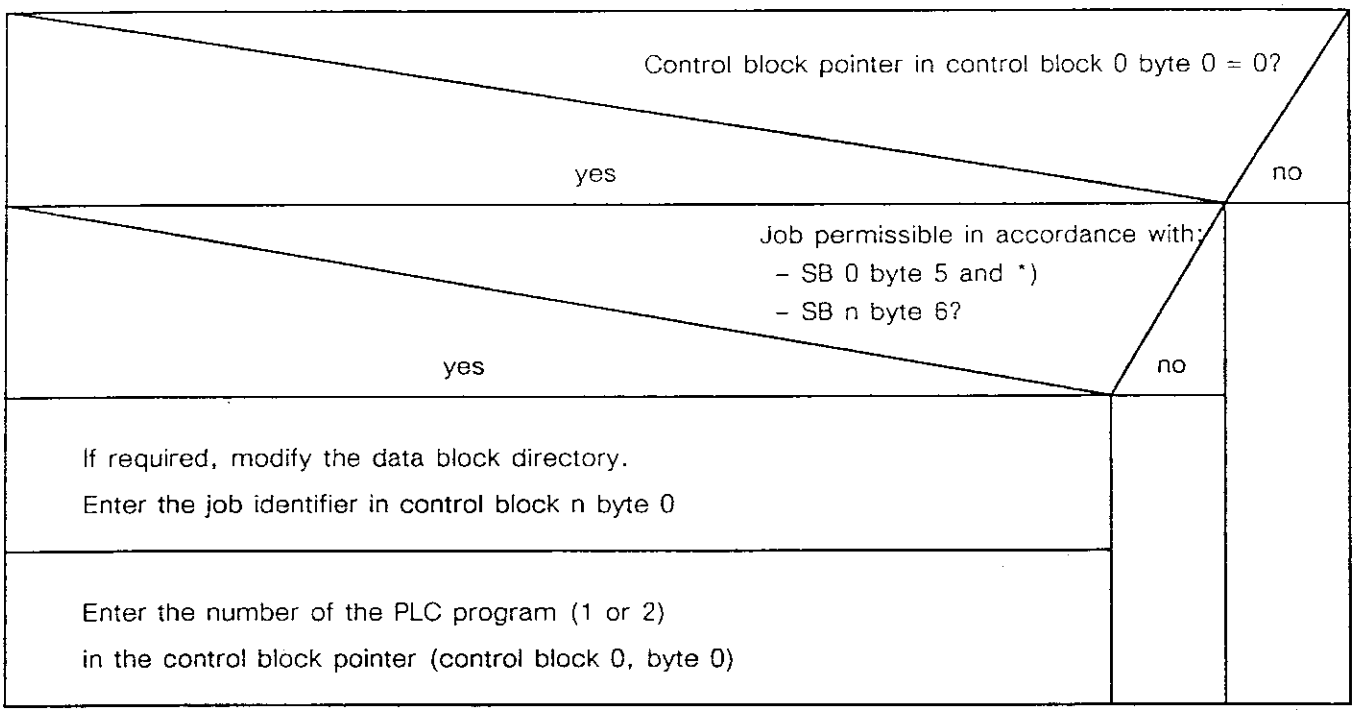
The initialization mode (INI-OPS/NOINI-OPS) can be defined with init flag 2. It has a higher priority than the operating mode configured by means of KONFS.

Init flag 2: 0 ---> No initialization (NOINI-OPS) of the operand memory
1 ---> Initialization (INI-OPS) of the operand memory

16.2.2 Realizing job assignment

The method of job assignment of a PLC program is defined exactly. Both the central control unit and the PLC must keep to this definition. Job assignment from the point of view of the central control unit and from the point of view of the PLC is shown by means of two structural diagrams.

Job assignment of the PLC program of n from the point of view of the central control unit (ZST)



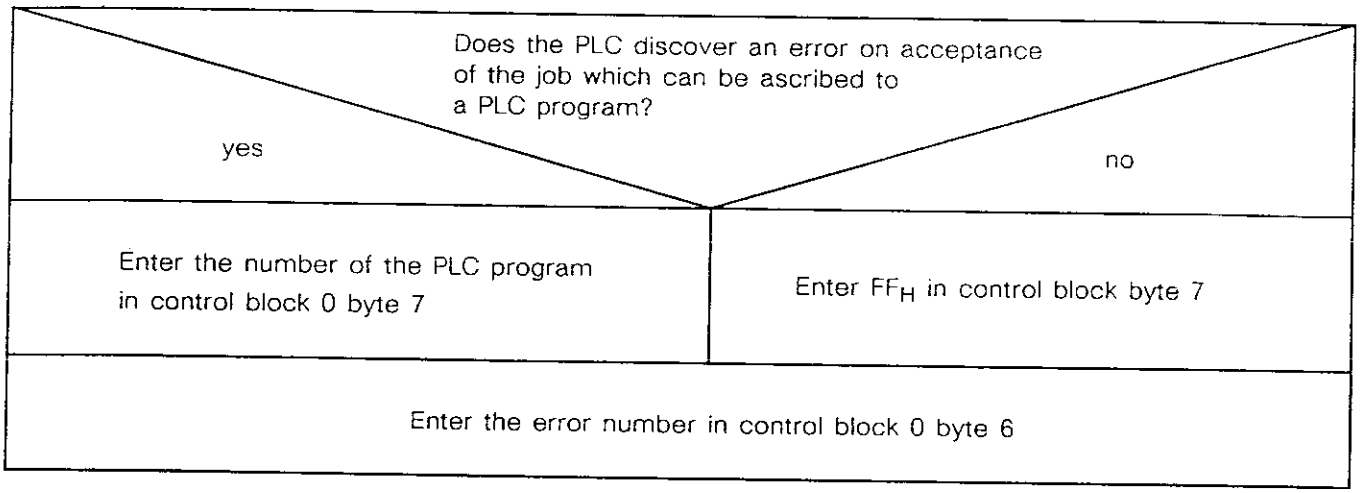
*) The PLC is in the *CHANGE RUN* operating mode if control block 0 byte 5 is not equal to 0. In this case only *one* program is possible in the PLC but its current incarnation may alternately be in program memory 1 or 2. The information relating to the program memory in which the PLC program is currently located is contained on control block 0, byte 5. Therefore, in addition to DIN 66264, the following applies to the central control unit:

SB 0 byte 5	The ZST may specify the following program number
0	1 and 2
1	1
2	2

Acceptance of a job from the central control unit (ZST) by the PLC

Accept the program number from control block 0 byte 0 and read job out of the affiliated control block (SB 1 or SB 2)
Check validity of the job in accordance with byte 6 of the affiliated control block
Enter the new permissible job in byte 6 of the affiliated control block
Clear control block pointer in control block 0 byte 0
Execute the job, i.e., start, stop, continue or abort PLC program
Enter the new PLC program status in byte 7 of the affiliated control block

Entry of the error number on acceptance of a job containing errors



17 Starting behaviour

As regards the starting behaviour of the PLC, a distinction is made between

- cold start and
- warm start.

The starting procedure is depicted in the following diagram.

A **cold start** is carried out when

- the voltage is connected to the PLC for the *first time* or
- the cold start command is issued explicitly by the user.

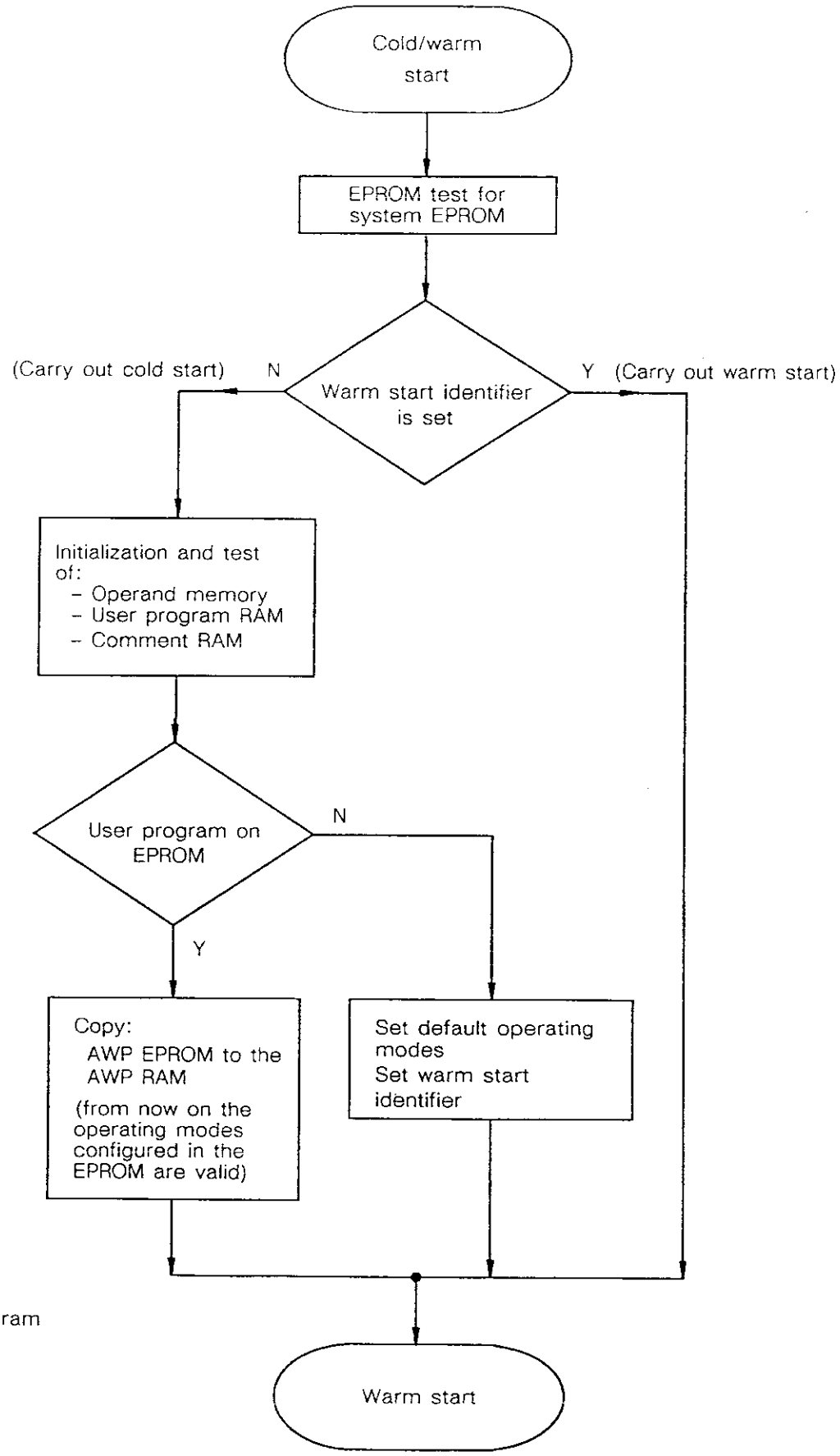
A **warm start** is carried out when

- the voltage is connected for the PLC (RESET) and the warm start identifier is set in the user program RAM. That is to say that a cold start has already been carried out once.
- the restart key has been pressed on the front panel
- the warm start command (WARM) is issued exclusively by the user. A warm start by means of the WARM command is not possible when the following modes are configured:
  - PLC with ZST and simultaneously
  - PLC mode.

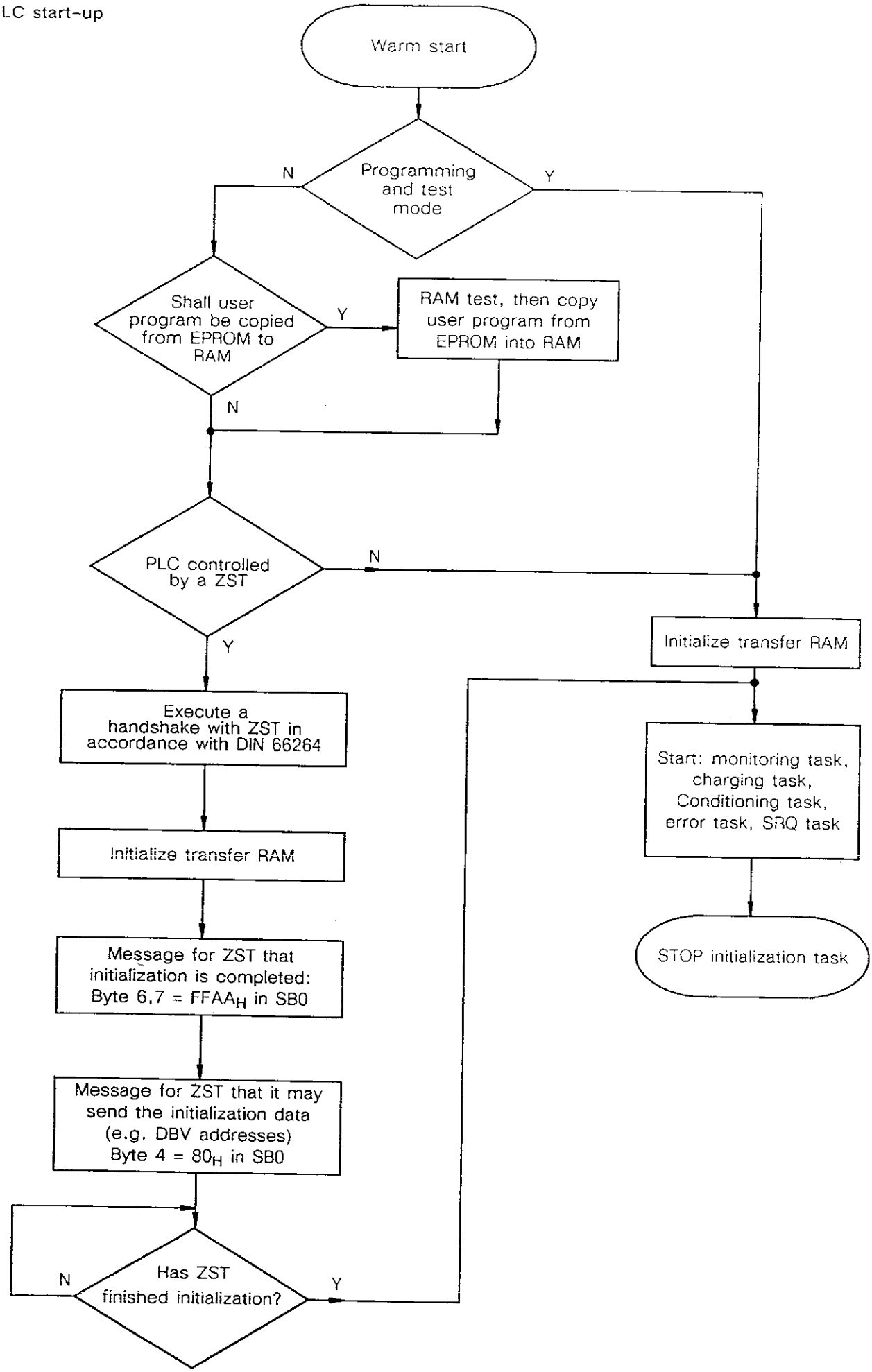
If this command is nevertheless entered, the PLC waits for initialization by the ZST. This waiting state can be aborted by entering <CTRL>C on a connected terminal.

Important:

The WARM (warm start) and KALT (cold start) commands must *not* be sent to a PLC by means of the $ function through the MPST bus. The fact is that a RAM test of the transfer RAM takes place during a warm and cold start and this is disturbed by the communication path and the PLC diagnoses a RAM error.



AWP = User program


Warm start
Programming and test mode
N
Y
Shall user program be copied from EPROM to RAM
Y
RAM test, then copy user program from EPROM into RAM
N
PLC controlled by a ZST
N
Y
Initialize transfer RAM
Execute a handshake with ZST in accordance with DIN 66264
Start: monitoring task, charging task, Conditioning task, error task, SRQ task
Initialize transfer RAM
Message for ZST that initialization is completed: Byte 6,7 = FFAA$_H$ in SB0
STOP initialization task
Message for ZST that it may send the initialization data (e.g. DBV addresses) Byte 4 = 80$_H$ in SB0
Has ZST finished initialization?
N
Y

18 Self-diagnosis and reactions to errors

The PLC carries out self-diagnosis measures both when starting and also during operation. If an error is detected, the PLC initiates an error reaction which, depending on its severity, may range from a display to deactivation of the PLC.

18.1 Error messages

Error messages are issued:

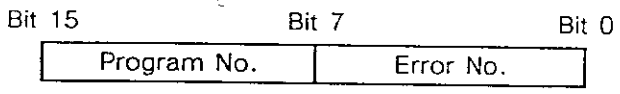
- in plain language on a connected terminal (if possible).
- with LEDs on the front panel of the PLC.
- as an error number in the error register of the control block of 0 in the transfer RAM (see also DIN 66264, part 2).

The error register is read out by means of both the command of FEHLER <CR> and also the command ST (PLC status), and its contents are displayed. The error register has the address of 6 in the transfer memory and can therefore also be read out through the MPST bus.

The last entered error number remains stored in the error register for as long as no power failure occurs.

As specified in DIN 66264, the error register is also used for the HANDSHAKE between the PLC and ZST on start-up of the PLC by the CENTRAL CONTROL UNIT (ZST). In this case (PLC controlled by ZST), any previously existing error message is overwritten during the HANDSHAKE. The ZST can, however, read out the error register (control block 0 of the PLC) before the HANDSHAKE.

Contents of the error register



Error No.:
Number of the last occurring error.

Program No.:
Number of the user program in which the error has occurred. If an error does not occur in conjunction with the user program, the HIGH BYTE of the error register contains the value of FF instead of the program number.

18.2 Error types and reactions

Error number: **01**
Error type: An invalid control block number has been specified for job-giving by the ZST.
Reaction: The job is ignored.
Message: Entry in error register: 01

Error number: **02**
Error type: The PLC program addressed by the ZST by job-giving is not loaded.
Reaction: The job is ignored.
Message: Entry in error register: 02

Error number: **03**
Error type: As indicated in byte 6 of the relevant control block, the job issued by the ZST is inadmissible, or there is just running a program substitution in the CHANGE RUN mode.
Reaction: The job is ignored.
Message: Entry in error register: 03

Error number: **04**
Error type: No cycle time is planned for the PLC program addressed by the ZST as the result of job-giving.
Reaction: The job is ignored.
Message: Entry in error register: 04

Error number: **05**
Error type: On job-giving by the ZST, an unknown job has been assigned.
Reaction: The job is ignored.
Message Entry in error register: 05

Error number: **06**
Error type: The PLC is overloaded. Together, the two user programs have a backlog of more than 16 program cycles in comparison with the planned cycle time or the occurring events.
Reaction: Both user programs are aborted and the PLC outputs are reset. Program number 2 is then set for the PLC.
Message: Entry in error register: 06
LED 3 is activated.

Error number: 07
Error type: More software timers are needed than available in the operating system of the PLC.
Reaction: Both user programs are aborted and the PLC's outputs are reset. Program number 2 is then set for the PLC.
Message: Entry in error register: 07
LED 3 is activated.

Error number: 08
Error type: During execution of the user program, the PLC meets up with an unknown operator or block type.
Reaction: Both user programs are aborted and the PLC's outputs are reset. Program number 2 is then set for the PLC.
Message: Entry in error register: 08
LED 3 is activated.

Error number: 09
Error type: A system error has occurred which cannot be specified in further detail (e.g. an invalid interrupt has been triggered during PLC initialization).
Reaction: Both user programs are aborted and the PLC's outputs are reset. Program number 2 is then set for the PLC.
Message: Entry in error register: 09
LED 3 is activated.

Error number: 0A
Error type: The POWER FAIL signal has responded.
Reaction: An interrupt is triggered for the PLC and the PLC immediately assumes STOP state. The PLC can be restarted only by a HARDWARE RESET or power OFF/ON.
Message: Entry in error register: $0A_H$
LED 3 and LED 2 are activated.

Error number: 0B
Error type: A checksum error in the system EPROMs is detected on starting the PLC. The stored and calculated checksums do not agree.
Reaction: Starting of the PLC is aborted and the PLC assumes STOP state.
Message: After input of any character on the keyboard, a plain language message is displayed on the terminal with details of the defective EPROM's mounting position designation.
Entry in error register: $0B_H$
"Running" light of LED 1, LED 2 and LED 3.

Error number: 0C
Error type: During start of the PLC an error is detected in the operand memory while RAM test was running.
Reaction: Starting of the PLC is aborted and the PLC assumes STOP state.
Message: After input of any character on the keyboard, a plain language message is displayed on the terminal with details of the defective EPROM's mounting position designation.
Entry in error register: $0C_H$
"Running light" of LED 1, LED 2 and LED 3.

Error number: 0D
Error type: A user program RAM error is detected during the RAM test while starting the PLC.
- When starting the PLC a RAM error is detected for mounting position XA6/XA7.
- When changing the program memory capacity from 56 kbyte to 120 kbyte or vice versa, a RAM error is detected for mounting position XA6/XA7.
- When copying a user program EPROM to the user program RAM, a RAM error is detected for mounting position XA6/XA7.

Reaction: The PLC assumes the STOP state.
Message: After input of any character on the keyboard, a plain language message is displayed on the terminal with details of the defective RAM's mounting position designation.
Entry in error register: $0D_H$
"Running light" of LED 1, LED 2 and LED 3.

Error number: 0E
Error type: A comment RAM error is detected during the RAM test on starting the PLC.This error is detected, if:
- the comment RAMs are defective
- no comment RAM has been plugged in the corresponding socket
- erased comment EPROMs have been plugged in the sockets

Reaction: Starting of the PLC is *not* aborted because the PLC can also be operated without the need for comments.
Message: Plain language message on the terminal along with details of the defective RAMs mounting position designation.
Entry in error register: $0E_H$

Error number: 0F
Error type: When setting the SENSOR mode, a RAM test is carried out for the sensor constants RAM (bank 70000... 77FFF). While doing this, an error is detected.

Reaction: The NO SENSOR mode is automatically set again.

Message: Plain language message on the terminal along with details of the defective RAM's mounting position designation.
Entry in error register: $0F_H$

Error number: 10

Error type: An *unused* interrupt has occurred after the PLC's initialization phase.

Reaction: None, the interrupt is ignored.

Message: Entry in error register: 10_H
LED 3 is activated.

Error number: 11

Error type: When executing the RAM test of the user program the PLC detects an error for the following procedures:
- When starting the PLC a RAM error is detected for mounting positions XA10/XA11.
- When changing the program memory capacity from 56 kbyte to 120 kbyte, a RAM error is detected for mounting position XA10/XA11.

Reaction: The program memory capacity is set on 56 kbyte.

Message: Plain language message on the terminal along with details of the defective RAM's mounting position designation.
Entry in error register: 11_H

Error number: A5

Error type: The TURBO translator detects a TURBO memory error during the RAM test.

Reaction: The relevant user program is not started. The other user program does, however, remain unaffected.

Message: Plain language message on the terminal along with details of the defective RAM's mounting position designation.
Entry in error register: $A5_H$

Error number: A6

Error type: While translating the user program into the machine code, the TURBO translator finds an unknown operator or block type.

Reaction: The relevant user program is not started. The other user program does, however, remain unaffected by this.

Message: Plain language message on the terminal along with details of the program address at which the error has occurred.
Entry in error register: $A6_H$

Error number: A7

Error type: The TURBO translator finds a bracket error at an arithmetic or boolean expression.

Reaction: The relevant user program is not started. The other user program does, however, remain unaffected by this.

Message: Plain language message on the terminal along with details of the program address at which the error has occurred.
Entry in error register: $A7_H$

Error number A8

Error type: The destination marker for a jumper block is missing.

Reaction: If the error is detected during starting of a PLC, this program will not be started. This error does not influence the other program.

Message: Plain language message on the terminal along with details of the program address at which the error has occurred.
Entry in error register: $A8_H$

Error number: A9

Error type: The program end is missing from a PLC program.

Reaction: If the error is detected during starting of a PLC, this program will not be started. This error does not influence the other program.

Message: Plain language message on the terminal along with details of the relevant program number.
Entry in the error register: $A9_H$

Error number: AA

Error type: The length of the historical value memory is not sufficient for the user program.

Reaction: If the error is detected during the starting procedure of a PLC, this program will not be started. This error does not influence the other program.

Message: Plain language message on the terminal along with details of the program address where the overflow has occurred.
Entry in the error register: AA_H

Error number: AB

Error type: Programming error in a block. The number of specified block parameters is incorrect.

Reaction: If the error is detected during the starting procedure of a PLC program, this program will not be started. This error does not influence the other program.

Message: Plain language message on the terminal along with details of the program address of the block start, the block type and the exact program address within the block where the error has been detected.
Entry in the error register: AB_H

19 Monitor functions

The monitor offers the specialist access at the hexadecimal level to the complete address area of the PLC. Memory areas can be displayed and modified and hardware tests can be performed.

Monitor commands which modify the memory areas may pose a risk to the PLC functions. This is why caution must be used when dealing with the monitor.

Switch over between operator control functions <---> monitor

Command:

Function:

The monitor program of the PLC is activated by simultaneously pressing the<CTRL> and W keys. Once you are in the monitor, you can return to the operator control program of the PLC again by entering <CTRL> W.

Syntax explanation:

- the monitor returns the character of * and waits for an input
- all number are hexadecimal numbers (leading zeros may be dropped)
- if more digits are specified than is necessary, only the last digits (the last 2 digits in byte commands and the last 4 digits in word commands) are valid
- the blank (space) character is ignored and can be used for more clearly understandable input
- the CTRL C character aborts the operation currently running
- all displays can be stopped by pressing <CTRL> S (XOFF) and continued with <CTRL> Q (XON)
- if no segment is specified when entering an address, the working segment is used (see Y instruction).

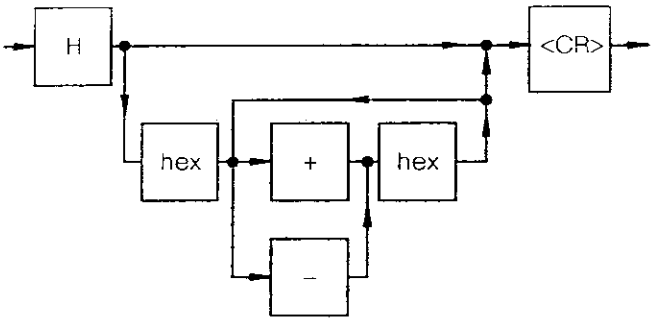
Summary of monitor functions

Function	Explanation	Page
H	Display help text/ calculate hexadecimally	19-2
D	Display memory contents	19-2
I	Fill memory area with value	19-3
M	Transfer memory areas	19-3
P	Read/write port	19-4
R	Read INTEL HEX file	19-4
S	Display/edit memory contents	19-4
U/V	Edit address output format	19-5
W	Write INTEL HEX file	19-5
Y	Display/edit work segment	19-5
ZA	Cyclic read and write	19-5
ZB	Cyclic read and write with waiting time	19-6
ZC	Read and write at key stroke	19-6
ZD	Cyclic write	19-6
ZE	Cyclic read	19-6
ZF	Cyclic write and read	19-7
ZG	Simultaneous output of 3 values	19-7
ZR	RAM test	19-8
ZT	Echo test of a serial interface	19-8
ZU	Short-circuit test of a serial interface	19-8
ZZA	Output of 3 values after input of a semicolon (;)	19-9
ZZF	Search for string	19-9
ZZV	Compare memory areas word by word	19-10

Display help text/calculate hexadecimally

After input of the command of H <CR>, all available functions of the monitor are displayed. This command also allows you to calculate simple hexadecimal arithmetic expressions.

Command:



hex: Hexadecimal value

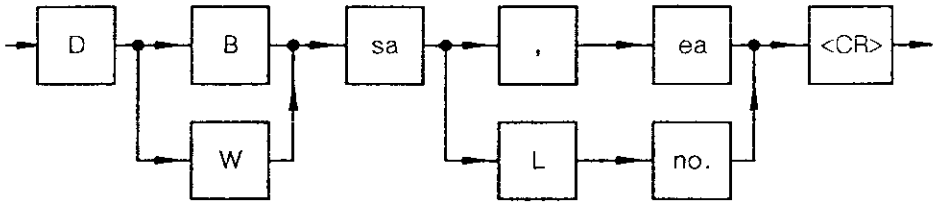
Example:

H <CR>	Display help text
H 3+4-2+A <CR>	Hexadecimal calculation
H 3+4-2+A <CR> = 000F	Display

Display memory contents

The memory contents can be displayed in bytes or words.

Command:



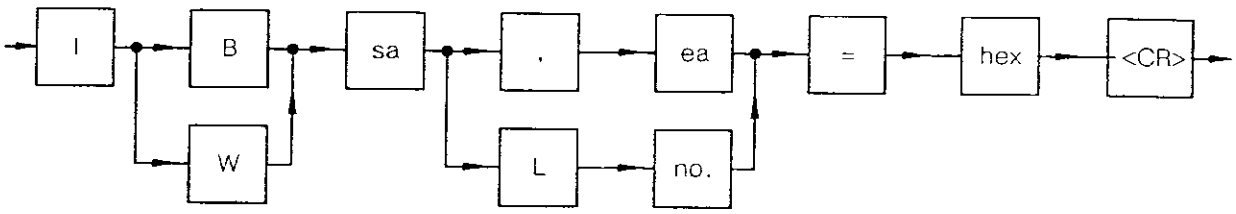
B: In bytes (key word)
W: In words (key word)
sa: Start address from which the memory contents are to be displayed
,: Key word (delimiter)
ea: End address of the memory contents to be output
L: Length (key word)
no.: Number of bytes/words to be output

Example:

DB 0:0L2 <CR>	Display memory contents in bytes
0000:0000 02 00	Screen display
DW 0,2 <CR>	Display memory contents in words
0000:0000 0002 0000	Screen display

Fill memory area with a value

Command:



B: In bytes (key word)
W: In words (key word)
sa: Start address from which the memory contents are to be filled with the specified value
,: Key word (delimiter)
ea: End address of the memory area
L: Length (key word)
no.: Number of the bytes/words to be filled
hex: Hexadecimal value with which the memory area is to be filled

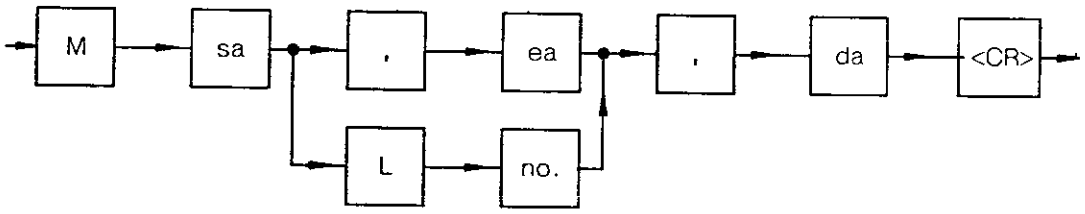
Example:

IB 8000:80L3=FF<CR> The memory contents of $8000:80_H$, $8000:81_H$ and $8000:82_H$ are overwritten with FF

Transfer of memory areas

A memory area can be copied to another area. The transfer takes place in words, whereby the number must be specified during input in bytes (i.e., 1 word is transferred with no. = 3).

Command:



sa: Start address from which the memory contents are to be copied
,: Key word (delimiter)
ea: End address of the memory area
L: Length (key word)
no.: Number of bytes to be copied
da: Memory area destination address

Example:

M 8000:80L4,8000:90 <CR> or
M 8000:80,84,8000:90 <CR>

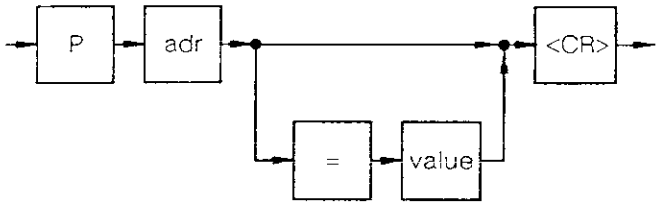
The following is copied:

$8000_H:80_H$ ---> $8000_H:90_H$
$:81_H$ ---> $:91_H$
$:82_H$ ---> $:92_H$
$:83_H$ ---> $:93_H$

Read/write port

A value out of the I/O area is displayed and edited in bytes.

Command:



adr: I/O address
value: Byte value which is written after the I/O address
=: key word

Example for the operator interface (address 100_H):

P 100 <CR> User command: read port

P 100=43 Screen display: the ASCII character of "C" (corresponding to 43_H) is read from the operator interface (I/O address of 100_H)

Read INTEL HEX file

Using the R command, it is possible to read in an INTEL HEX file through serial interface 1 (bottom interface) of the PLC and to store the HEX file data in the PLC. The following records (see the chapter on serial interfaces) are accepted:

- Address extension record
- Data record
- End record

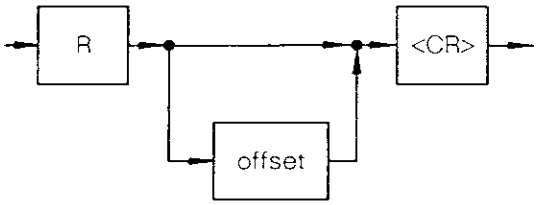
The following transmission format applies:

- 8 Data bits
- No parity bit
- 1 stop bit

The data of the INTEL HEX files is stored in the PLC from the following address:

- The segment address is determined by the address in the address extension record of the INTEL HEX file. If an offset is specified on input of the command, this offset is added to the segment address in the address extension record. A new segment address is thus produced, from which the data of the HEX file are stored. This allows you to specify the storage area for the HEX file data in the PLC.
- The offset address is determined by the address in the data record of the INTEL HEX file.

Command:



offset: Offset (with the segment address of the address extension record, resulting in a new segment address)

Example:

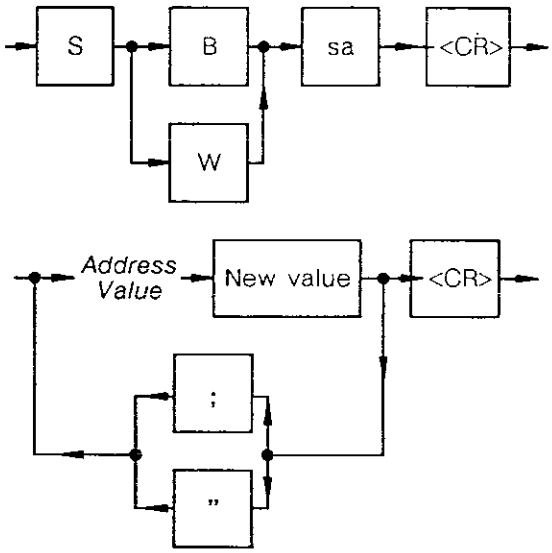
R <CR> The PLC is ready to receive an INTEL HEX FILE

R 2F00 <CR> The PLC is ready to receive an INTEL HEX file. The HEX value of $2F00_H$ is added to the segment address of the address extension record. The resulting new segment address is decisive as to the storage of the HEX file data.

Display/edit memory contents

The memory contents can be displayed and edited in bytes or words.

Command:

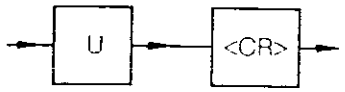


B: In bytes (key word)
W: In words (key word)
sa: Start address from which the memory contents are to be displayed/edited
Address: Address of the memory contents
Value: Value of the memory contents
New value: New value of the memory (user input)
;: When a semicolon is entered, the address is incremented by 1 (SB command) or by 2 (SW command)
↑: By entering circumflex (on PC ^), the address is decremented by 1 (SB command) or by 2 (SW command).

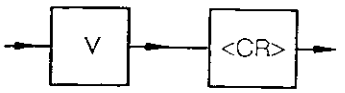
Altering the address output format

By initialization, the monitor is set to the address format of SEGMENT:OFFSET, which is used in every address output. The address output format can be chosen freely by using the user commands of U <CR> (Segment:Offset format) and V <CR> (absolute format).

Command:



E.g: address output 1000:00A0



E.g: address output 100A0

Write INTEL HEX file

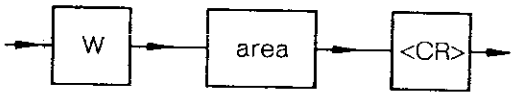
Using the W command, it is possible to output a data area of the PLC through serial interface 1 (bottom interface) of the PLC as an INTEL HEX file. The following records (see the chapter on serial interfaces) are generated:

- Address extension record
- Data record
- End record

The following transmission format applies:

- 8 data bits
- No parity bit
- 1 stop bit

Command:



area: Memory area to be output as an INTEL HEX file.

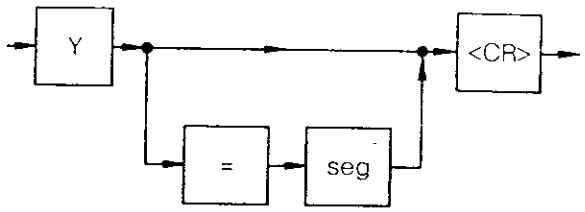
Example:

W 8000:0,FFFF <CR>	The memory area from $8000:0_H$ up to and including $8000:FFFF_H$ is output through serial interface 1 of the PLC as an INTEL HEX file.
W 8000:0LFFFF <CR>	The memory area from $8000:0_H$ up to and including $8000:FFFE_H$ is output through serial interface 1 of the PLC as an INTEL HEX file.

Display/edit work segment

If no segment is specified during address input and, instead of this, only an offset, the work segment of Y is used. The default of the work segment is zero.

Command:



seg: New segment address of the work segment
=: Key word

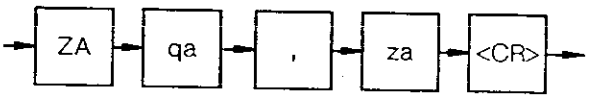
Example:

Y <CR>	User command: display work segment
Y 0000	Display
DB 0L2	Display memory contents in bytes
0000:0000 02 00	Display
Y=8000 <CR>	Edit work segment

Cyclical read and write

A value is read cyclically from a source address and is written to a target address. The operation can be aborted by pressing CTRL C.

Command:



qa: Source address from which the value is read
za: Target address to which the value is written
,: key word (delimiter)

Example:

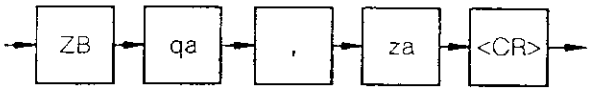
ZA 1000:0, 1000:100 <CR>

From address $1000:0_H$ a value is read cyclically and written to address $1000:100_H$.

Cyclic read and write with waiting time

A value is read cyclically from a source address and is written to a target address. The operation can be aborted by pressing CTRL C. The waiting time between two read cycles is approx. 1 ms.

Command:



qa: Source address from which the value is read
za: Target address to which the value is written
,: Key word (delimiter)

Example:

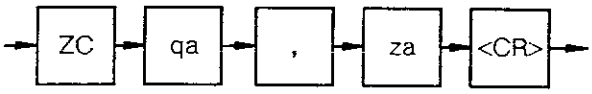
ZA 1000:0, 1000:100 <CR>

A value is read cyclically from the address of $1000:0_H$ and is written to the address of $1000:100_H$.

Read and write at key stroke

A value is read from a source address and is written to a target address after every key stroke. The operation can be aborted by pressing CTRL C.

Command:



qa: Source address from which the value is read
za: Target address to which the value is written
,: Key word (delimiter)

Example:

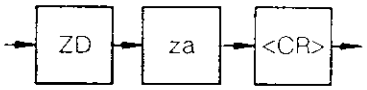
ZC 1000:0, 1000:100 <CR>

A value is read from the address of $1000:0_H$ and is written to the address of 1000:100H after every key stroke.

Cyclic write

The value of a counter is decremented and written to a target address. The operation can be aborted by pressing CTRL C.

Command:



za: Target address to which the value is written

Example:

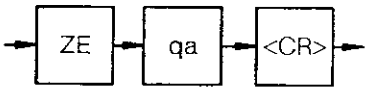
ZE 1000:100 <CR>

The value of a counter is written to the address of $1000:100_H$. The counter is decremented after every write operation.

Cyclic read

A source address is read cyclically. The operation can be aborted by pressing CTRL C.

Command:



qa: Source address from which the value is read

Example:

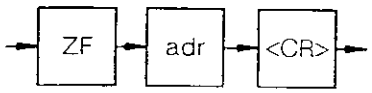
ZE 1000:100 <CR>

The value of address $1000:100_H$ is read cyclically.

Cyclic write and read

The value of a counter is written cyclically to an address and then read again. The operation can be aborted by pressing CTRL C.

Command:



adr: Address to which the value of the counter is written and from which the value is read

Example:

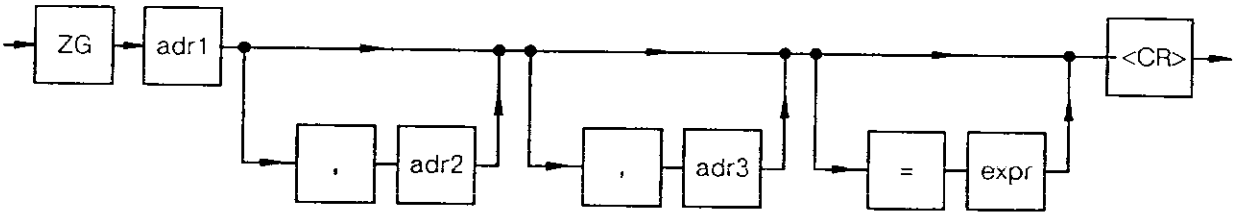
ZF 1000:0 <CR>

The value of a counter is written to the address of $1000:100_H$. After every write operation, the value at address $1000:100_H$ is read and the counter is decremented.

Simultaneous output of 3 values

Using the ZG command, it is possible to display the values of up to 3 addresses. The values displayed are updated whenever the value pertaining to the first address changes. The expression of "expr" specifies how often updating of the values is to be suppressed.

Command:



adr1: 1st address, the value of which is displayed on the screen. The values on the screen are updated if the value of adr 1 changes.
adr2: 2nd address, the value of which is displayed on the screen.
adr3: 3rd address, the value of which is displayed on the screen.
expr: Number of times updating of the values displayed is to be suppressed when the value of adr 1 changes.
,: Key word (delimiter)
=: Key word

Example:

ZG 1000:0, 1000:100 <CR>

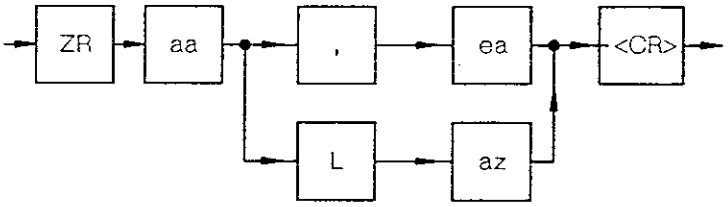
The values of the addresses of $1000:0_H$ and $1000:100_H$ are displayed on the screen. If the value of address $1000:0_H$ changes, the values of the two addresses displayed on the screen are updated.

RAM test

A test pattern (FFFF, 5555, AAAA) is written into the specified area and a check is then made as to whether the specified area contains the test values without any error having occurred. The address, the actual value and the setpoint are output if an error is discovered. The test can be continued by pressing any key (not <SPACE>) and can be aborted by pressing CTRL C.

3 test cycles are run through, during which the sequence of the test values is swapped. The 4th test cycle consists of storing a counter at the start address, of checking for faultless storage and of repeating the test with the decremented counter until it reaches zero. The RAM test is then concluded with the monitor message (*).

Command:



aa: Start address of the RAM area
ea: End address of the RAM area
L: Length (key word)
az: Quantity of bytes in the RAM area
,: Key word (delimiter)

Example:

ZR 1000:0L100 <CR> RAM test of the specified memory area.

Echo test of a serial interface

Using the ZT command it is possible to command an echo test of the interface specified in expr (*not the operator interface with the I/O address of* 100_H, see the chapter entitled "Serial interfaces"). The corresponding printable character is output each time a key is pressed on the connected periphery. The echo test can be aborted via the operator interface by pressing CTRL C.

Command:

expr: I/O address of the interfaces 1 to 3

Example: Interface 1 (I/O address 102):

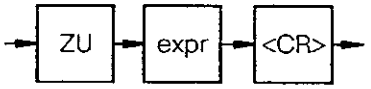
ZT 102 <CR>

Echo test of serial interface 1.

Short-circuit test of a serial interface

Using the ZU command it is possible to test the specified interface (not the operator interface with the I/O address of 100_H, see the chapter entitled "Serial interfaces"). To do this, pins 2 and 3 of the interface must be short-circuited. All 256 bit combinations are used. After 256 runs, a "greater than" character (>) is output through the operator interface. The test can be aborted by entering CTRL C through the operator interface.

Command:



expr: I/O address of interfaces 1 to 3

Example: Interface 1 (I/O address 102):

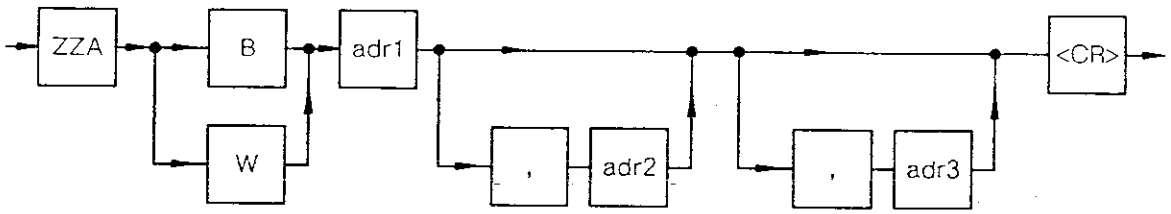
ZU 102 <CR>

Short-circuit test of serial interface 1.

Output of 3 values after input of a semicolon (;)

Using the ZZA command it is possible to display the values (byte or word) of up to 3 addresses after input of a semicolon (;). The command can be aborted by pressing <CR>.

Command:



B: Byte by byte (key word)
W: Word by word (key word)
adr1: 1st address, the value of which is displayed on the screen.
adr2: 2nd address, the value of which is displayed on the screen.
adr3: 3rd address, the value of which is displayed on the screen.
,: Key word (delimiter)

Example:

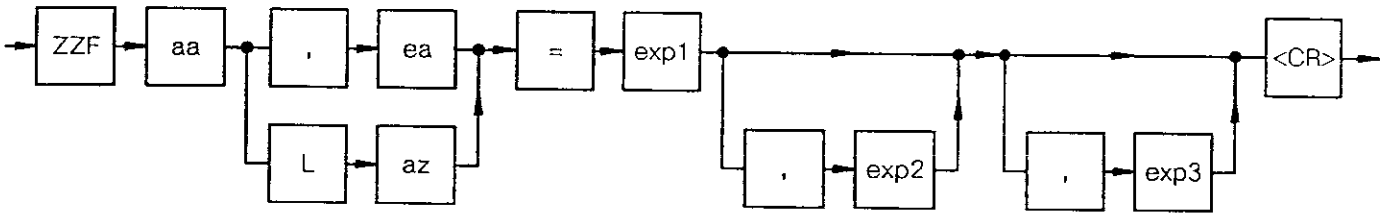
ZZA 1000:0, 1000:100 <CR>

After input of a semicolon (;) the values of the addresses of $1000:0_H$ and $1000:100_H$ are displayed on the screen.

Search for a string

Using the ZZF command you can search for a string with a maximum of 3 words in the specified memory area. If the string is found, its address is displayed. The search can be continued by entering a semicolon (;). The monitor message of <#07> is output if the string is not found.

Command:



aa: Start address of the memory area
ea: End address of the memory area
L: Length (key word)
az: Number of words in the memory area
exp1: 1st string word
exp2: 2nd string word
exp3: 3rd string word
,: Key word (delimiter)

Example:

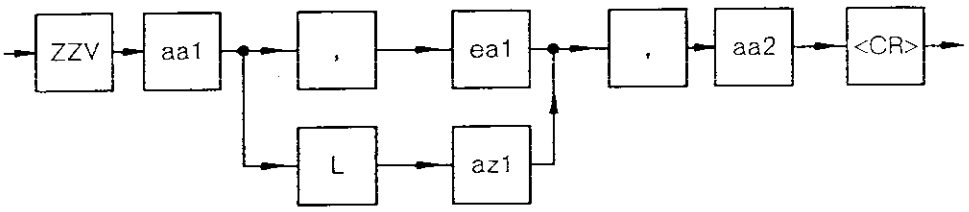
ZZF 1000:0, 100 = AAAA, BBBB <CR>

A search is made for the entered string ($AAAA_H$, $BBBB_H$) in the area from $1000:0_H$ to $1000:100_H$.

Compare memory areas word-by-word

Using the command ZZV a memory area 1 is compared word-by-word to a memory area 2. The address 1, contents 1, the address 2 and contents 2 are displayed if a difference is discovered. The operation can be aborted by pressing CTRL C.

Command:



aa1: Start address of the memory area 1
ea1: End address of the memory area 1
L: Length (key word)
az1: number of words in the memory area 1
aa2: number of words in the memory area 2
,: Key word (delimiter)

Example:

ZZV A000:0 L 100, 8000:0 <CR>

The memory area 1 between $A000:0_H$ and $A000:100_H$ is compared to memory area 2 as from $8000:0_H$.

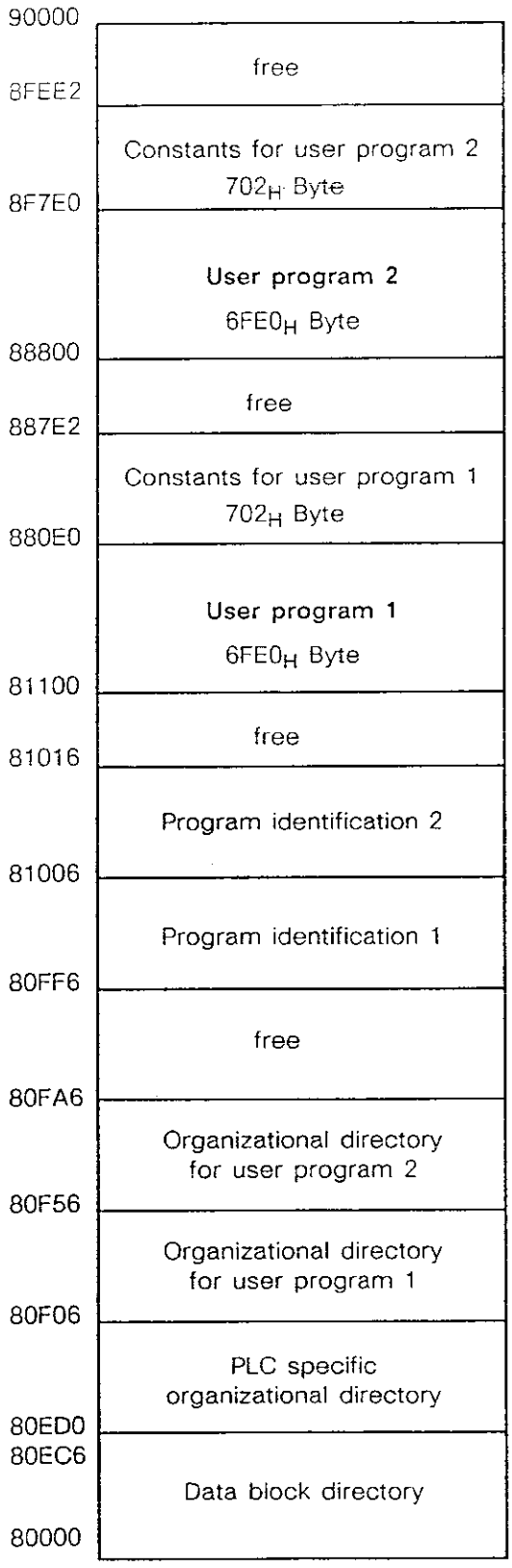
20 Memory subdivision

20.1 Summary of the PLC's complete address area

Address range	Socket number	Fitting	Function
0000_H:0000_H ... ... 0000_H:$FFFF_H$	A104/A105	RAM 43256	System RAM
1000_H:0000_H ... ... 1000_H:$FFFF_H$	A4/A24	RAM 43256	Transfer RAM
2000_H:0000_H ... ... 2000_H:$FFFF_H$			Passive stations
3000_H:0000_H ... ... 3000_H:$FFFF_H$			Active stations
4000_H:0000_H ... ... 4000_H:$FFFF_H$	A84/A85	RAM 43256	TURBO memory of user program 1
5000_H:0000_H ... ... 5000_H:$FFFF_H$	XA0/XA1	RAM 43256	TURBO memory of user program 1
6000_H:0000_H ... ... 6000_H:$FFFF_H$	XA2/XA3	RAM 43256	TURBO memory of user program 2
7000_H:0000_H ... ... 7000_H:$FFFF_H$	XA4/XA5	RAM 43256	TURBO memory of user program 2
8000_H:0000_H ... ... 8000_H:$FFFF_H$	XA6/XA7	RAM 43256	User program memory RAM
9000_H:0000_H ... ... 9000_H:$FFFF_H$	XA8/XA9	RAM 43256	Operand memory
$A000_H$:0000_H ... ... $A000_H$:$FFFF_H$	XA10/XA11	RAM 43256 or EPROM 27256	120 kB program memory size: program RAM 56 kB program memory size: program EPROM
$B000_H$:0000_H ... ... $B000_H$:$FFFF_H$	XA10/XA11	RAM 43256	applicable only if 56 kB progr. memory is configurated Comment menory of user program 2
$C000_H$:0000_H ... ... $C000_H$:$FFFF_H$	XA12/XA13	RAM 43256	applicable only if 56 kB progr. memory is configurated Comment memory of user program 2
$B000_H$:0000_H ... ... $C000_H$:$FFFF_H$	XA12/XA13	EPROM 27512	120 kB program memory size: program EPROM 56 kB program memory size: program EPROM
$D000_H$:0000_H ... ... $D000_H$:$FFFF_H$			not used
$E000_H$:0000_H ... ... $F000_H$:$FFFF_H$	A74/A75	EPROM 27512	System EPROM

20.2 Detailed summary of the user program memory

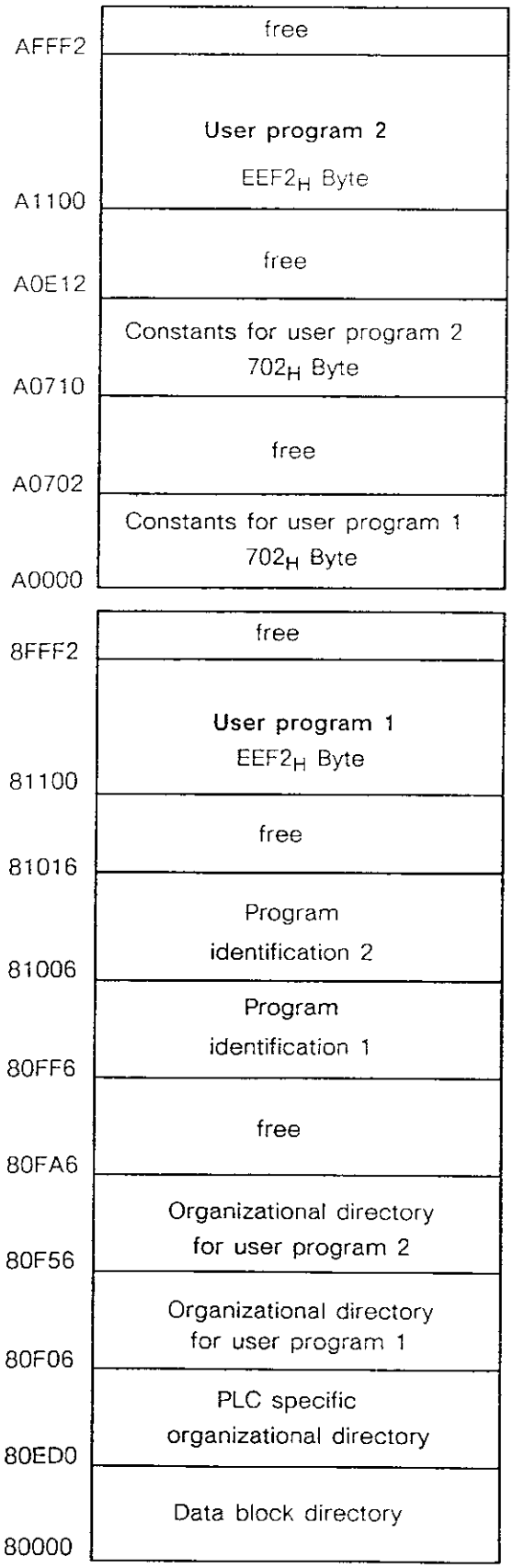
20.2.1 Subdivisions if the program memory size is 56 kbytes



Explanations:

- Data block directory; marshalling list between the I/O operands and the I/O modules
- Organizational directory
  - PLC-specific: Organizational data concerning the entire PLC is stored here
  - for user program 1: Organizational data concerning user program 1 is stored here
  - for user program 2: Organizational data concerning user program 2 is stored here
- Program identification 1: 16 bytes for the identifier for user program 1 (e.g. date, name of the creating programmer).
- Program identification 2: 16 bytes for the identifier for user program 2 (e.g. date, name of the creating programmer).
- User program 1: user program 1 is located in this area.
- Constants for user program 1: the indirect constants of user program 1 are located in this area.
- User program 2: user program 2 is located in this area.
- Constants for user program 2: the indirect constants of user program 2 are located in this area.

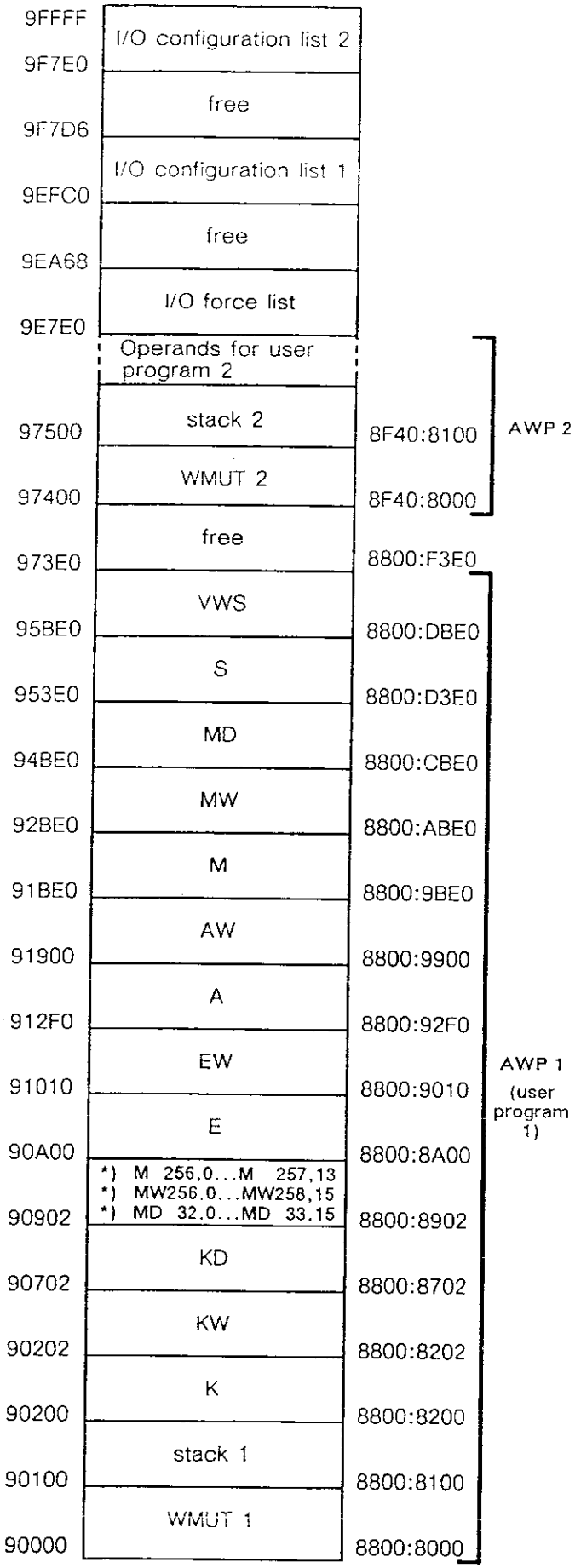
20.2.2 Subdivisions if the program memory size is 120 kbytes



Explanation:

- Data block directory: marshalling list between the I/O operands and the I/O modules.
- Organizational directory
  - PLC-specific: organizational data concerning the entire PLC is stored here.
  - for user program 1: organizational data concerning user program 1 is stored here.
  - for user program 2: organizational data concern ing user program 2 is stored here.
- Program identification 1: 16 bytes for the identifier for user program 1 (e.g. date, name of the creating programmer).
- Program identification 2: 16 bytes for the identifier for user program 2 (e.g. date, name of the creating programmer).
- User program 1: user program 1 is located in this area.
- Constants for user program 1: the indirect constants of user program 1 are located in this area.
- User program 2: user program 2 is located in this area.
- Constants for user program 2: the indirect constants of user program 2 are located in this area.

20.3 Detailed summary of the operand memory



Explanation:

WMUT 1: work memory for user program 1

Stack 1: stack for user program 1

K: indirect constant, BINARY

KW: indirect constant, WORD

KD: indirect constant, DOUBLE WORD

*): These flags are always remanent, i.e in INI OPS mode they are not affected by initialization

E: process image of the inputs, BINARY

EW: process image of the inputs, WORD

A: process image of the outputs, BINARY

AW: process image of the outputs, WORD

M: flag, BINARY

MW: flag, WORD

MD: flag, DOUBLE WORD

S: step chains

VWS: historical value memory

WMUT 2: work memory for user program 2

Stack 2: stack for user program 2

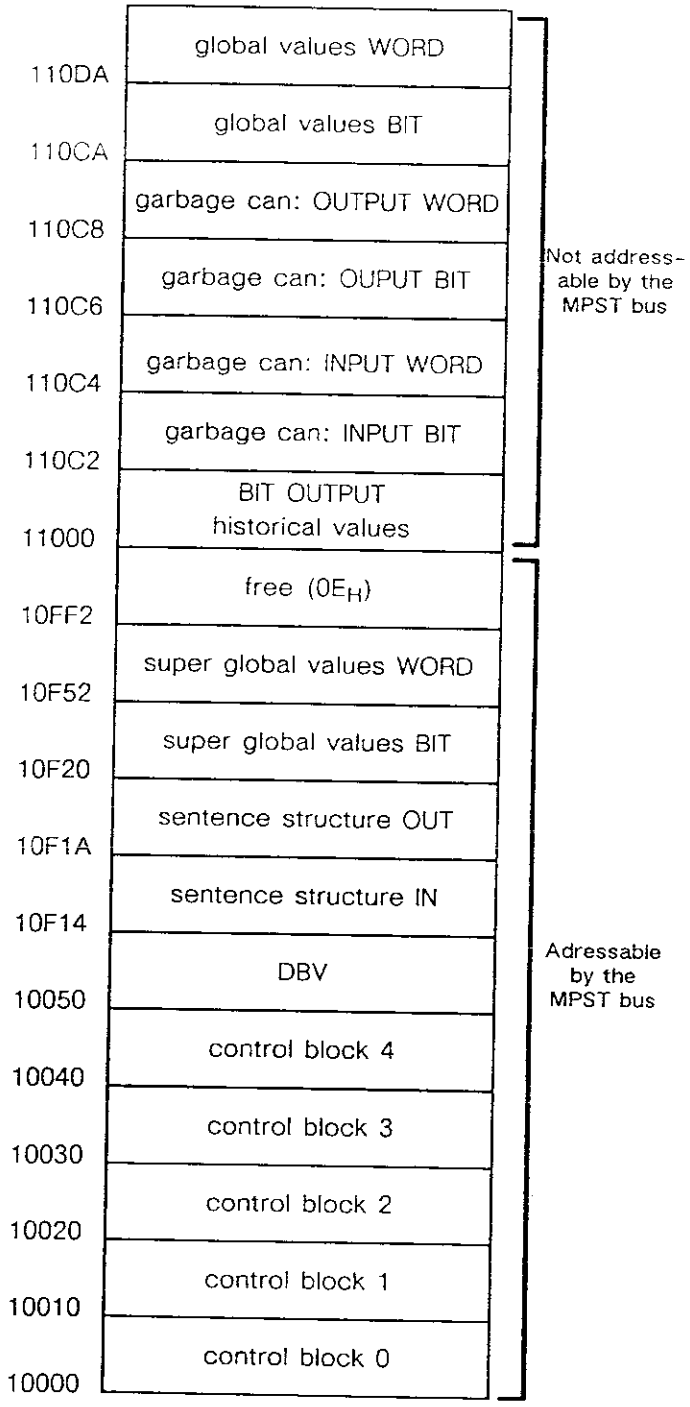
Operands for user program 2:
Process image for inputs and outputs
Flags and step chains for user program 2

I/O force lists:
The I/O signals to be forced and their force values are entered here.

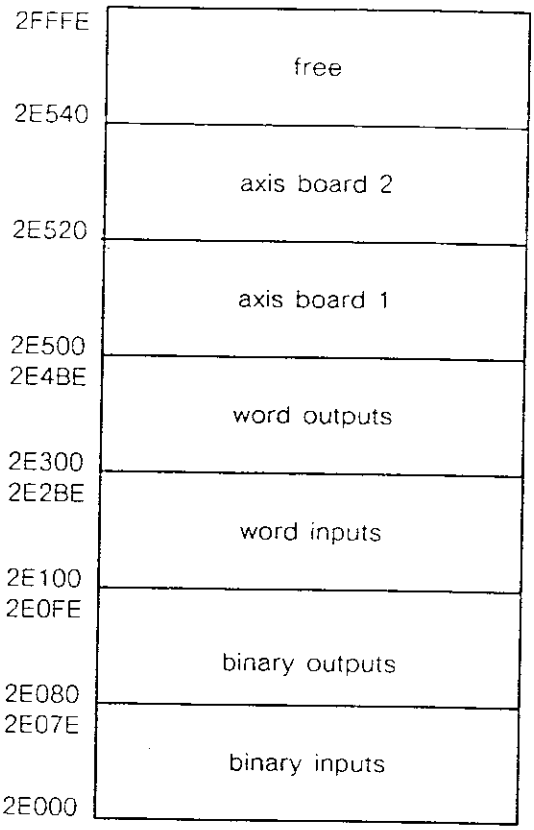
I/O configuration list 1:
The I/O signals occurring in program 1 are entered here so that they will be taken into account during the course of creation and output of the process image.

I/O configuration list 2:
The I/O signals occurring in program 2 are entered here so that they will be taken into account during the course of creation and output of the process image.

20.4 Detailed summary of the transfer RAM



20.5 Detailed summary of the standard peripheral addresses



21 Extension of old systems and upward compatibility

21.1 Extension of 935 PC 81 (V3.0) with 935 PC 83 (as of V5.0)

Old systems based on the modules

- 35 ZP 93 (GJR5 1332 00 R11) Processor card
- 35 DS 90 (GJR5 1333 00 R20) Data interface

and the software

- 935 PC 81 (Version 3.0)

can be extended with a PLC, based on the modules

- 35 ZP 93 (GJR5 1332 00 R31) Processor card
- 35 DS 91 (GJR5 1374 00 R1) Data interface or the compact modules
- 35 ZE 93 and 35 ZE 94

and the software

- 935 PC 83 (version 5.0 or upward)

On expansion, the following must be observed:

- Use of the operator control command $ (operation of several PLC processors via *one* serial interface) between the version V 3.0 and new PLC is *forbidden* because the data structures have different organizations in the transfer RAMs. If it is nevertheless used, a memory word is overwritten with an invalid value in the transfer RAM of the PLC selected by the command.

- The allocation between the I/O signals and the I/O module addresses differs between version V 3.0 and later versions.

- The names of superglobal values and their physical storage in the transfer RAMs of the version V 3.0 and later versions differ.

- The physical addresses of the I/O modules and superglobal values can, however, be remarshalled very easily in the PLC versions as of V 5.0 by the DBV command.

Communication between Version 3.0 and the versions as of 5.0 by way of superglobal values is thus easy to configure. In doing this, the version 5.0 or newer must always be operated as the SLAVE. In version 5.0 or newer the pointers for physical storage of the superglobal values are set in the new PLC to the corresponding memory cells in the version 3.0 transfer RAM.

21.2 Upward compatibility

All PLCs as from Version 5.0 are upwardly compatible. That is to say, user programs and user program EPROMs of one PLC version are also executable on all PLCs with a higher version number. The following points must be observed in this case:

- As of Version 6.0, the gain factor in the PI, PIDT1, DT1 and INTK blocks is specified as a percentage (see also block catalog).

- As of Version 6.0, a different affiliation between the analog and digital value ranges applies in the ANEIN block (see also block catalog).

- As of Version 7.0, the blocks PORD and POWR are no longer available for the single-axis positioning module 35 PO 90. They have been replaced by the POKANF, POKO and POKEND blocks.

- As of Version 7.0, the BASI and BASO blocks are also no longer available.

- As of Version 7.0, the ADAPT block has been equipped with three additional outputs.

Appendix: Catalog of Blocks

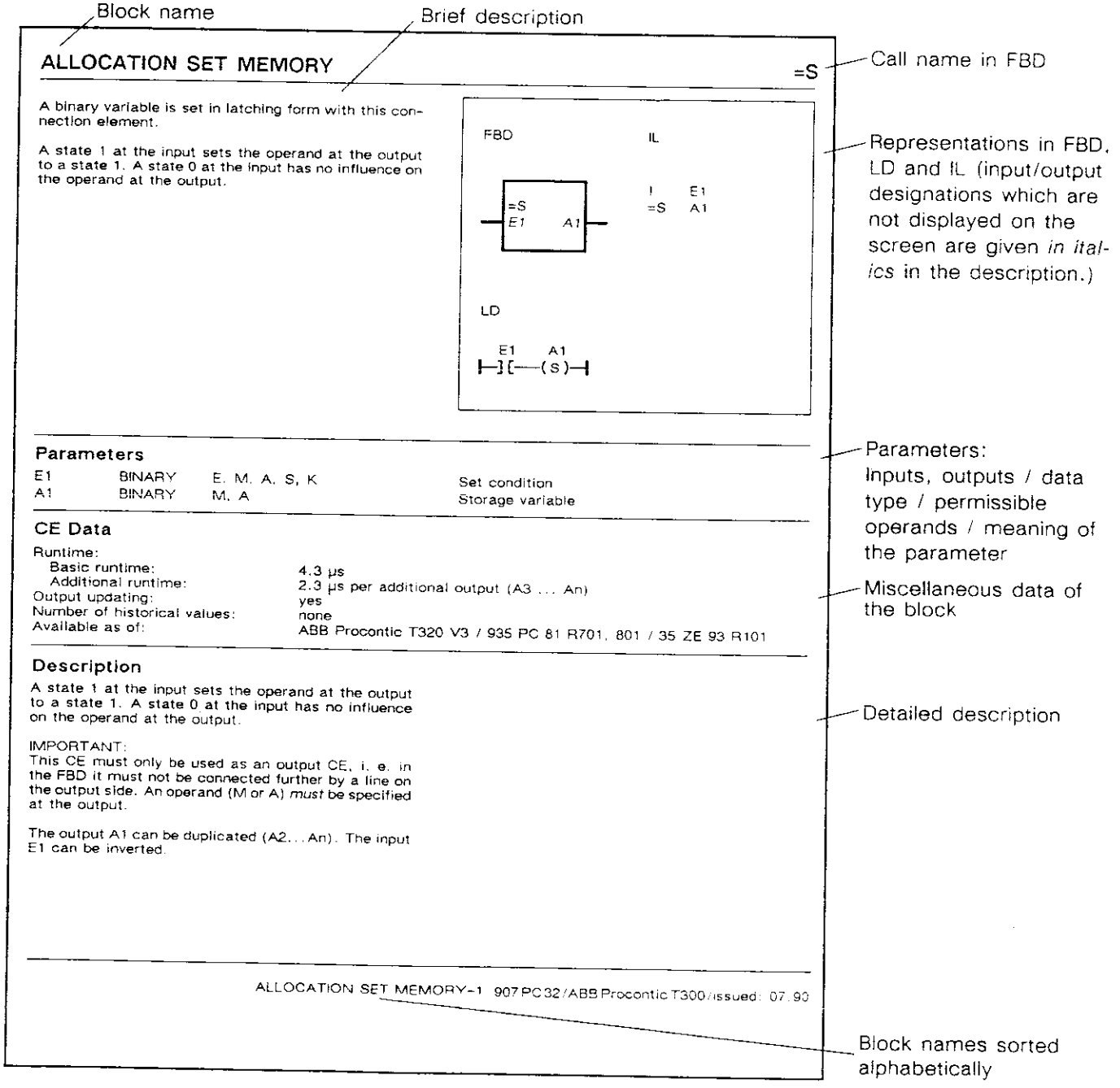
Notes

This catalog of blocks provides you first of all with two overviews of all blocks,

- one arranged according to function groups and
- the other arranged alphabetically according to the call names in FBD/LD and IL.

CAUTION!
In this catalog the blocks are sorted alphabetically according to their *block names*. They are not sorted according to the call name in FBD!

The description of each block is structured as follows:



Block name

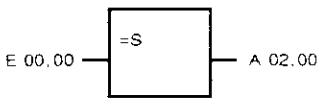
ALLOCATION SET MEMORY =S

Call name in FBD

Example

Example

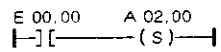
FBD



IL

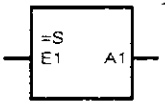
```
I    E 00.00
=S   A 02.00
```

LD



CE FBD Definition

Formal definition of CE



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	L	Y	P	N	0	0
A1	A	L	N	P	N	1	0

CE IL Definition

```
00000   I     PP 0   E1   Input BINARY
        [     1
00002   =S    PP 1   A1   Output BINARY (capable of duplication)
        ]     1
```

Operand comment (long text)
Symbolic operand
Formal operand
Operator
Occupied program memory word address

CE definitions
(These serve to define the graphical symbol and as a specification for translating the graphical symbol to the IL; see also 907 PC 32 description, General Part, Chapter 15 (library/CE editor/CE instruction editor)

907 PC 32/ABB Procontic T300/Issued: 07.90 ALLOCATION SET MEMORY-2

Further explanations

CE Data

Runtime:
Basic runtime: Runtime without duplication
Additional runtime: Runtime with duplication

Output updating
This specifies whether or not the outputs are re-allocated in each cycle. If "no", a direct connection to a line leading to further CEs is not possible. In this case, a flag must be assigned that is *not used multiply*.

CE FBD Definition

(See also 907 PC 32 description, General Part, Chapter 15 (library/CE editor/CE instruction editor))

Param.:
Information on the inputs and outputs (symbolic name).

Param. group:
Information on the parameter group (E = variable capable of interrogation, A =variable capable of allocation, S=special variable, K=constant, X=all).

Param. type:
Information on the data type (L=bit, B = byte, W = word, D = double word, F = floating, A = analog, T = text constant, Z = time constant, S = miscellaneous, X = all).

Inv.:
Information on whether or not the input/output can be inverted.

Allocation:
Only P allowed.

Displ. Screen:
Information on whether or not the parameter name is displayed in the CE on the screen.

Param. block:
Information on whether or not the input can be duplicated (input cannot be duplicated: 0; input can be duplicated: > 0).

Dupli. type:
Information on the type of duplication (single = 0, in blocks = 1).

Abreviations

You will find abbreviations used in Section 3, Page 0–1 of the binder entitled "907 PC 32, General Part".

Notes on the extended IL

In the extended IL, all PLC functions can be called by way of a selection menu. This menu is the same as in the case of the FBD/LD. When a function is called, a mask appears on the screen in which all inputs and outputs have the same designations as in the FDB/LD. The user then plans the operands he requires at the inputs and outputs of the mask.

Overview of blocks arranged according to function groups

Function group / Block name	Call name in FBD/LD, ext. IL / IL [1]
Binary functions	
AND	& [2]
OR	/ [2]
Exclusive OR	=1 [2]
Majority	MAJ
Allocation	= [2]
Latch functions	
Allocation set memory	=S [2]
Allocation reset memory	=R [2]
Set memory, dominating	RS [2]
Reset memory, dominating	SR [2]
Arithmetic functions, word	
Addition	+ [2]
Subtraction	- [2]
Multiplication	* [2]
Division	: [2]
Multiplication with division	*: / MULDI
Multiplication by 2 to the power of N	MUL2N
Absolute value generator	BETR
Allocation word	=W [2]
Alloc. dir. const. to word var.	ZUDKW
Square root	SQRT
Arithmetic functions, double word	
Addition double word	+D / ADDD
Subtraction, double word	-D / SUBD
Multiplication, double word	*D / MULD
Division, double word	:D / DIVD
Double word multiplication by 2 to the power of N	MUL2ND
Negation, double word	NEGD
Absolute value generator, double word	BETRD
Allocation, double word	=D / ZUWD
Square root	SQRT
Comparison functions	
Greater than	> [2]
Greater than or equal to	>= [2]
Equal	=? [2]
Unequal	>< [2]
	<> [2]

Function group / Block name	Call name in FBD/LD, ext. IL / IL [1]
Comparison functions	
Less than or equal to	<= [2]
Less than	< [2]
Comparator with 3-point response	VGL3P
Comparator with unilateral hyst.	VGLEH
Comp. with asymmetrical hyst.	VGLUH
Comparison functions, double word	
Greater than, double word	>D / VGRD
Less than, double word	<D / VKLD
Equal, double word	=?D / VGLD
Timer functions	
On delay	ESV
Off delay	ASV
Monostable element "abort"	MOA
Monostable elem. "constant"	MOK
Variable delay element	VVZ
Counter functions	
Up, down counter	VRZ
Up, down counter, double word	VRZD
Program control functions	
Conditional jump to label	SPBM
Target label	MRK
Program end	PE
Conditional program end	=PE [2]
Soubroutine call for an assembler program	CALLUP
Run number block	LZB
System call	SYSTEM
Analog value processing	
Read analog val. from 35EA90	A / ANAEIN
Format conversion	
BCD to binary conversion	BCDDUAL / BCDBIN
Binary to BCD conversion	DUALBCD / BINBCD
Pack binary variables in word	PACK
Unpacking a word into binary variables	UNPACK
Word to double w. conversion	WDW
Double w. to word conversion	DWW

[1] If there exists another call for IL than for FBD/LD, it is given additionally and separated by /.

[2] This function is realized in IL by a sequence of commands and/or blocks.

Function group / Block name	Call name in FBD/LD, ext. IL / IL [1]
Format conversion	
BCD to binary conversion, double word	BCDDUALD/BCDDW
Binary to BCD conversion, double word	DUALBCDD/DWBCD
Pack binary variables in double word	PACKD
Unpacking a double word into binary variables	UNPACKD/UNPAD
Pulse	
Pulse (positive edge)	I+ [2]
Pulse (negative edge)	I- [2]
Logic functions with word values	
AND combination, word	WAND
OR combination, word	WOR
Exclusive OR combination, word	WXOR
Mask	MASKE
Shift block	SHIFT
Logic functions with double word values	
AND combination, double word	DWAND
OR combination, double word	DWOR
Excl. OR combination, double w.	DWXOR
Mask, double word	MASKED
Shift block	SHIFT
Access to physical addresses	
Read word with enabling	WOL
Write word with enabling	WOS
Write word in the event of value change	WAES
Copying memory areas	COPY
Double word access to physical addresses	
Read double w. with enabling	DWOL
Write double w. with enabling	DWOS
Write double word in the event of value change	DWAES
Access to physical addresses (I/O ports)	
Read byte val. from I/O address	IOR
Write byte val. to I/O address	IOW
Higher order functions	
Word decoder	WDEC
Word recoder	WUMC
Double word recoder	DWUMC

Function group / Block name	Call name in FBD/LD, ext. IL / IL [1]
Higher order functions	
Binary selection gate	AWTB
Selection gate, word	AWT
Selection gate, double word	AWTD
Switchover gate	UST
Switchover gate with reset	USTR
Switchover gate, double word	USTD
Switchover gate with reset, double word	USTRD
Maximum value generator	MAX
Max. val. generator, double w.	MAXD
Minimum value generator	MIN
Min. val. generator, double w.	MIND
Max. val. gen. as a fct. of time	MAZ
Maximum value generator as a function of time, double w.	MAZD
Limiter	BEG
Limiter, double word	BEGD
Ramp-function generator	HLG
Function generator	FKG
Stack, Last in/First out	LIFO
Stack, First in/First out	FIFO
Error searcher with auto. del.	FEHSU
Error searcher with storage	SFEHSU
List allocator	LIZU
Address selection	ADRWA
Selection multiplexer	AWM
Switchover multiplexer	USM
Analog val. change annunciator	AMELD
Analog value change annunciator double word	AMELDD
Bin. value change annunciator	BMELD
Bit searcher	BITSU
Demultiplexer	DMUX
Demultiplexer double word	DMUXD
Multiplexer with reset	MUXR
Multiplex., double w. with reset	MUXRD
Read bin. variable, indexed	IDLB
Read word variable, indexed	IDLm / IDL
Write bin. variable, indexed	IDSB
Write word variable, indexed	IDSm / IDS
Illumination pushbutton control	LDT
Clock	UHR

[1] If there exists another call for IL than for FBD/LD, it is given additionally and separated by /.

[2] This function is realized in IL by a sequence of commands and/or blocks.

Function group / Block name	Call name in FBD/LD, ext. IL / IL [1]
Higher order functions	
Initialize memory area in the operand memory with zero	INiTS
Initialize variables	INITV
Automatic control engineering functions	
Differentiator with delay of the 1st order	DT1
Integrator (extended)	INTK
Proportional-integral-controller	PI
PT1 element	PT1
PIDT1 controller	PIDT1
Pulse duration modulator	PDM
Adaption for adaptive temperature control	ADAPT [3]
Communication via serial interface	
Initialization and configuration of the serial interfaces	SIN/SINIT
Initialization and configuration of the serial interfaces	SINIT
Output of ASCII characters and HEX values through a serial interface	DRUCK
Reception of characters	EMASm / EMAS
Additional comparison telegrams for EMASm	EMASmVT [4]
Historical values	
Read binary values from historical values memory	RDB
Write binary values into historical values memory	WRB
Read word values from historical values memory	RDW
Write word values to historical values memory	WRW
Read double word values from historical values memory	RDDW
Write double word values into historical values memory	WRDW

Function group / Block name	Call name in FBD/LD, ext. IL / IL [1]
Functions for axis control	
PO90 communication start	POKANF
PO90 communication end	POKEND
PO90 communication	POKO
Axis positioning	POS92L
Axis positioning	POS92S
Communication with video sensor OMS-F 35 IV 90	
Clear all measurement windows	CLWIN
Area center of gravity calculation	COGRA
Generation of a window	GENWI
Modify and load the measurement window frames	MODWI
Definition of histogram steps	PROHIS
Loading the result counters with offset values	PROOFF
Loading measurement window frames	PROWI
Read counter results	READC
Set binary video threshold	SETBIN
Set image processing mode	SETMOD
Conversion of pixels to real units of measure	UMIMA
Configuration	
Configuration 35 CS 91	KONFCS91
Special functions	
If then	not possible/ [2]
If then, word	not possible/ [2]
Notbit	not possible/ [2]

[1] If there exists another call for IL than for FBD/LD, it is given additionally and separated by /.

[2] This function is realized in IL by a sequence of commands and/or blocks.

[3] Call only possible in IL.

[4] Call not possible in IL.

Overview of blocks arranged alphabetically according to call names

Call name	Block name
&	AND
*	Multiplication
*:	Multiplication with division
*D	Multiplication, double word
+	Addition
+D	Addition double word
-	Subtraction
-D	Subtraction, double word
/	OR
:	Division
:D	Division, double word
<	Less than
<=	Less than or equal
<D	Less than, double word
=	Allocation
=1	Exklusive OR
=?	Equal
=?D	Equal, double word
=D	Allocation, double word
=PE	Conditional program end
=R	Allocation reset memory
=S	Allocation set memory
=W	Allocation word
>	Greater than
<>	Unequal
><	Unequal
>=	Greater than or equal to
>D	Greater than, double word
A	Read analog value from 35 EA 90
ADAPT	Adaption for adapt. temperature contr.
ADDD	Addition double word
ADRWA	Address selection
AMELD	Analog value change annunciator
AMELDD	Analog val. change annunc., double w.
ANAEIN	Read analog value from 35 EA 90
ASV	Off delay
AWM	Selection multiplexer
AWT	Selection gate, word
AWTB	Binary selection gate
AWTD	Selection gate, double word
BCDBIN	BCD to binary conversion

Call name	Block name
BCDDUAL	BCD to binary conversion
BCDDUALD	BCD to bin. conversion, double word
BCDDW	BCD to bin. conversion, double word
BEG	Limiter
BEGD	Limiter, double word
BETR	Absolute value generator
BETRD	Absolute val. generator, double word
BINBCD	Binary to BCD conversion
BITSU	Bit searcher
BMELD	Binary value change annunciator
CALLUP	Subroutine call for an assembler progr.
CLWIN	Clear all measurement windows
COGRA	Area center of gravity calculation
COPY	Copying memory areas
DIVD	Division, double word
DMUX	Demultiplexer
DMUXD	Demultiplexer, double word
DRUCK	Output of ASCII characters and HEX values through a serial interface
DT1	Differentiator with delay of the 1st order
DUALBCD	Binary to BCD conversion
DUALBCDD	Binary to BCD conversion, double word
DWAES	Write double word in the event of value change
DWAND	AND combination, double word
DWBCD	Binary to BCD conversion, double word
DWOL	Read double word with enable
DWOR	OR combination, double word
DWOS	Write double word with enabling
DWUMC	Double word recoder
DWW	Double word to word conversion
DWXOR	Exklusive OR combination, double word
EMAS	Reception of characters
EMASm	Reception of characters
EMASmVT	Additional comparison telegrams for EMASm
ESV	On delay
FEHSU	Error searcher with automatic deletion
FIFO	Stack, First in/First out
FKG	Function generator

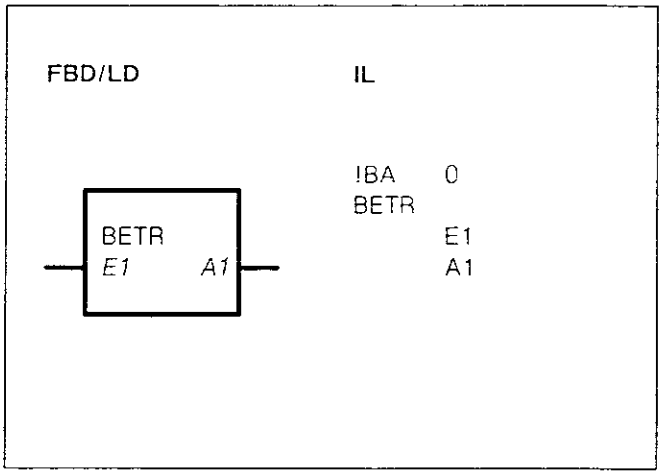
Call name	Block name
GENWI	Generation of a window
HLG	Ramp function generator
I+	Pulse (positive edge)
I-	Pulse (negative edge)
IDL	Read word variable, indexed
IDLB	Read bin. variable, indexed
IDLm	Read word variable, indexed
IDS	Write word variable, indexed
IDSB	Write bin. variable, indexed
IDSm	Write word variable, indexed
INITS	Initialize memory area in the operand memory with zero
INITV	Initialize variables
INTK	Integrator (extended)
IOR	Read byte value from I/O address
IOW	Write byte value to I/O address
KONFCS91	Configuration 35 CS 91
LDT	Illumination pushbutton control
LIFO	Stack, Last in/First out
LIZU	List allocator
LZB	Run number block
MAJ	Majority
MASKE	Mask
MASKED	Mask, double word
MAX	Maximum value generator
MAXD	Maximum value generator, double w.
MAZ	Max. val. gen. as a function of time
MAZD	Maximum value generator as a function of time, double word
MIN	Minimum value generator
MIND	Minimum value generator, double w.
MOA	Monostable element "abort"
MODWI	Modify and load the measurement window frames
MOK	Monostable element "constant"
MRK	Target lable
MUL2N	Multiplication by 2 to the power of N
MUL2ND	Double w. mult. by 2 to the power of N
MULD	Multiplication, double word
MULDI	Multiplication with division
MUXR	Multiplexer with reset
MUXRD	Multiplexer, double word with reset
NEGD	Negation, double word
PACK	Pack bin. variables in word

Call name	Block name
PACKD	Pack bin. variables in double word
PDM	Pulse duration modulator
PE	Program end
PI	Proportional-integral-controller
PIDT1	PIDT1 contoller
POKANF	PO90 communication start
POKEND	PO90 communication end
POKO	PO90 communication
POS92L	Axis positioning
POS92S	Axis positioning
PROHIS	Definition of histogram steps
PROOFF	Loading the result counters with offset values
PROWI	Loading measurement window frames
PT1	PT1 element
READC	Read counter results
RDB	Read binary values from historical values memory
RDDW	Read double word values from historical values memory
RDW	Read word values from historical values memory
RS	Set memory, dominating
SETBIN	Set binary video threshold
SETMOD	Set image processing mode
SFEHSU	Error searcher with storage
SHIFT	Shift block
SIN	Initialization and configuration of the serial interfaces
SINIT	Initialization and configuration of the serial interfaces
SPBM	Conditional jump to label
SQRT	Square root
SR	Reset memory, dominating
SUBD	Subtraction, double word
SYSTEM	System call
UHR	Clock
UMIMA	Conversion of pixels to real units of measure
UNPACK	Unpacking a word into bin. variables
UNPACKD	Unpacking a double word into bin. var.
UNPAD	Unpacking a double word into bin. var.
USM	Switchover multiplexer

Call name	Block name
UST	Switchover gate
USTD	Switchover gate, double word
USTR	Switchover gate with reset
USTRD	Switchover gate with reset, double w.
VGL3P	Comparator with 3-point response
VGLEH	Comparator with unilateral hysteresis
VGLUH	Comparator with asymmetrical hyst.
VGLD	Equal, double word
VGRD	Greater than, double word
VKLD	Less than, double word
VRZ	Up, down counter
VRZD	Up, down counter, double word
VVZ	Variable delay element
WAES	Write word in the event of val. change
WAND	AND combinatin, word
WDEC	Word decoder
WDW	Word to double word conversion
WOL	Read word with enabling
WOR	OR combination, word
WOS	Write word with enabling
WRB	Write binary values into historical values memory
WRDW	Write double word values into historical values memory
WRW	Write word values to historical values memory
WUMC	Word recoder
WXOR	Exclusive OR combination, word
ZUDKW	Allocation direct const. to word var.
ZUWD	Allocation, double word

ABSOLUTE VALUE GENERATOR — BETR

The absolute value of the word operand at the input E1 is generated and is allocated to the operand at the output A1.



Parameters

E1	WORD	EW, MW, AW, KW	Input value
A1	WORD	MW, AW	Absolute value of the input value

CE Data

Runtime:	
Basic runtime:	28 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, 801 / 35 ZE 93 R101

Description

The absolute value of the word operand at the input E1 is generated and is allocated to the operand at the output A1.

The input and the output can neither be duplicated nor negated.

Number range

Integer word (16 bits).

The following applies here particularly to the input E1:

If the inadmissible value 8000 H (-32768) is present at the input E1, the maximum possible value 7FFF H (+32767) is allocated to the output A1.

The following generally applies:

- low limit: 8001 H -32767
- high limit: 7FFF H +32767
- inadmissible value: 8000 H ---

In twos complement arithmetic, the value 8000 H (-32768) lies *outside* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations on the part of the user or
- by reading in from outside the PLC or
- by an indirect word constant

under no circumstances may a negation or subtraction be done on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the forbidden value 8000 H (-32768) is corrected to the allowed value 8001 H (-32767).

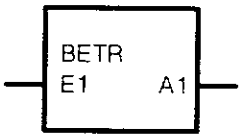
Example

FBD/LD

IL

```
!BA    0
BETR
       EW 00,00
       AW 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
A1	A	W	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	BETR			
00002		PP 0	E1	Input WORD
00003		PP 0	A1	Output WORD

ABSOLUTE VALUE GENERATOR, DOUBLE WORD — BETRD

The absolute value of the operand at the input E1 is generated and the result is allocated to the operand at the output A1.

FBD/LD

IL

```
!BA   0
BETRD
      E1
      A1
```

Parameters

E1	DOUBLE WORD	MD, KD	Input value
A1	DOUBLE WORD	MD	Absolute value of the input value

CE Data

Runtime:	
Basic runtime:	32 ... 36 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The absolute value of the operand at the input E1 is generated and the result is allocated to the operand at the output A1.

If, for any particular reason, the invalid value 8000 0000 H (-2 147 483 648) is present at the input E1, the value 7FFF FFFF H (+2 147 483 697) is allocated to the output A1. Therefore, the invalid value is 8000 0000 H is first of all corrected to the valid value 8000 0001 H and only then is the absolute value generated.

The input and the output can neither be duplicated nor negated.

Number range

Integer double word (32 bits)

• low limit:	8000 0001 H	-2 147 483 647
• high limit:	7FFF FFFF H	+2 147 483 647
• invalid value:	8000 0000 H	---

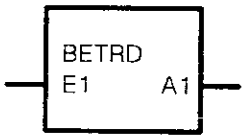
Example

FBD/LD

IL

```
!BA    0
BETRD
       MD 00,00
       MD 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	BETRD			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	A1	Output DOUBLE WORD

ADAPTION FOR ADAPTIVE TEMPERATURE CONTROL

Call in FBD/LD not defined

The adaption block serves to determine optimized controller parameters for the controller block PIDT1. This block can be used for temperature controlled systems in process engineering and in the plastics-processing industry.

The adaption block does not serve to continuously determine the controller parameters and only determines them whenever required (pushbutton adaption).

FBD/LD	IL	
not defined *)	!BA	0
	ADAPT	
		W
		X
		YR
		START
		READY
		ERROR
		Y
		INIT
		KP
		TN/TZ
		TV/TZ
		T1/TZ
		D-FREI
		KS
		TU
		TG

*) Note: A CE can be defined for the adaption block. See 907 PC 32 description, general part, Chapter 15 (library/CE editor/CE instruction editor).

Parameters

W	WORD	EW, AW, MW, KW	Command variable (setpoint)
X	WORD	EW, AW, MW, KW	Controlled variable (actual value)
YR	WORD	EW, AW, MW, KW	Manipulated variable of the controller at the starting time of adaption
START	BINARY	E, A, M, S	Start of adaption
READY	BINARY	E, A, M	Adaption completed message
ERROR	WORD	AW, MW	Error messages
Y	BINARY	A, M	Excitation signal (manipulated variable) during adaption
INIT	WORD	AW, MW	Initial value for the controller
KP	WORD	AW, MW	Proportional coefficient
TN/TZ	WORD	AW, MW	Integral action time, scaled to PLC cycle time TZ
TV/TZ	WORD	AW, MW	Derivative action time, scaled to PLC cycle time TZ
T1/TZ	WORD	AW, MW	Returning time, scaled to PLC cycle time TZ
D-FREI	BINARY	A, M	Enable for the controller's D component
KS	WORD	AW, MW	Gain of the controlled system
TU	WORD	AW, MW	Delay time of the controlled system
TG	WORD	AW, MW	Stabilization time of the controlled system

CE Data

Runtime:	
Basic runtime:	not available
Additional runtime:	not available
Output updating:	no
Number of historical values:	70 Words
Available as of:	ABB Procontic T300 V6 / 935 PC 83 R301 / 35 ZE 93 R101

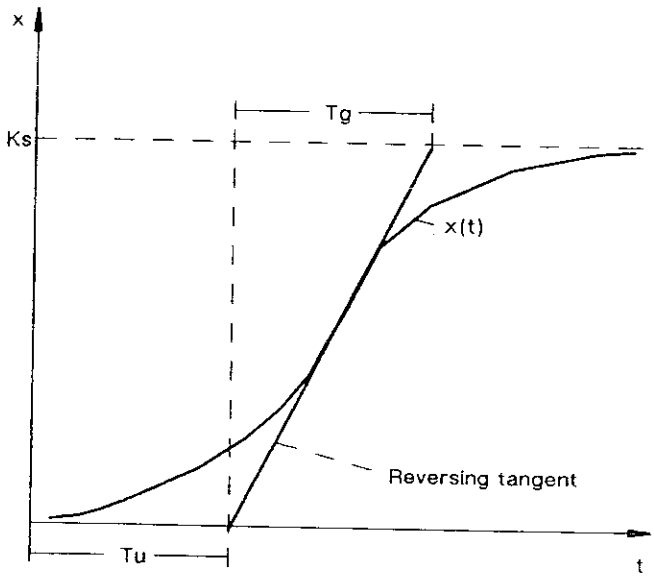
Description

Note: Modified in comparison with PLC version V6.0. As from PLC version V7.0, the controlled system parameters Ks, Tu and Tg are available as additional outputs.

The adaption block serves to determine optimized controller parameters for the controller block PIDT1. This block can be used for temperature controlled systems in process engineering and in the plastics-processing industry. The dynamics of these thermal processes can be approximated adequately by means of the characteristic quantities

Tu: Delay time
Tg: Stabilization time
Ks: Transfer factor (controlled system gain)

which can be defined on the basis of the transfer function. In this process, the actual progression of the transfer function is replaced by the tangent placed at its reversing point.



The above figure shows the transfer function, its reversing tangent and the resulting characteristic quantities in aperiodic processes.

The adaption block is designed especially for processes whose characteristic quantities fulfill the following conditions:

$5 \leq Tg/Tu \leq 100$
$20\ sec \leq Tu \leq 300\ sec$
$0.25 \leq Ks \leq 2$

Adaption is only ever permissible when the control loop has stabilized. Adaption can be realized both when starting up the process (initial setting) and also whenever the command variable changes. This enables adaption of the controller parameters if the process has to be run at various operating points and a fixed setting of the controller parameters does not result in the required control response. In this case, an actuation degree change (command variable change/gain of the controlled system) of at least 20% is presumed. The identification of the controlled system leads to inexact controller parameters in the event of actuation degree changes below 20% which result in larger-scale overshoot.

During adaption, the adaption block controls the process and the controller is isolated from it.

Starting/aborting adaption

Adaption begins by a 0/1 edge at the START input. A running adaption is aborted by means of a 1/0 edge at the START input.

Carrying out Adaption

During adaption, the output Y supplies the binary manipulated variable for the controlled system. The PIDT1 controller is inactive, i.e. the controller must not influence the system. This must be ensured by a control logic in the PLC program.

At the start of adaption the block requires the initial values of the currently available controlled and manipulated variables. These initial values are only known exactly when set for the first time (controlled variable = ambient temperature, manipulated variable = 0).

In the event of adaption in controlled mode (resetting in the event of a setpoint change), the initial value of the manipulated variable is defined by the actuation output of the PIDT1 controller. Owing to the planned control structure (PIDT1 followed by a PDM block), the PIDT1 controller does not supply a stationary manipulated variable. Continuous oscillation of the controller's manipulated variable sets in, preventing exact measurement of the constant component. Low pass filtering of the controller's manipulated variable is therefore recommended in order to provide the initial value of the manipulated variable for the adaption block (PT1 block, time constant > 10 s).

As the result of continuous oscillation of the controller's manipulated variable, the controlled variable also suffers continuous oscillation. However, this oscillation is attenuated very substantially by the low pass response of the controlled system. Additional filtering of the controlled variable is therefore not necessary.

Monitoring adaption

The adaption block contains a monitoring level. This performs a plausibility check on the identified controlled system parameters and the input variables. Correct termination or discrepancies detected during adaption are reported through the ERROR output. In this case, a distinction is made between errors and warnings.

Errors:

These always lead to premature aborting of adaption. No controller parameters are computed and the actuation output Y is set to "0". The READY signal remains set to "0".

Warnings:

These indicate that the computed controlled system parameters lie outside of the permissible value range. However, adaption is still terminated in the regular fashion (READY = 1). Nevertheless, a satisfactory control response cannot be guaranteed. The control response must be checked for its usefulness during commissioning.

- ERROR = 0; no error, no warning.

- ERROR = 1: error
At the start of adaption, the controlled variable difference is less than the minimum value MINXD = 3843. Adaption is aborted.
Remedy: Increase the initial controlled deviation.

- ERROR = 2: error
Less than 16 interpolation points were available for calculation of the reversing tangent. Adaption is aborted.
Remedy: Reduce the cycle time.

- ERROR = 3: error
The delay time TU lies outside of the permissible value range ($TU \leq 0$). Adaption is aborted.
Remedy: Consult ABB.

- ERROR = -1: warning
The actuation degree change lies below the required minimum value. Adaption is terminated in the regular fashion. However, a satisfactory control response cannot be guaranteed.
Remedy: Check the control response on commissioning. Consult ABB if this is unsatisfactory.

- ERROR = -2: warning
The ratio TU/TG is greater than the permissible maximum value. Adaption is terminated in the regular fashion. However, a satisfactory control response cannot be guaranteed.
Remedy: Check the control response on commissioning. Consult ABB if this is unsatisfactory.

Termination of adaption

READY = 1 indicates that adaption has been terminated in the regular fashion.

The following controller parameters are then present at the block's outputs

- INIT
- KP
- TN/TZ
- TV/TZ
- T1/TZ
- D-FREI

and the controlled system parameters

- KS
- TU
- TG

The controlled system parameters KS, TU and TG are only of significance for the specialist and normally do not need to be evaluated by the user.

The D-FREI output indicates whether or not a PI or a PIDT1 controller is necessary.

The following applies:

D-FREI	Meaning
0	DT1 component is deactivated -> PI controller
1	DT1 component is connected -> PIDT1-controller

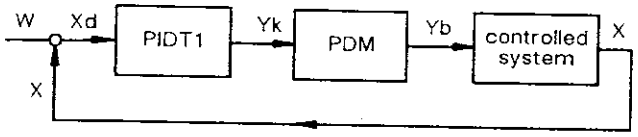
An adjustment value for switchover from the adaption block to the controller is available at the INIT output.

The Y output is set to "0" after termination of adaption.

PLANNING

Temperature control with switching actuators

In ABB Procontic T300, the following controller structure is used for temperature control with switching actuators:



W: Command variable (setpoint)
X: Controlled variable (actual value)
Xd: Controlled deviation
Yk: Continuous manipulated variable
Yb: Binary manipulated variable

As the result of the pulse width modulator (PDM), residual oscillation of the controlled variable at the operating point occurs in contrast to a continuous actuator. This residual oscillation is defined by the cycle ratio TA/TZ of the PDM block. In this case, TA is the period and TZ the cycle time with which the PDM block is processed.

Recommended settings:

TZ (PDM) = 100 ms: cycle time of the PDM
TA/TZ (PDM) = 5: cycle ratio of the PDM
TZ(PIDT1) = 500 ms: cycle time of the PIDT1

Configuration of controller and adaption block

A few elementary points must be observed in respect of planning:

- The following blocks must be run with a cycle time which is at least 5 times lower than that of the controller and adaption block
  - Pulse width modulator (PDM)
  - Selection gate (AWT) for the actuating signals of the controller and adaption block.

  This is achieved by virtue of the fact that all blocks and instructions belonging to adaptive control, with the exception of the pulse width modulator (PDM) and the selection gate (AWT) are processed within one running number block (LZB) with a running number equal to 5 and are therefore only processed in every 5th program cycle.

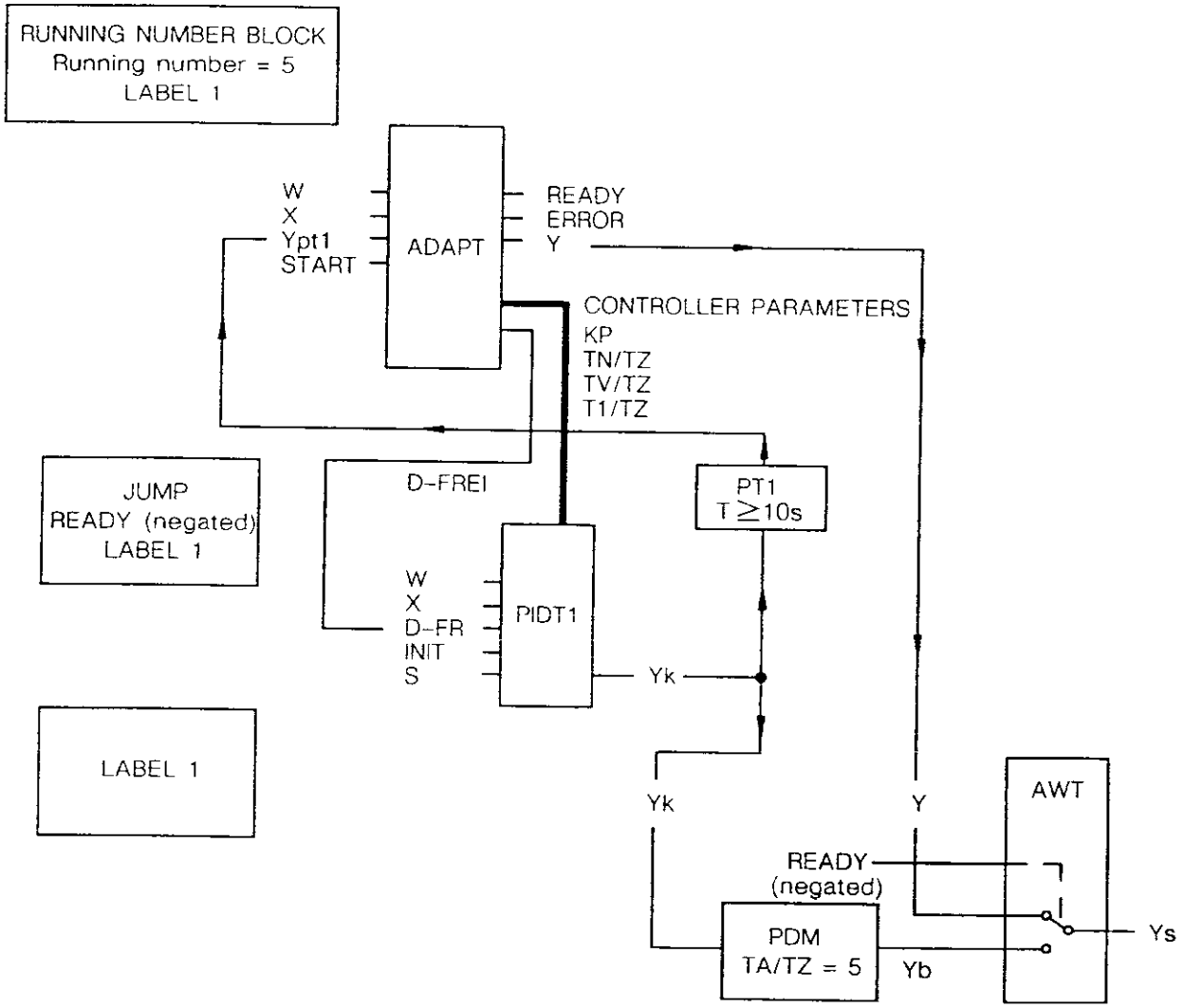
- The PIDT1 controller must be skipped when adaption is active. This is achieved by virtue of the fact that the negated READY signal of the adaption block is taken as a jump condition.

- Either the manipulated variable of the adaption or of the closed-loop control block is selected with the selection gate (AWT). This ensures that the controlled system is run either only by the controller (closed-loop control mode) or only by the adaption block (adaption mode). The selection gate is switched with the READY signal of the adaption block.

- The current manipulated variable of the controller is needed when adaption begins. As this is subject to a continuous oscillation in controlled mode, it must be fed to the adaption block through a PT1 element. The PT1 element's time constant must be at least 10 seconds.

- The adjustment value (INIT) supplied by the adaption block can be transferred to the controller as follows:
  - A binary flag is set to the value "1" for *one* cycle with the 0/1 edge of the READY signal of the adaption block. This binary flag is applied to the input S of the controller as a setting condition. It is best to disable the controller's D component during setting (see also block PIDT1, "setting and resetting the controller").

Planning overview of the controller and adaption block



W: Command variable
X: Controlled variable
Ypt1: Filtered Yk
Yb: Binary manipulated variable in closed-loop control mode
Y: Manipulated variable during adaption

Yk: Continuous manipulated variable
Ys: Manipulated variable for controlled system
READY: ADAPT-ready message
START: Start of adaption
D-FREI: DT1 component enabled for PIDT1 controller

W WORD

Command variable (setpoint).

X WORD

Controlled variable (actual value).

YR WORD

Current manipulated variable of the controller at the start of adaption.

START BINARY

Start input.

0/1-edge: start of adaption.
1/0-edge: abort adaption.

READY BINARY

Ready message.

1-Signal reports regular termination of adaption.

ERROR WORD

Fault message.

• ERROR = 1: Error
At the start of adaption the controlled variable difference is less than the minimum value MINXD = 3843. Adaption is aborted.
Remedy: Increase the initial controlled deviation.

• ERROR = 2: Error
Less than 16 interpolation points were available for calculation of the reversing tangent. Adaption is aborted.
Remedy: Reduce the cycle time.

• ERROR = 3: Error
The delay time TU lies outside the permissible value range (TU ≤ 0). Adaption is aborted.
Remedy: Consult ABB.

• ERROR = -1: Warning
The actuation degree change lies below the required minimum value. Adaption is terminated in the regular fashion. However, a satisfactory control response cannot be guaranteed.
Remedy: Check the control response on commissioning. Consult ABB if this is unsatisfactory.

• ERROR = -2: Warning
The ratio TU/TG is greater than the permissible maximum value. Adaption is terminated in the regular fashion. However, a satisfactory control response cannot be guaranteed.
Remedy: Check the control response on commissioning. Consult ABB if this is unsatisfactory.

Y BINARY

Excitation signal (manipulated variable) for the controlled system during adaption.

INIT WORD

Initial value for the controller after termination of adaption.

KP WORD

Proportional coefficient: This is output as a percentage.

TN/TZ WORD

Integral action time, scaled through the PLC cycle time TZ.

TV/TZ WORD

Derivative action time, scaled to the PLC cycle time TZ.

T1/TZ WORD

Returning time, scaled to the PLC cycle time TZ.

D-FREI BINARY

Enabling for the controller's D component. This output is used to decide whether a PI or PIDT1 controller is required.
D-FREI = 0 -> PI controller
D-FREI = 1 -> PIDT1 controller

KS WORD

Gain of the controlled system.

TU WORD

Delay time of the controlled system.

TG WORD

Stabilization time of the controlled system.

EXAMPLE

FBD/LD	IL
not defined *)	!BA 0 ADAPT KW 01.00 EW 00.03 MW 05.11 E 00.08 M 04.07 MW 05.12 M 04.08 MW 06.00 MW 06.01 MW 06.02 MW 06.03 MW 06.04 M 04.04 MW 06.05 MW 06.06 MW 06.07

CE FBD Definition

not defined *)

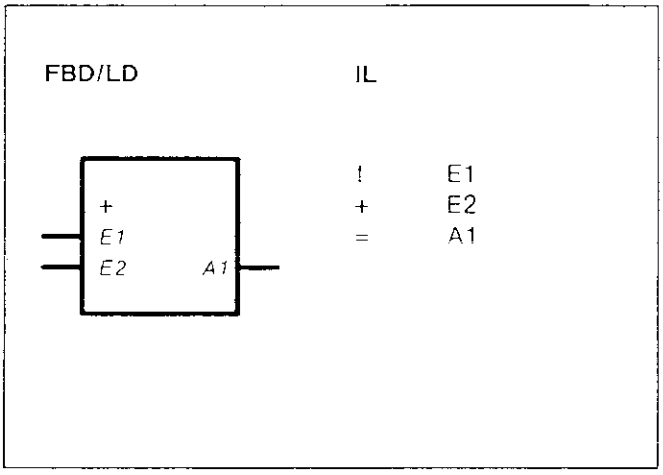
CE IL Definition

not defined *)

*) Note: A CE can be defined for the adaption block. Refer to 907 PC 32 description, general part, Chapter 15 (library, CE editor/ CE instruction editor).

ADDITION +

The values of the operands at the inputs of the connection element are added and the result is allocated to the operand at the output.



Parameters

E1	WORD	EW, MW, AW, KW	Summand 1
E2	WORD	EW, MW, AW, KW	Summand 2; The input can be duplicated
A1	WORD	MW, AW	Total

CE Data

Runtime:	
Basic runtime:	< 12 µs
Additional runtime:	5 µs per additional input (E3 ... En)
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The values of the operands at the inputs of the connection element are added and the result is allocated to the operand at the output.

The input E2 can be duplicated (E2 ... En). The input E1 and the output A1 can be negated.

Number range

Integer Word (16 Bits)

The following applies particularly here for the *non-negated* input E1:

If the inadmissible value 8000 H (-32768) is present at E1, it is corrected automatically to the permissible value 8001 H (-32767) before it is processed.

The following particularly applies here to the *non-negated* input E2: low limit: 8000 H (-32768)

The following generally applies:

- Low limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In two's complement arithmetic, the value 8000H (-32768) lies *outside* of the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value should reach the PLC

- through bit manipulations by the user or
- by reading in from outside the PLC or
- by an indirect word constant

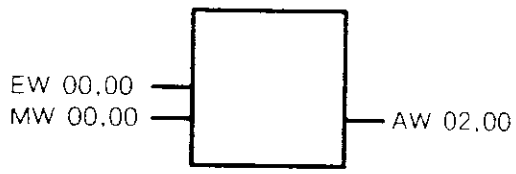
under no circumstances may a negation or subtraction be carried out on this value.

A permissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

In the case of allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

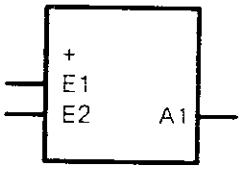
FBD/LD



IL

```
!    EW 00.00
+    MW 00.00
=    AW 02.00
```

CE FBD Definition

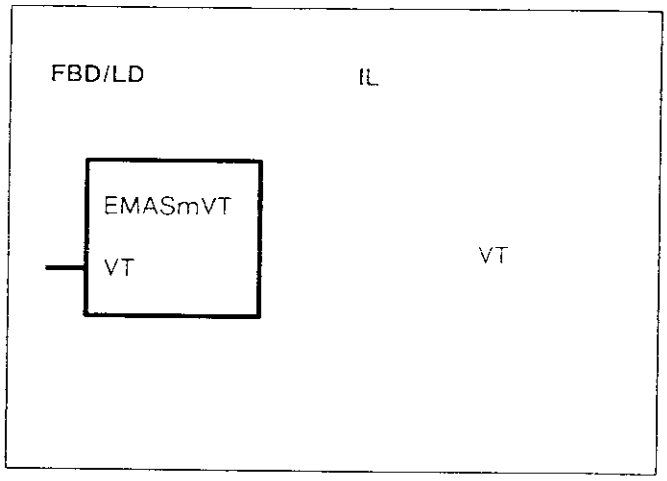


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	Y	P	N	0	0
E2	E	W	N	P	N	1	0
A1	A	W	Y	P	N	0	0

CE IL Definition

00000	!	PP 0	E1	Input WORD
	[	1		
00002	+	PP 1	E2	Input WORD (Duplication possible)
	]	1		
00005	=	PP 0	A1	Output WORD

The connection element EMASmVT serves to additionally plan comparison telegrams when using the EMASm block.



Parameters

VT	ALL	Comparison telegram, input capable of duplication

CE Data

Runtime:	
Basic runtime:	not available
Additional runtime:	not available
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

The connection element EMASmVT serves to additionally plan comparison telegrams when using the EMASm block.

In the programming system, the number of duplications of an input/output is limited to 63. Therefore, the number of comparison telegrams that can be planned is also limited to a maximum of 32.

Using the connection element EMASmVT, further comparison telegrams can be added for each EMASm block. The maximum number of comparison telegrams per EMASm is 99. At the same time, the comparison telegrams added with the connection element EMASmVT are also counted.

All comparison telegrams planned on the EMASm block are stored directly after EMASm in the instruction list. The comparison telegrams planned on the connection element EMASmVT are stored in the IL directly after the comparison telegrams of the EMASm block.

Important:
The connection element(s) EMASmVT must be planned in the FBD directly after the EMASm block.

The maximum occurring number of outputs (#ANU) for the received user data is specified at the EMASm block. When specifying this value, the comparison telegrams planned on the connection element EMASmVT must also be taken into account.

VT Comparison telegrams

The comparison telegrams are planned on the input VT capable of duplication. Planning is completely the same as on the VT input of the EMASm block.

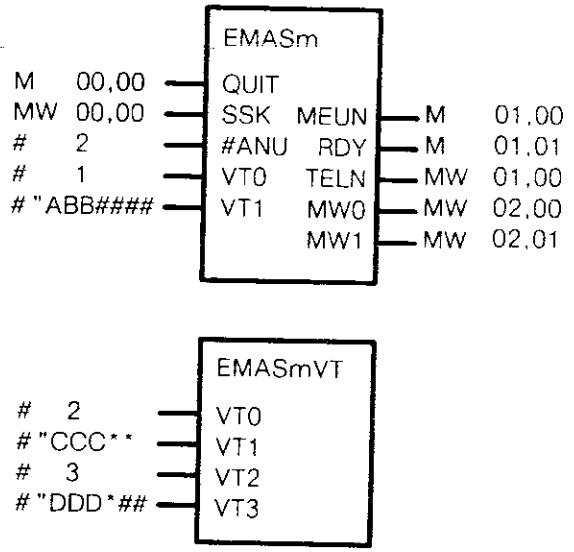
Note:

If comparison telegrams are not stored in the programming system in the IL, but are entered directly in the PLC by means of a terminal, the following applies:
- the comparison text begins with a start identifier: #"
- the comparison text ends with the end identifier: "#

Refer also to the PLC description, ABB Procontic T300, chapter entitled text processing.

Example

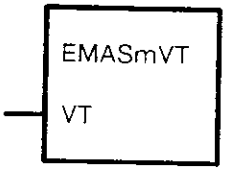
FBD/LD



IL

```
!BA
EMAS
      M   00,00      QUIT
      MW  00,00      SSK
      #   2          #ANU
      M   01,00      MEUN
      M   01,01      RDY
      MW  01,00      TELN
      MW  02,00      MW0
      MW  02,01      MW1
      #   1          Telegram No. 1
      #"ABB####      Text 1
      #   2          Telegram No. 2
      #"CCC**        Text 2
      #   3          Telegram No. 3
      #"DDD*##       Text 3
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
VT	E	X	N	P	Y	1	0

CE IL Definition

```
        [       1
00000           PP  1       VT      Input comparison telegrams
        ]       1
```

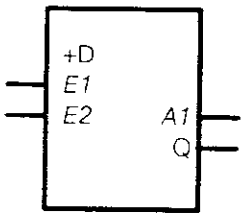
ADDITION DOUBLE WORD

+D

The value of the operand at the input E1 is added to the value of the operand at the input E2 and the result is allocated to the operand at the output A1.

The result is limited to the maximum or minimum value of the number range. If limiting has taken place, a one signal is allocated to the binary operand at the output Q. If no limiting has taken place, a zero signal is allocated to the binary operand at the output Q.

FBD/LD



IL

```
!BA   0
ADDD
      E1
      E2
      A1
      Q
```

Parameters

E1	DOUBLE WORD	MD, KD	Summand 1
E2	DOUBLE WORD	MD, KD	Summand 2
A1	DOUBLE WORD	MD	Total
Q	BINARY	A, M	Total, limited

CE Data

Runtime:	
Basic runtime:	49 ... 52 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the operand at the input E1 is added to the value of the operand at the input E2 and the result is allocated to the operand at the output A1.

The result is limited to the maximum or minimum value of the number range. If limiting has taken place, a 1 signal is allocated to the binary operand at the output Q. If no limiting has taken place, a 0 signal is allocated to the binary operand at the output Q.

The inputs and outputs cannot be duplicated, nor can they be negated.

Number range

Integer, double word (32 bits).

The following particularly applies here to the inputs E1 and E2:

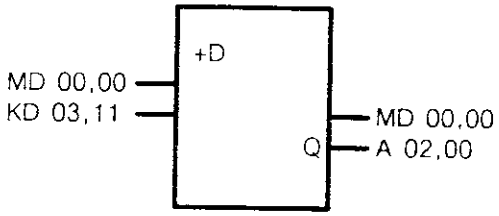
- Low limit: 8000 0000 H - 2 147 483 648

The following generally applies:

- Low limit: 8000 0001 H -2 147 483 647
- High limit: 7FFF FFFF H +2 147 483 647
- Inadmissible value: 8000 0000 H ---

Example

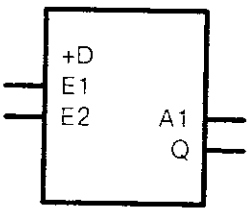
FBD/LD



IL

```
!BA    0
ADDD
       MD 00,00
       KD 03,11
       MD 00,00
       A  02,00
```

CE FBD Definition



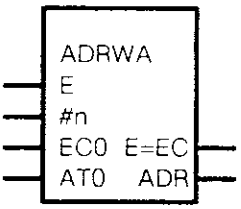
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
E2	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	ADDD			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	E2	Input DOUBLE WORD
00004		PP 0	A1	Output DOUBLE WORD
00005		PP 0	Q	Output BINARY

One of the operands planned at the inputs AT0 ... ATn-1 is selected with this function block. Of this selected operand, the indirect address is generated and is provided at the output ADR.

FBD/LD



IL

```
!BA     0
ADRWA
        E
        #n
        EC0
        AT0
        E=EC
        ADR
```

Parameters

E	WORD	EW, AW, MW, KW	Input value
#n	DIRECT CONSTANT	#, #H	Quantity EC or AT
EC0	WORD	EW, AW, MW, KW	Input code; input can be duplicated
AT0	WORD	EW, AW, MW, KW	Output table, input is also duplicated with EC0
E=EC	BINARY	A, M	Input value = input code
ADR	WORD	AW, MW	indirect address

CE Data

Runtime:	
Basic runtime:	44 µs
Additional runtime:	8 µs per operand EC0..EC_{n-1}
Output updating:	no
Number of historical values:	none
Available as of:	ABB Procontic T320 V4 / 935 PC 82 R101 / 35 ZE 93 R101

Description

One of the operands planned at the inputs AT0 ... ATn-1 is selected with this function block. Of this selected operand, the indirect address is generated and is provided at the output ADR.

The inputs and outputs cannot be negated/inverted. The input EC0 can be duplicated, whereby the input AT0 is also duplicated automatically.

Reading/writing operands indirectly

The function blocks AWM or USM use the indirect address generated by the block ADRWA in order to read or write the operand selected with the block ADRWA. Therefore, the block ADRWA *and* the block AWM or USM is needed to read or write operands indirectly. To do this, the operands to be read or written are listed at the inputs AT0...ATn-1 of the ADRWA block and the read or write access is then performed by the AWM or USM block.

Advantages of indirect addressing:

- In suitable applications, the PLC program is simplified substantially, thus reducing the planning effort.
- Access to any number of operands (multiplex function) is achieved with only *one* block (AWM or USM). In this process, the ADRWA block represents a powerful tool with which the operands to be accessed can be selected in a very flexible manner.

Selecting an operand from the output table AT0 ... ATn-1

The block compares the value at the input E successively against the values at the inputs EC0 ... ECn-1. The comparison is restarted each time the block is called, i.e. it begins with the input EC0.

- If the value at the input E agrees with one of the values at the inputs EC0 ... ECn-1:
  - The output E=EC is set to 1 (hit),
  - The *allocated* operand is selected from the output table AT0...ATn-1.
- If the value at the input E does not agree with one of the values at the inputs EC0...ECn-1:
  - The output E=EC is set to 0 (no hit)
  - *no* operand is selected from the output table AT0...ATn-1.

Convention for allocation between EC0...ECn-1 and AT0...ATn-1:

EC0 -> AT0
EC1 -> AT1
. .
. .
EC_{n-1} -> AT_{n-1}

Generation of the indirect address
The indirect address of an operand is generated if it has been selected from the output table AT0...ATn-1. This is done by taking the address of the selected operand and by allocating it as a value to the operand specified at the ADR output. The value of the operand specified at the ADR output is therefore the address of the operand selected from the output table AT0...ATn-1.
Def.: An indirect address is an operand whose value is the address of another operand.

Use of an indirect address
The function blocks AWM and USM access operands with indirect addressing. Therefore, the blocks AWM and USM require the indirect address generated by the ADRWA block at the inputs.

E WORD
One of the operands is selected from the output table AT0...ATn-1 with the aid of the operand specified at the input E and the operands specified at the inputs EC0...ECn-1 and its indirect address is then generated.
To do this, the block compares the value at the input E successively against the values at the inputs EC0... ECn-1. The comparison restarts each time the block is called, i.e. beginning with the input EC0.

- If the value at the input E agrees with one of the values at the inputs EC0...ECn-1:
  - The output E=EC is set to 1 (hit).
  - The *allocated* operand is selected from the output table AT0...ATn-1 and its indirect address is generated.
- If the value at the input E does not agree with one of the values at the inputs EC0...ECn-1:
  - the output E=EC is set to 0 (no hit)
  - no operand is selected from the output table AT0... ATn-1 and accordingly no indirect address is generated either.

#n DIRECT CONSTANT
The number of the planned inputs EC0...ECn-1 is specified at the input #n. It is specified as a direct constant.
Example: The following are planned: EC0, EC1, EC2 -> #n = 3.

EC0...ECn-1 WORD
The operands for the comparison values are specified at the inputs EC0...ECn-1. The input EC0 can be duplicated. The value at the input E is compared against these comparison values and is checked for conformity. If they agree, the allocated operand is selected from the output table AT0...ATn-1 and its indirect address is generated. The values at the inputs EC0...ECn-1 are variable because they are normal operands. This is why the function for comparison between the value at the input E and the values at the inputs EC0...ECn-1 is very flexible and powerful.

Affiliations between EC0...ECn-1 and AT0...ATn-1:

EC0 ---> AT0
EC1 ---> AT1
. .
. .
EC_{n-1} ---> AT_{n-1}

AT0...ATn-1 WORD
The operands whose indirect addresses are to be generated are specified at the outputs AT0...ATn-1. When the input EC0 is duplicated, the input AT0 is also duplicated automatically.

E=EC BINARY

The output E=EC indicates whether or not the value at the input E agrees with one of the values at the inputs EC0...ECn-1.

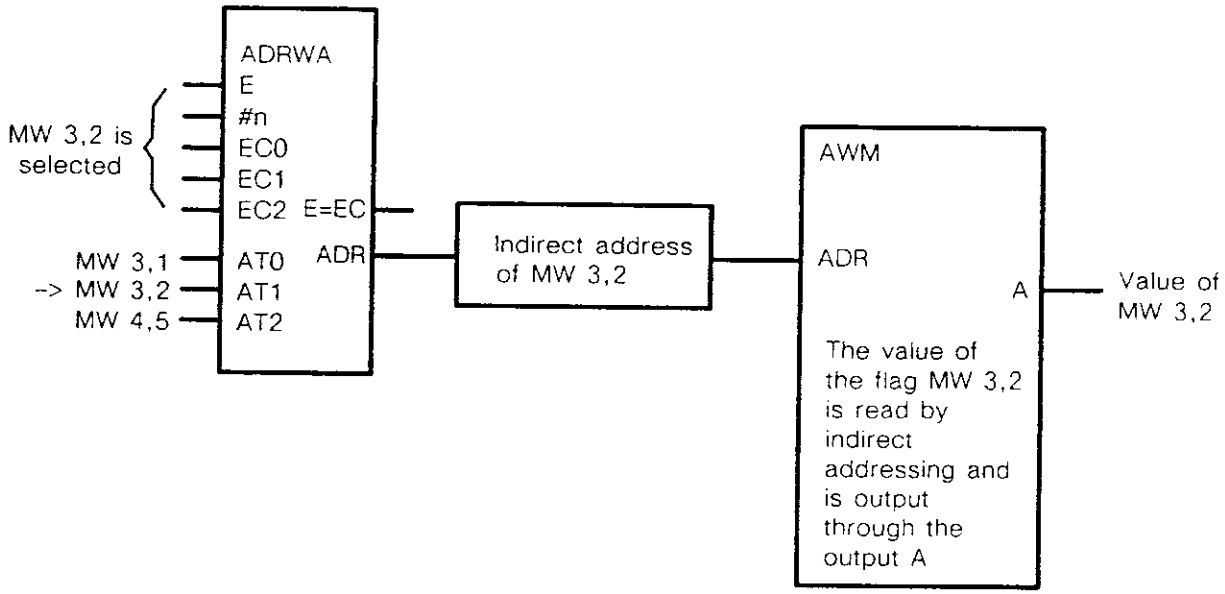
E=EC = 0 -> No agreement
E=EC = 1 -> The value at the input E agrees with one of the values at the inputs EC0...ECn-1.

ADR WORD

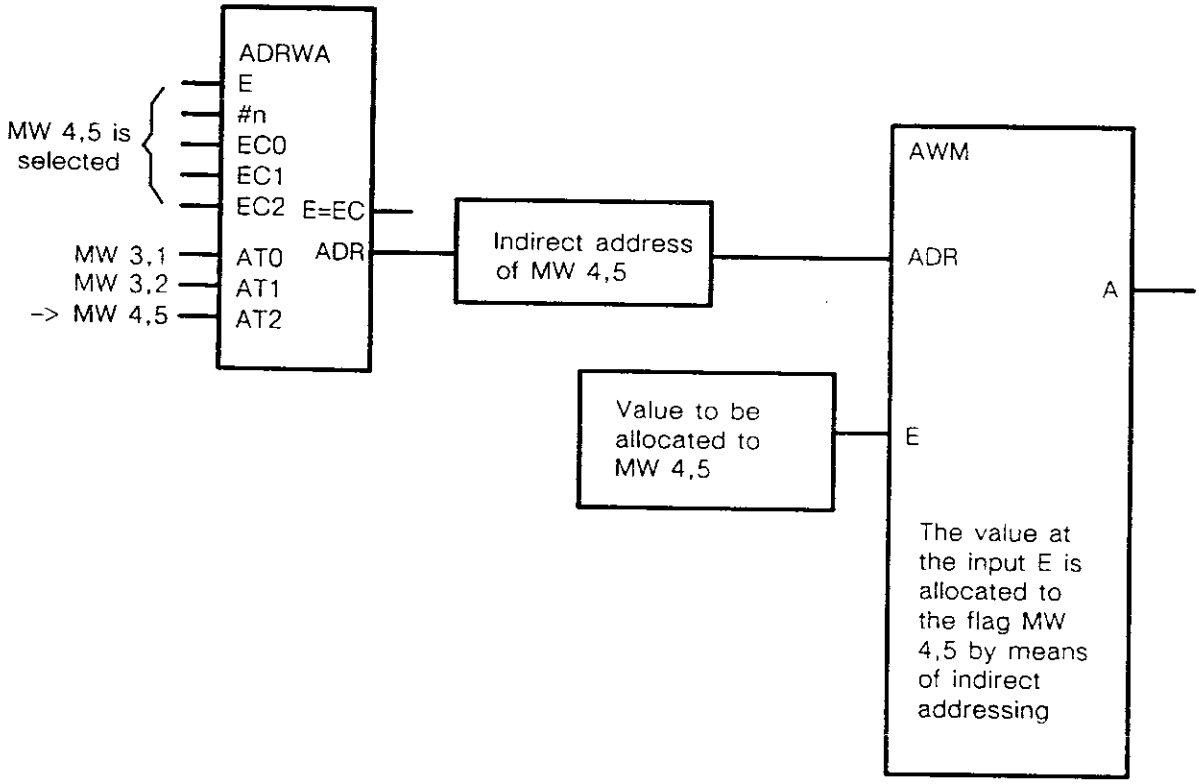
Together with its value, the operand at the ADR output represents an indirect address. This is the indirect address of the operand selected from the output table AT0...ATn-1. The indirect address is produced by virtue of the fact that the address of the selected operand is allocated as a value to the operand at the output ADR.

If no agreement between the input E and the inputs EC0...ECn-1 is determined during comparison, then no indirect address is generated either. Therefore, no value is allocated to the ADR output. In this case, the ADR output is not updated.

Example: Indirect reading of the flag MW 3,2

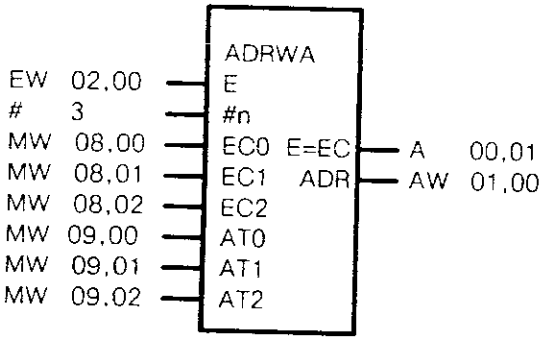


Example: Indirect writing of the flag MW 4,5



Example

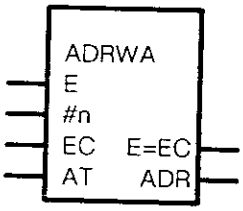
FBD/LD



IL

```
!BA    0
ADRWA
       EW  02,00
       #   3
       MW  08,00
       MW  08,01
       MW  08,02
       MW  09,00
       MW  09,01
       MW  09,02
       A   00,01
       AW  01,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
EC	E	W	N	P	Y	1	0
AT	E	W	N	P	Y	1	0
E=EC	A	L	N	P	Y	0	0
ADR	A	W	N	P	Y	0	0

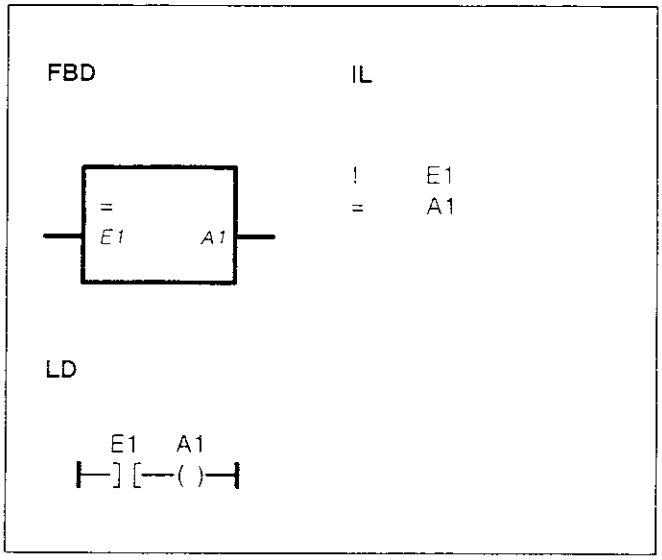
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001				
00002		PP 0	E	Input WORD (Input value)
00003		PP 0	#n	Input WORD (Number of ECs and ATs)
	[	1		
00004		PP 0	EC	Input WORD (Input code)
	]	1		
	[	1		
00005		PP 0	AT	Input WORD (output table)
	]	1		
00006		PP 0	E=EC	Output BINARY (Input value=Ecode)
00007		PP 0	ADR	Output WORD (operand address)

ALLOCATION

=

This connection element allocates the value of the operand at the input to the operand at the output.



Parameters

E1	BINARY	E, M, A, S, K	Source
A1	BINARY	M, A, S	Target, output capable of duplication

CE Data

Runtime:	
Basic runtime:	4.3 µs
Additional runtime:	2.3 µs per additional output (A2 ... An)
Ouput updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801/ 35 ZE 93 R101

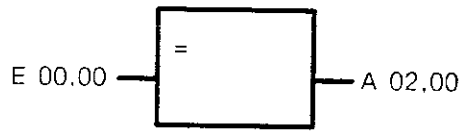
Description

This connection element allocates the value of the operand at the input to the operand at the output.

The output A1 is capable of duplication. The input and the output are capable of inversion.

Example

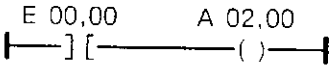
FBD



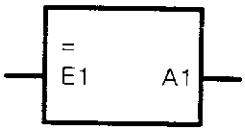
IL

```
!     E 00,00
=     A 02,00
```

LD



CE FBD Definition



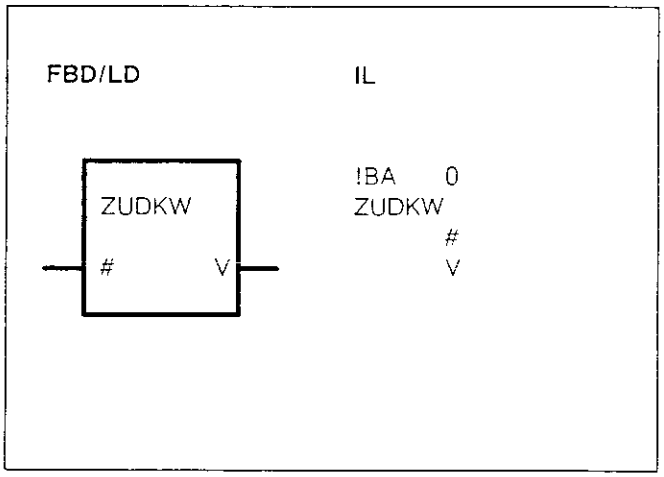
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	L	Y	P	N	0	0
A1	A	L	Y	P	N	1	0

CE IL Definition

00000	!	PP 0	E1	Input BINARY
	|	1	
00002	=	PP 1	A1	Output BINARY (capable of duplication)
	¦	1		

ALLOCATION DIRECT CONSTANT TO WORD VARIABLE ZUDKW

The function block serves to allocate a numerical value to a word variable. The numerical value is specified as a direct constant.



Parameters

#	DIRECT KONSTANT	#, #H	Numerical value which is to be allocated to the word variable at output V
V	WORD	EW, MW, AW	Word variable to which the numerical value of input # is to be allocated

CE Data

Runtime:	
Basic runtime:	10 µs
Additional runtime:	----
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T300 V8.5

Description

The function block serves to allocate a numerical value to a word variable. The numerical value is specified as a direct constant.

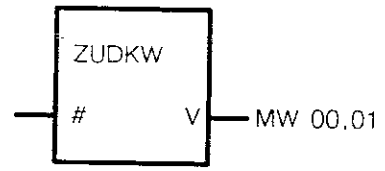
Note
EPROM programming in case of this block is not possible via the programming system. EPROM programming must be carried out in terminal emulation as operating function using command PU. At the same time the EPROM programming unit must be connected to serial interface 1.

DIRECT CONSTANT (#, #H)
This numerical value is allocated to the word variable at output V.

V WORD
Word variable to which the numerical value of input # is allocated.

Example

FBD/LD



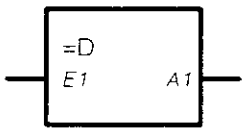
IL

```
!BA    0
ZUDKW
# 1234
MW 00.01
```

ALLOCATION, DOUBLE WORD =D

The value of the operand at the input E1 is allocated to the operand at the output A1.

FBD/LD



IL

```
!BA   0
ZUWD
      E1
      A1
```

Parameters

E1	DOUBLE WORD	MD, KD	Source
A1	DOUBLE WORD	MD	Target

CE Data

Runtime:	
Basic runtime:	31 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the operand at the input E1 is allocated to the operand at the output A1.

If the inadmissible value 8000 0000 H should appear at the input for any particular reason, the permissible value 8000 0001 H (–2 147 483 647) will be allocated to the output A1. Therefore, the inadmissible value will be corrected.

The input and the output can neither be duplicated nor negated.

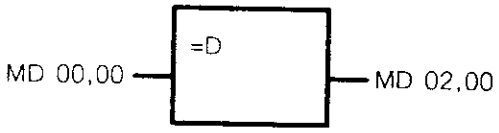
Number range

Integer double word (32 bits)

• Low limit:	8000 0001 H	–2 147 483 647
• High limit:	7FFF FFFF H	+2 147 483 647
• Inadmissible value:	8000 0000 H	---

Example

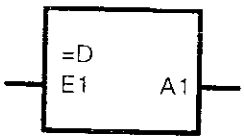
FBD/LD



IL

```
!BA    0
ZUWD
       MD 00,00
       MD 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0

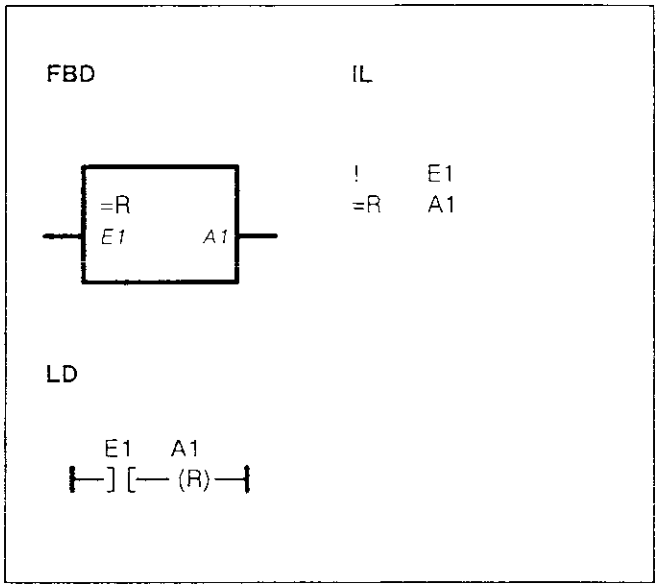
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	ZUWD			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	A1	Output DOUBLE WORD

ALLOCATION RESET MEMORY =R

This connection element resets a stored binary variable.

A state 1 at the input sets the operand at the output to a state 0. A state 0 at the input has no influence on the operand at the output.



Parameters

E1	BINARY	E, M, A, S, K	Reset condition
A1	BINARY	M, A	Store variable

CE Data

Runtime:	
Basic runtime:	4.3 µs
Additional runtime:	2.3 µs per additional output (A3 ... An)
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

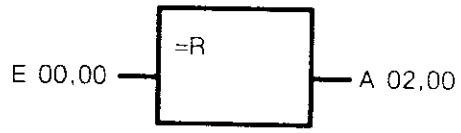
A state 1 at the input sets the operand at the output to a state 0. A state 0 at the input has no influence on the operand at the output.

IMPORTANT:
This CE must only be used as an output CE, i. e. in the FBD it must not be connected further by a line on the output side. An operand (M or A) *must* be specified at the output.

The output A1 can be duplicated (A2...An). The input E1 can be inverted.

Example

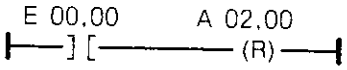
FBD



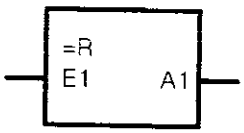
IL

```
!     E 00.00
=R    A 02.00
```

LD



CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	L	Y	P	N	0	0
A1	A	L	N	P	N	1	0

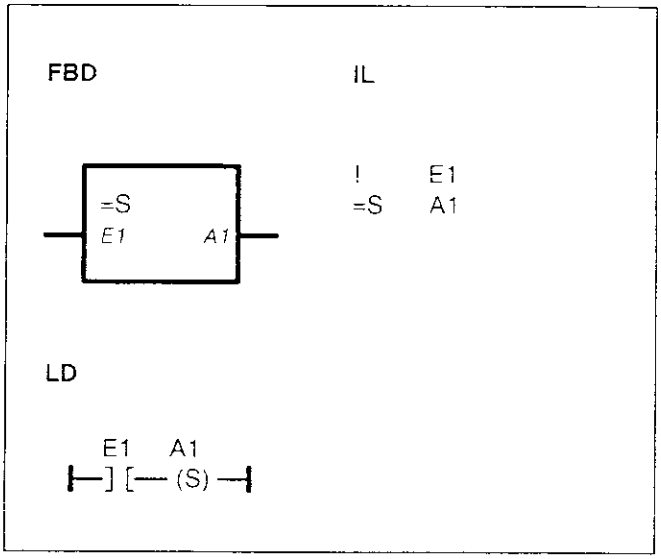
CE IL Definition

00000	! [	PP 0 1	E1	Input BINARY
00002	=R]	PP 1 1	A1	Output BINARY (capable of duplication)

ALLOCATION SET MEMORY =S

A binary variable is set in latching form with this connection element.

A state 1 at the input sets the operand at the output to a state 1. A state 0 at the input has no influence on the operand at the output.



Parameters

E1	BINARY	E, M, A, S, K	Set condition
A1	BINARY	M, A	Storage variable

CE Data

Runtime:	
Basic runtime:	4.3 µs
Additional runtime:	2.3 µs per additional output (A3 ... An)
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

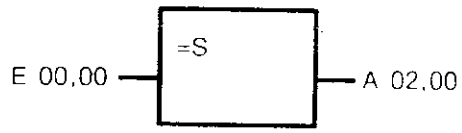
A state 1 at the input sets the operand at the output to a state 1. A state 0 at the input has no influence on the operand at the output.

IMPORTANT:
This CE must only be used as an output CE, i. e. in the FBD it must not be connected further by a line on the output side. An operand (M or A) *must* be specified at the output.

The output A1 can be duplicated (A2...An). The input E1 can be inverted.

Example

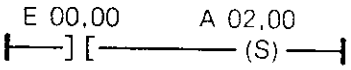
FBD



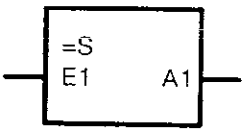
IL

```
!     E 00,00
=S    A 02,00
```

LD



CE FBD Definition

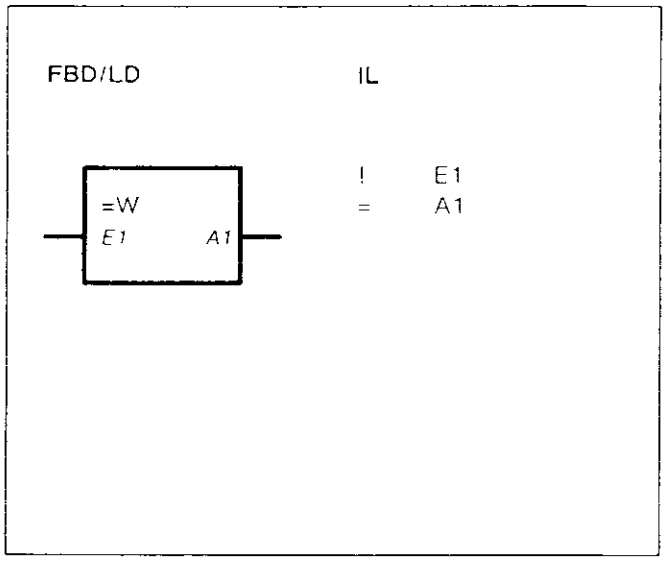


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	L	Y	P	N	0	0
A1	A	L	N	P	N	1	0

CE IL Definition

00000	! [	PP 0 1	E1	Input BINARY
00002	=S]	PP 1 1	A1	Output BINARY (capable of duplication)

This connection element allocates the value of the operand at the input to the output capable of duplication.



Parameters

E1	WORD	EW, MW, AW, KW	Source
A1	WORD	MW,AW	Target; output capable of duplication

CE Data

Runtime:	
Basic runtime:	7 µs
Additional runtime:	5 µs per additional output
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, 801 / 35 ZE 93 R101

Description

This connection element allocates the value of the operand at the input to the output capable of duplication.

The output A1 can be duplicated (A2...An). The input and the output can be negated.

Number range

Integer word (16 bits).

The following especially applies here to the non-negated input E1: If the inadmissible value 8000 H (-32768) is present at the input E1, the permissible value 8001 H (-32767) is allocated to the output A1.

The following generally applies:

- Low limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

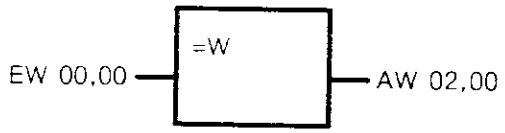
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

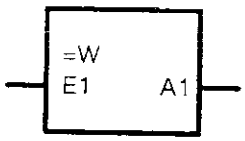
FBD/LD



IL

```
!     EW 00,00
=     AW 02,00
```

CE FBD Definition

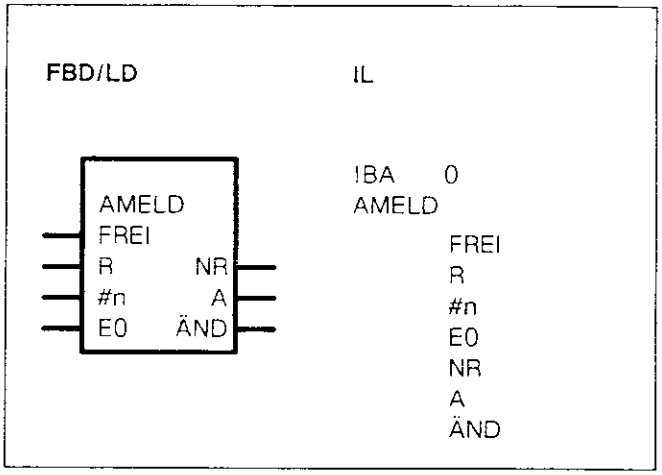


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	Y	P	N	0	0
A1	A	W	Y	P	N	1	0

CE IL Definition

00001	! [	PP 0 1	E1	Input WORD
00002	=]	PP 1 1	A1	Output WORD (capable of duplication)

This function block monitors the analog values present at the input E0 capable of duplication for a change.



Parameters

FREI	BINARY	A, E, M, S, K	Block enabling
R	BINARY	A, E, M, S, K	Reset
#n	DIRECT CONSTANT	#, #H	Number of input values
E0	WORD	EW, AW, MW, KW	Input values: Input can be duplicated
NR	WORD	AW, MW	Number of the input value
A	WORD	AW, MW	Current input value
ÄND	BINARY	A, M	Change detected

CE Data

Runtime:	
Basic runtime:	57 µs
Additional runtime:	31 µs per input E0...E_{n-1}
Output updating:	yes, if FREI = 1
Number of historical values:	#n + 3 words
Available as of:	ABB Procontic T320 V4 / 935 PC 82 R101 / 35 ZE 93 R101

Description

This function block monitors the analog values present at the input E0 capable of duplication for a change.

The inputs and outputs cannot be negated/inverted. The input E0 can be duplicated.

Recognition of a change

Each time the block is processed, the current input values at the inputs E0...En-1 are successively compared against the historical values (input values from the previous processing of the block). If a change is recognized at one of the inputs E0...En-1:

- this is indicated at the ÄND output
- the number of the input where the change was recognized is output through the NR output
- the changing input value is output through the A output

Each time the block is processed, a change at *one* input only is recognized. If a change is recognized, the inputs following the one where the change was previously discovered are monitored the next time the block is processed.

Initialization of historical values

The first time the block is processed after PLC initialization (FREI = 1) or enabling of processing after it had been disabled (FREI changes from 0 to 1), all current input values are assumed once as historical values and all outputs are set to the value 0. These initialized historical values now represent the starting basis for recognition of changes.

FREI BINARY

Processing of the block is enabled with the FREI input.
FREI = 0 -> Block is not processed
FREI = 1 -> Processing of the block is enabled

If FREI = 0, the outputs of the block are also no longer updated.

R BINARY

The block can be reset with the R input.
R = 0 -> No reset
R = 1 -> Reset of the block
Reset signifies:
- Adoption of the current values at the inputs E0...En-1 as historical values.
- All outputs are set to the value 0

#n DIRECT CONSTANT

The number of values to be monitored at the inputs E0...En-1 is specified at the input #n. The number is specified as a direct constant.
Range for #n: $1 \leq \#n \leq 127$

E0...En-1 WORD

The input E0 can be duplicated (E0...En-1).
The operands to be monitored for a change are specified at the inputs E0...En-1.

NR WORD

The serial number of the input E0...En-1 where a change has been discovered is output through the output NR.
If *no* output change is discovered during processing of the block, the number of the input changing last is still output through the output NR.
The following affiliations apply:
Change discovered at E0 -> NR = 0
Change discovered at E1 -> NR = 1
Change discovered at En-1 -> NR = n-1

A WORD

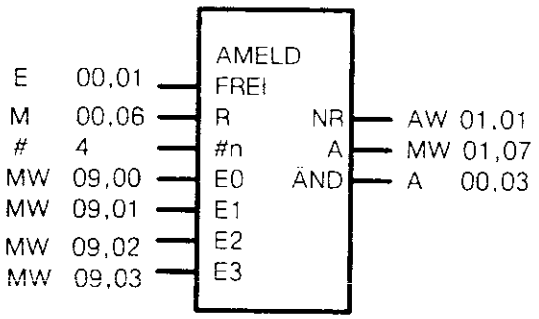
If a change is discovered at one of the inputs E0...En-1, the changing input value is allocated to the output A.
If *no* change is discovered at the inputs E0...En-1 during processing of the block, the value of the input changing last is still output through the output A.

ÄND BINARY

The output ÄND indicates whether or not a change has been discovered at the inputs E0...En-1.
ÄND = 0 -> No change discovered
ÄND = 1 -> Change discovered

Example

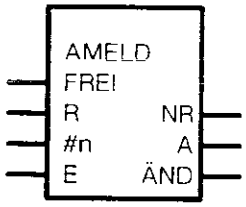
FBD/LD



IL

```
!BA    0
AMELD
       E    00,01
       R    00,06
       #    4
       MW   09,00
       MW   09,01
       MW   09,02
       MW   09,03
       AW   01,01
       MW   01,07
       A    00,03
```

CE FBD Definition



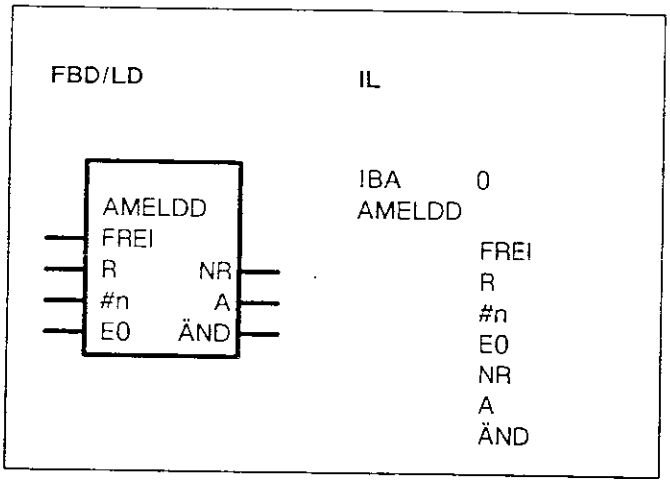
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
R	E	L	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
E	E	W	N	P	Y	1	0
NR	A	W	N	P	Y	0	0
A	A	W	N	P	Y	0	0
ÄND	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	AMELD			
00002		PP 0	FREI	Input BINARY (block enable)
00003		PP 0	R	Input BINARY (Reset)
00004		PP 0	#n	# CONSTANT (Number of input values)
	[	1		
00005		PP 1	E	Input WORD (input values)
	]	1		
00006		PP 0	NR	Output WORD (No.)
00007		PP 0	A	Output WORD (current input value)
00008		PP 0	ÄND	Output BINARY (change detected)

ANALOG VALUE CHANGE ANNUNCIATOR DOUBLE WORD — AMELDD

This function block monitors the analog values present at the input E0 capable of duplication for changes.



Parameters

FREI	BINARY	A, E, M, S, K	Block enable
R	BINARY	A, E, M, S, K	Reset
#n	DIRECT CONSTANT	#, #H	Number of input values
E0	DOUBLE WORD	MD, KD	Input values; input can be duplicated
NR	WORD	AW, MW	Number of the input value
A	DOUBLE WORD	MD	Current input value
ÄND	BINARY	A, M	Change detected

CE Data

Runtime:	
Basic runtime:	64 µs
Additional runtime:	41 µs per input E0 ... En-1
Output updating:	yes if FREI = 1
Number of historical values:	(2 * #n) + 4 words
Available as of:	ABB Procontic T320 V6 / 935 PC 82 R301 / 35 ZE 93 R201

Description

This function block monitors the analog values present at the input E0 capable of duplication for changes.

The inputs and outputs cannot be negated/inverted. The input E0 can be duplicated.

Recognitioin of a change

Each time the block is processed, the current input values at the inputs E0...En-1 are successively compared against the historical values (input values from the previous processing of the block). If a change is recognized at one of the inputs E0...En-1:

- this is indicated at the ÄND output
- the number of the input where the change was recognized is output through the NR output
- the changing input value is output through the A output

Each time the block is processed, a change at *one* input only is recognized. If a change is recognized, the inputs following the one where the change was previously discovered are monitored the next time the block is processed.

Initialization of historical values

The first time the block is processed after PLC initialization (FREI = 1) or enabling of processing after it had been disabled (FREI changes from 0 to 1), all current input values are assumed once as historical values and all outputs are set to the value 0. These initialized historical values now represent the starting basis for recognition of changes.

FREI BINARY

Processing of the block is enabled with the FREI input.
FREI = 0 -> Block is not processed
FREI = 1 -> Processing of the block is enabled
If FREI = 0, the outputs of the block are also no longer updated.

R BINARY

The block can be reset with the R input.
R = 0 -> No reset
R = 1 -> Reset of the block
Reset signifies:
- Adoption of the current values at the inputs E0...En-1 as historical values.
- All outputs are set to the value 0.

#n DIRECT CONSTANT

The number of values to be monitored at the inputs E0...En-1 are specified at input #n. It is specified as a direct constant.
Range for #n: $1 \leq \#n \leq \#63$

E0...En-1 DOUBLE WORD

The input E0 can be duplicated (E0...En-1).
The operands to be monitored for a change are specified at the inputs E0...En-1.

NR WORD

The serial number of the input E0...En-1 where a change has been discovered is output through the output NR.
If *no* change is discovered during processing of the block the number of the input changing last is still output through the output NR.
The following affiliations apply:
Change discovered at E0 -> NR = 0
Change discovered at E1 -> NR = 1
Change discovered at En-1 -> NR = n-1

A DOUBLE WORD

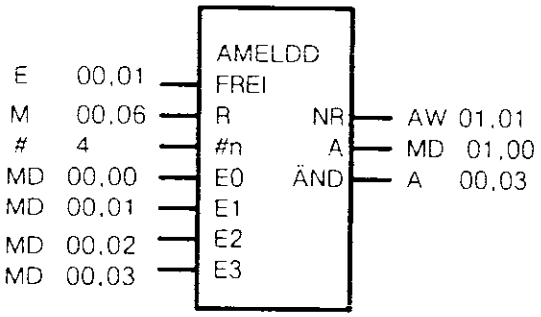
If a change is discovered at one of the inputs E0...En-1, the changing input value is allocated to the output A.
If *no* change is discovered at the inputs E0...En-1 during processing of the block, the value of the input changing last is still output through the output A.

ÄND BINARY

The output ÄND indicates whether or not a change has been discovered at the inputs E0...En-1.
ÄND = 0 -> No change discovered
ÄND = 1 -> Change discovered

Example

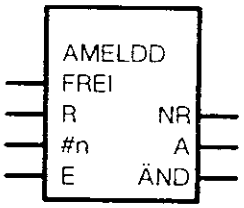
FBD/LD



IL

```
!BA     0
AMELDD
        E     00.01
        R     00.06
        #     4
        MD    00.00
        MD    00.01
        MD    00.02
        MD    00.03
        AW    01.01
        MD    01.00
        A     00.03
```

CE FBD Definition

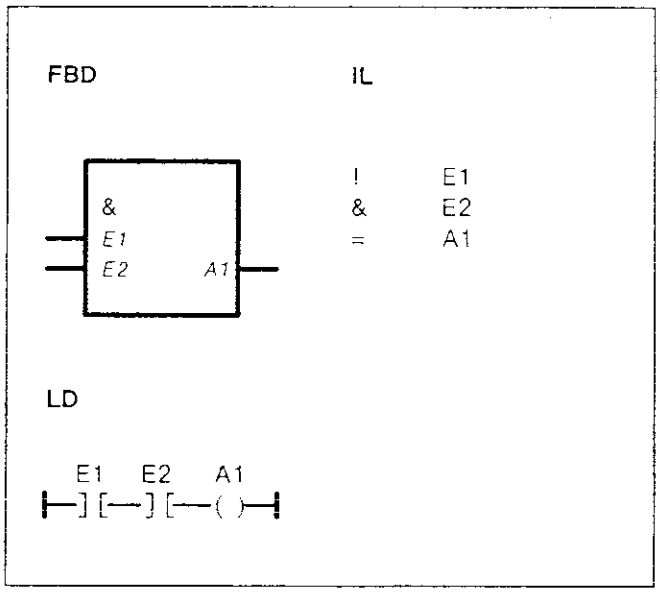


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
R	E	L	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
E	E	D	N	P	Y	1	0
NR	A	W	N	P	Y	0	0
A	A	D	N	P	Y	0	0
ÄND	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	AMELDD			
00002		PP 0	FREI	Input BINARY (block enable)
00003		PP 0	R	Input BINARY (Reset)
00004		PP 0	#n	# CONSTANT (number of input values)
	[	1		
00005		PP 1	E	Input DOUBLE WORD (input values)
	]	1		
00006		PP 0	NR	Output WORD (No.)
00007		PP 0	A	Output DOUBLE WORD (current input value)
00008		PP 0	ÄND	Output BINARY (change detected)

This connection element realizes a logical AND combination of the operands at the inputs. The result is allocated to the operand at the output.



Parameters

E1	BINARY	E, M, A, S, K	Operand 1 of the AND combination
E2	BINARY	E, M, A, S, K	Operand 2 of the AND combination, capable of duplication
A1	BINARY	M, A, S	Result of the AND combination

CE Data

Runtime:
- Basic runtime: 6.6 µs
- Additional runtime: 2.3 µs per additional input

Output updating: yes
Number of historical values: none
Available as of: ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This connection element realizes a logical AND combination of the operands at the inputs. The result is allocated to the operand at the output.

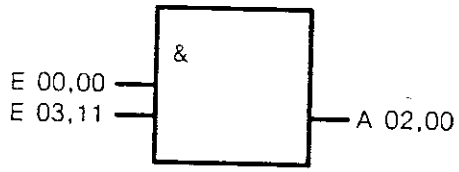
The input E2 is capable of duplication. All inputs and the output are capable of inversion.

Truth table:

E1	E2	A1
0	0	0
1	0	0
0	1	0
1	1	1

Example

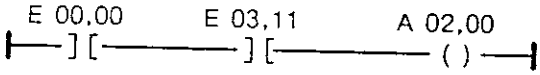
FBD



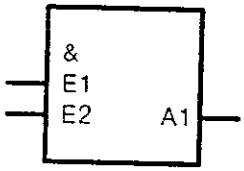
IL

```
!    E 00,00
&    E 03,11
=    A 02,00
```

LD



CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	L	Y	P	N	0	0
E2	E	L	Y	P	N	1	0
A1	A	L	Y	P	N	0	0

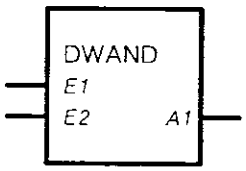
CE IL Definition

00000	!	PP 0	E1	Input BINARY
	[	1		
00002	&	PP 1	E2	Input BINARY (capable of duplication)
	]	1		
00004	=	PP 0	A1	Output BINARY

AND COMBINATION, DOUBLE WORD — DWAND

This function block generates, bit by bit, the AND combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

FBD/LD



IL

```
!BA     0
DWAND
        E1
        E2
        A1
```

Parameters

E1	DOUBLE WORD	MD, KD	Operand 1
E2	DOUBLE WORD	MD, KD	Operand 2
A1	DOUBLE WORD	MD	Result of the AND combination

CE Data

Runtime:	
Basic runtime:	49 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block generates, bit by bit, the AND combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

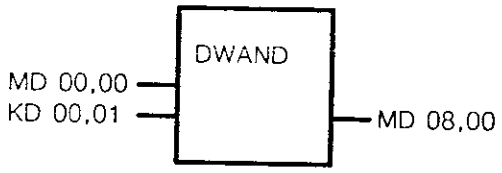
The inputs and the output can neither be duplicated nor inverted/negated.

Example

E1	1.0.0.0	0.0.1.1	0.0.1.0	0.1.1.0	1.0.1.0	1.1.0.0	0.0.1.1	0.1.0.1
E2	1.0.0.1	0.1.1.0	0.0.1.0	1.1.1.1	1.1.1.1	0.0.0.0	0.1.1.0	1.1.0.0
A1	1.0.0.0	0.0.1.0	0.0.1.0	0.1.1.0	1.0.1.0	0.0.0.0	0.0.1.0	0.1.0.0

Example

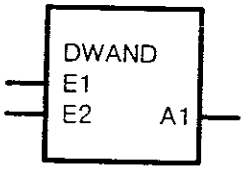
FBD/LD



IL

```
!BA    0
DWAND
       MD 00,00
       KD 00,01
       MD 08,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
E2	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0

CE IL Definition

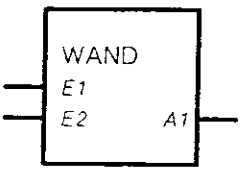
00000	!BA	0	Nr	Block No. (preset to 0)
00001	DWAND			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	E2	Input DOUBLE WORD
00004		PP 0	A1	Output DOUBLE WORD

AND COMBINATION, WORD

WAND

This function block generates, bit by bit, the AND combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

FBD/LD



IL

```
!BA   0
WAND
      E1
      E2
      A1
```

Parameters

E1	WORD	EW, MW, AW, KW	Operand 1
E2	WORD	EW, MW, AW, KW	Operand 2
A1	WORD	MW, AW	Result of the AND combination

CE Data

Runtime:

Basic runtime:	28 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block generates, bit by bit, the AND combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

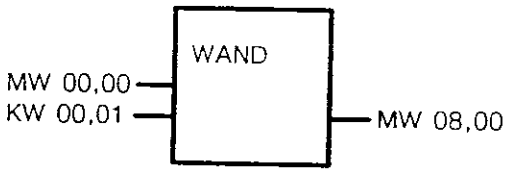
The inputs and the output can neither be duplicated nor inverted/negated.

Example:

E1	0.0.0.0	0.0.1.1	0.0.1.0	0.1.1.0
E2	1.0.0.1	0.0.0.0	0.0.1.0	1.1.1.1
A1	0.0.0.0	0.0.0.0	0.0.1.0	0.1.1.0

Example

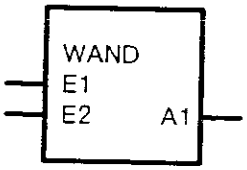
FBD/LD



IL

```
!BA    0
WAND
       MW 00,00
       KW 00,01
       MW 08,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
E2	E	W	N	P	N	0	0
A1	A	W	N	P	N	0	0

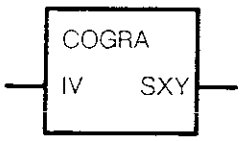
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WAND			
00002		PP 0	E1	Input WORD
00003		PP 0	E2	Input WORD
00004		PP 0	A1	Output WORD

This function block computes the directional component of the area center of gravity in the X direction (SX) or in the Y direction (SY).

This block is needed when using the video sensor OMS-F.

FBD/LD



IL

```
!BA     0
COGRA
        IV
        SXY
```

Parameters

IV	WORD	EW, AW, MW, KW	Number of the 35 IV 90 module
SXY	WORD	AW, MW	SX is the X component of the area center of gravity SY is the Y component of the area center of gravity

CE Data

Runtime:	
Basic runtime:	154 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block computes the directional component of the area center of gravity in the X direction (SX) or in the Y direction (SY).

To do this, the block reads the area moment of the first order (MX or MY) and the area of the measured object from the iconic processor 35 IV 90. The measured object must be in measurement window 1!

Two measurements are needed to define the center of gravity, one for the X component (SX) and one for the Y component (SY). A different setting of the mode register (SETMOD) is needed in each case to measure SX and SY.

To do this, the directional components SX and SY are not calculated in a real dimension, but as "quantities of pixels". That is to say, SX is the number of pixels in the X direction from the zero point to the center of gravity. SY is the number of pixels in the Y direction from the zero point to the center of gravity. Zero point is located in the top left corner of the picture. If the directional components of the center of gravity (SX and SY) are needed as real units of measure (e. g. mm), they can be then calculated to these using the UMIMA block.

Attention must also be paid to ensuring that the "histogram analysis" mode in the mode register is deactivated and that the "MX determination" or "MY determination" mode is activated (see also SETMOD block). The mode register must be set at least 20 ms before calling the COGRA block, i.e. in the previous program cycle.

IV WORD

Number of the 35 IV 90 module.

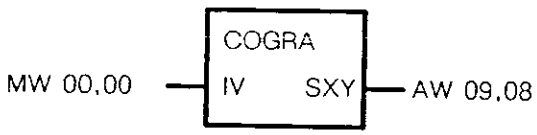
The number of the required 35 IV 90 module with which measurement is to take place is specified with the operand at this input.
Value range: 0 ... 5.

SXY WORD

SX is the X component of the area center of gravity as the number of pixels between the zero point and the center of gravity. SY is the Y component of the area center of gravity as the number of pixels between the zero point and the center of gravity.

Example

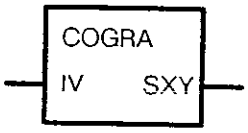
FBD/LD



IL

```
!BA     0
COGRA

        MW 00,00
        AW  09,08
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
IV	E	W	N	P	Y	0	0
SXY	A	W	N	P	Y	0	0

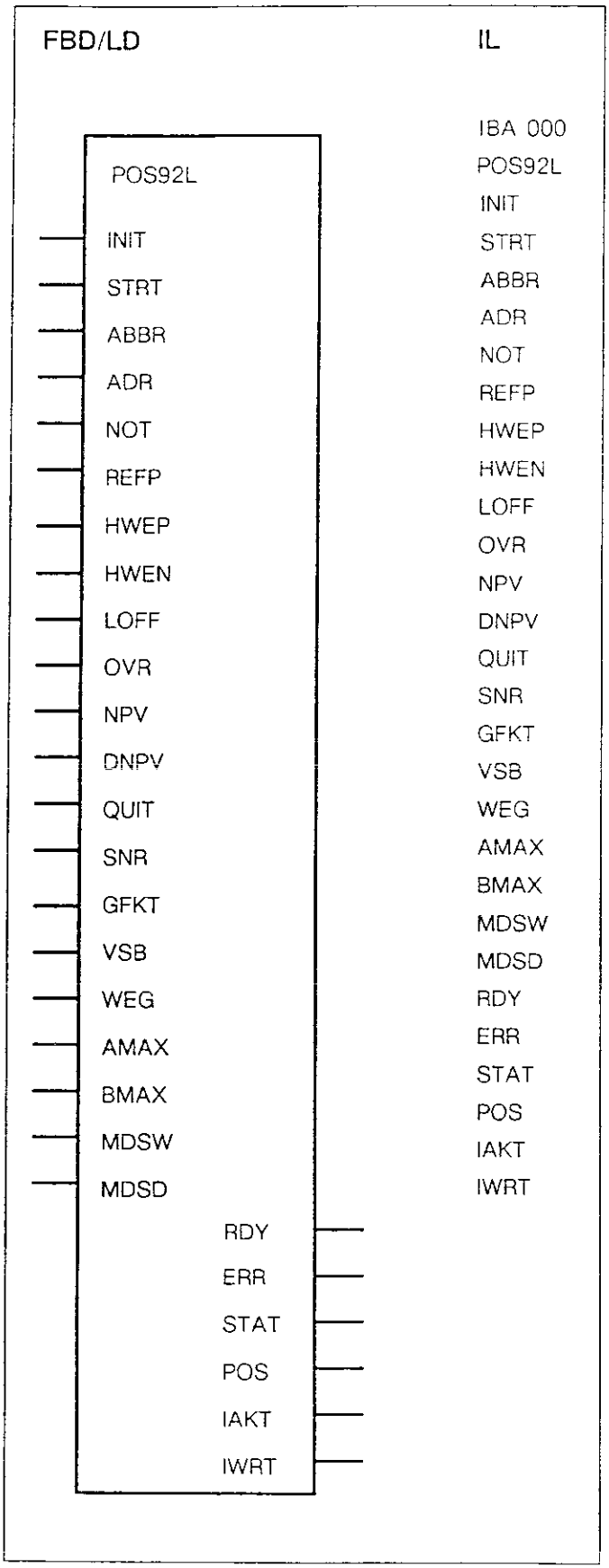
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	COGRA			
00002		PP 0	IV	Input WORD (IV No.)
00003		PP 0	SXY	Output WORD (X/Y components of the area center of gravity)

The two function blocks **POS92L** and **POS92S** serve the purpose of positioning and position control of axes.

They differ only by virtue of the method of ramp calculation and this is the reason why this description applies to both. The two function blocks are therefore referred to as POS92 in the following description. The function block POS92L is used in the examples.

The acceleration and braking ramps can be adjusted separately for each positioning sentence. In the case of the POS92L, braking and acceleration take place along a linear ramp and, in the case of the POS92S, along a $\sin^2$ ramp. The function block needs the position control clock for internal calculations. To do this, it directly accesses the planned cycle time (KD00.00) in the operand memory.



907 PC 32/ABB Procontic T300/Issued: 06.91

Parameters

INIT	BINARY	E, A, M, K, S	Enable of initialization
STRT	BINARY	E, A, M, K, S	Start, job assignment to the function block
ABBR	BINARY	E, A, M, K, S	Abort a running job
ADR	#	#, #H	Address of the axis board 35 AE 92
NOT	BINARY	E, A, M, K, S	Emergency stop switch (active low)
REFP	BINARY	E, A, M, K, S	Reference point cam (active high)
HWEP	BINARY	E, A, M, K, S	Hardware limit switch, positive (active low)
HWEN	BINARY	E, A, M, K, S	Hardware limit switch, negative (active low)
LOFF	BINARY	E, A, M, K, S	Open position control loop
OVR	WORD	EW, AW, MW, KW	Feed override
NPV	BINARY	E, A, M, K, S	Datum offset
DNPV	DOUBLE WORD	MD, KD	Delta datum offset
QUIT	BINARY	E, A, M, K, S	Error acknowledgement input
SNR	WORD	EW, AW, MW, KW	Sentence number
GFKT	WORD	EW, AW, MW, KW	List start for G functions
VSB	WORD	EW, AW, MW, KW	List start for feed rates
WEG	DOUBLE WORD	MD, KD	List start for travel distances
AMAX	WORD	EW, AW, MW, KW	List start for maximum acceleration values
BMAX	WORD	EW, AW, MW, KW	List start for maximum deceleration values
MDSW	WORD	EW, AW, MW, KW	List start for machine data record words
MDSD	DOUBLE WORD	MD, KD	List start for machine data record double words
RDY	BINARY	A, M, K	Position control loop closed; function block successfully initialized; function block ready
ERR	BINARY	A, M, K	Error
STAT	WORD	AW, MW, KW	Error status
POS	BINARY	A, M, K	Position reached
IAKT	BINARY	A, M, K	Axis inactive (standstill)
IWRT	DOUBLE WORD	MD, KD	Actual position value

CE Data

Runtime:	
Basic runtime:	POS92L 0.85 ... 0.90 ms POS92S 0.90 ... 1.05 ms
Additional runtime:	—
Output updating:	not applicable
Number of previous values:	76 words
Available as of:	ABB Procontic T300 (PLC) V8.0 / 35 ZE 94 R0101

Description

The two function blocks **POS92L** and **POS92S** serve the purpose of positioning and position control of axes.

They differ only by virtue of the method of ramp calculation and this is the reason why this description applies to both. The two function blocks are therefore referred to as POS92 in the following description. The function block POS92L is used in the examples.

The acceleration and braking ramps can be adjusted separately for each positioning sentence. In the case of the POS92L, braking and acceleration take place along a linear ramp and, in the case of the POS92S, along a $\sin^2$ ramp. The function block needs the position control clock for internal calculations. To do this, it directly accesses the planned cycle time (KD00.00) in the operand memory.

INIT BINARY
Initialization of the position control loop

After the function block POS92 is called, the position control loop remains open until a 0→1 edge is detected at the INIT input. As the result of this, the machine data record (containing all data of the controlled system relating to the control dynamics) is read and plausibility of the data is checked.

If the machine data contains no errors, the binary outputs RDY, POS and IAKT are set to 1. The error flag ERR (binary) and the error status are set to 0.

A reference point or positioning travel can be assigned to the function block in this state.

If the machine data contains an error, ERR is set to 1 and STAT contains the error status message 'MDS error'. Inadmissible values in the machine data record are overwritten by defaults. A START job is then possible, but is not expedient.

The function block is only operable if the INIT input and the RDY output have a 1 signal.
A 0 signal means that the position control loop is open.

STRT BINARY
Job assignment to the function block

0→1 edge:
The sentence whose number is found at SNR (sentence number) is started or continued.

The following applies:
SNR = 0: no job; the 'Job not allowed' message appears at the STAT.output; ERR = 1.

SNR = −1: Start or continuation job for reference point travel (if the job had not previously been stopped, this is a start job); outputs IAKT and POS are set to 0 if the position setpoint does not agree with the actual position.

SNR > 0: Start or continuation job for positioning travel; outputs IAKT and POS are set to 0 if the position setpoint does not agree with the actual position. A start or continuation job is only possible if the RDY output is 1.

1→0 edge:
Start or abort a running job whereby:

SNR = 0: Abort running job; outputs IAKT and POS are set to 1; then only a start job is possible.

SNR = previous SNR:
Stop job for the running job (reference point travel or positioning travel). The axis is stopped and the output IAKT is set to 1. POS output remains 0 if the position setpoint has not yet been reached at the time of stopping.

After a stop job, only a continuation job or aborting via the ABBR input (0→1 edge at STRT) is possible.

Status 0:
no job

Status 1:
job running;

The outputs IAKT and POS are set to 1 when the job has been concluded correctly.

Assignment of a new positioning sentence (i.e. 0→1 edge at START and new SNR<>previous SNR) is only allowed if the previous sentence has been run completely or aborted. If this is not the case, a 1 signal appears at the ERR output and an error message at STAT. No start or continuation job is possible either if emergency stop is actuated.

ABBR BINARY
Abort a running job

Status 0: Jobs are executed as normally.

0→1 edge: Running job is aborted; this is the same function as a 0→1 edge at START when sentence number 0 is present: The axis is stopped (along a braking ramp) and the outputs IAKT and POS become 1.

1→0 edge: The function block is then ready for a start job. No job assignment to the function block is possible as long as the ABBR input remains 1. The message 'Job not allowed' appears at STAT if an attempt is made to assign an abort job without a job running at the time.

ADR DIRECT CONSTANT
Channel address of the axis board 35 AE 92

(EXXX or FXXX, adjustable with S1 on the axis board).

The function block POS92L/POS92S accesses the registers of the axis board.

NOT BINARY
Emergency stop switch, active low

Status 0: Start or continuation job is allowed.

Status 1: a) When a job is running (axis is travelling): Abrupt braking of the axis (no braking ramp) and opening of the position control loop. A start or continuation job is not allowed. RDY and POS are set to 0 and IAKT and ERR are set to 1. The message 'Emergency stop actuated' appears at STAT. Reinitialization is necessary before a new job can be started.

Explanation: When Emergency Stop is triggered in a plant, the drive amplifiers are normally also disconnected from the power. To ensure that the axis will not start to move with a still pending setpoint when the malfunction is remedied (i.e. when the voltage is reactivated), the position control loop is open when Emergency Stop is actuated. This guarantees orderly start-up on reactivation.

b) The position control loop remains closed. RDY, POS, IAKT and ERR remain set to 1. The message 'Emergency stop actuated' appears at STAT. The position control loop is opened – analogously to a) – if an attempt is made to assign a job in this state. Reinitialization is then necessary.

REFP BINARY
Reference point cam, active high

Status 0: The reference switch (cam) has been actuated.

Status 1: The reference switch (cam) has not been actuated.

HWEP BINARY
Hardware limit switch in positive direction, active low

This switch is checked when travelling in the positive direction. The axis is stopped without a braking ramp if it is actuated (i.e. 0 at the input HWEP). The position control loop is opened.
Contact with a hardware limit switch, just like Emergency Stop, is treated as a fatal error. Reinitialization is necessary before a new job can be assigned.

HWEN BINARY
Hardware limit switch in negative direction, active low

Functions analogous to HWEP.

LOFF BINARY
Open position control loop

Status 0: Position control loop is closed, controlling to position setpoint.

0→1 edge: The axis is braked abruptly and the position control loop is opened;
Explanation: In this way, the axis can be moved by external influencing, whereby the setpoint and actual value are updated continuously (internally, the setpoint is continuously set equal to the actual value).

1→0 edge: The axis is abruptly accelerated to the speed that was present before the 0→1 edge at LOFF. The current positioning sentence is continued.

OVR WORD
Feed rate override

The input is only active during positioning movements with appropriately programmed G functions. A feed rate override is generally inactive for movements to the reference point.

OVR is specified in 0.5 % steps (of the programmed feed rate setpoint). The programmed feed rate is multiplied with this value, thus enabling modification of the travel speed during positioning travel. The new speed setpoint then runs along a ramp.

NPV BINARY
Datum offset

Every time a 0→1 edge appears at NPV, the word present at the word input DNPV is added, with the correct sign, to the function block's output IWRT (actual position value).

Explanation: By means of the datum offset, it is possible to program travels relative to a workpiece datum, for instance, instead of the machine datum.

DNPV DOUBLE WORD
Delta datum offset

Value by which the datum is to be shifted (cf. NPV). It is specified in increments.

QUIT BINARY
Error acknowledgement input

Errors reported at ERR and STAT are acknowledged with a 0→1 edge at QUIT. ERR remains 1 until the error message has been acknowledged.

SNR WORD
Sentence number

The various positioning sentences are selected by way of the sentence number:

SNR = −1: Reference point travel = first sentence in the positioning sentence list

SNR = 0: Abort job (with a simultaneous 1→0 edge at STRT), no input in the positioning sentence list; this refers to the currently running positioning sentence

SNR n > 0: nth positioning sentence, entry number n+1 in the positioning sentence list

Positioning sentence lists

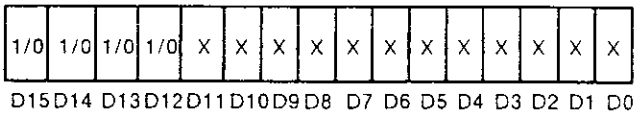
A positioning sentence list contains the travel information for one positioning job. It consists of:
- G function
- feed rate
- distance
- maximum permissible acceleration and
- maximum permissible deceleration

The positioning sentences are stored in five lists described below. The data at the start of a list is intended in each case for reference point travel.

GFKT WORD
List start for G functions

The operand placed at the GFKT input must contain the G function for reference point travel. The G functions for n programmed positioning sentences must then be stored in the n following word operands.

Meanings of the individual bits in the G function bit pattern:



D15: 1: Relative distance information, incremental dimension
0: Absolute distance information

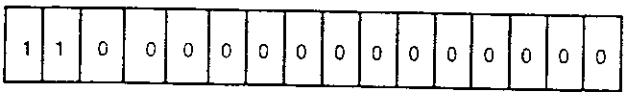
D14: 1: Feed rate override is inactive
0: Feed rate override is active.
This bit is only significant for positioning travel; during reference point travel the override is generally inactive.

D13: 1: The positioning sentence is executed even if the programmed distance exceeds the software limit switch. The travel distance is limited to the software limit switch.
No error message is issued.
0: The positioning sentence is not accepted if the programmed distance exceeds the software limit switch.
The error message 'Software limit switch exceeded' appears.
The positioning sentence cannot be started.

D12: 1: Reference point travel without reversal
0: Reference point travel with reversal.
This bit is only relevant for the first sentence in the positioning sentence list (= sentence for reference point travel).

D11 – D0: Don't care (bits can be occupied at will)

Example: G function $C000_H$



means:
- Relative distance information
- Override inactive
- The positioning sentence is not accepted if the programmed distance exceeds a software limit switch (with error message)
- Reference point travel with reversal

VSB WORD
List start for feed rates

The operand applied to the VSB input must contain the feed rate setpoint for reference point travel. The speed setpoints for n programmed positioning sentences must then be stored in the n following word operands.

When the corresponding positioning sentence is assigned, the feed rate values are limited to the maximum speed programmed in the machine data record. Reaching of the maximum speed is a fatal error, i.e. the axis is abruptly braked (IAKT, ERR are 1 and STAT contains the message 'Maximum feed rate'). Reinitialization is necessary after this error because RDY is set to 0.

If a feed rate of 0 mm/min is specified for a positioning travel, the applicable sentence cannot be started (RDY = 0, ERR = 1, STAT = 'Feed rate=0'). In the case of reference point travel, a start job is possible in this case. The job is executed with a speed of 1 mm/min.

Example: Feed rate $A00_H = 2560_{DEC}$ means:
travel with a speed of
2560 mm/min

WEG DOUBLE WORD
List start for travel distances

The double word operand applied to the WEG input must contain the travel distance for reference point travel. The distance setpoints for n programmed positioning sentences must then be stored in the n following double word operands.

Example: Distance $F6FA_H = -2310_{DEC}$ means:
When the distance is specified as a relative parameter: Travel from the current position over a distance of 2310 increments in the negative direction.

When the distance is specified as an absolute parameter: Travel from the current actual position to the absolute position -2310 (travel distance = position setpoint [incr] - actual position [incr])

AMAX WORD
List start for maximum acceleration values

The operand applied to the AMAX input must contain the maximum permissible acceleration for reference point travel. The maximum acceleration values for n programmed positioning sentences must then be stored in the n following word operands. AMAX is specified in units of 1/256 m/s^2.

Example: AMAX $100_H = 256_{DEC}$ means:
Acceleration must not exceed 1 m/s^2 when executing this positioning sentence.

BMAX WORD
List start for maximum deceleration values

The operand applied to the BMAX input must contain the maximum permissible deceleration for reference point travel. The maximum deceleration values for n programmed positioning sentences must then be stored in the n following word operands. BMAX is specified in units of 1/256 m/s^2.

Example: BMAX $100_H = 256_{DEC}$ means:
Deceleration must not exceed 1 m/s^2 when executing this positioning sentence.

Machine data record
The machine data record contains data specific to the drive. It consists of:
- Distance resolution
- Gain (KV value)
- Maximum permissible following error
- Control window
- Backlash compensation
- Drift compensation
- Maximum permissible feed rate (maximum speed)
- Positive software limit switch
- Negative software limit switch

The machine data record is stored in the **two** lists described below (MDSW, MDSD).

MDSW WORD
List start, machine data record, word operands
The list of the word operands contains:

1st word: Distance resolution,
smallest distance that can be specified; characteristic value of the incremental encoder (at the same time, the setting of the axis board, 1 to 4-fold evaluation, must be observed).
Example: WEGAUF = 64_H = 100_{DEC} means 1µm/incr

2nd word: Gain,
measure of the gain in the control loop, characteristic value of the control loop (P controller), specifying with what speed in m/min an axis can travel until a following error of 1 mm is reached.
Example: KVWER = 10_H = 16_{DEC} means a gain of 1

3rd word: Maximum following error,
maximum permissible following error of the axis in relation to the position calculated and output by the function block.
Example: SLPMAX = $C8_H$ = 200_{DEC} means 200 incr maximum following error

4th word: Control window,
control window defining quantity for the POS output, the 'Position reached' message is issued if the actual position deviates from the position setpoint only by the control window.
Example: WINDOW = 120_H = 288_{DEC} means a control window of 288 incr

5th word: Backlash compensation,
compensation of the drive system's backlash.
The most significant bit in this parameter contains the information which backlash is to be compensated:
a) D15 = 0: Backlash between the drive and moving element (drive and measurement system are connected to each other without backlash) or
b) D15 = 1: Backlash between the measurement system and moving element (drive and moving element are connected to each other without backlash)
Example: LOSEKP = FF_H = 255_{DEC} means a backlash compensation of 255 incr in the event of backlash between the drive and moving element.

6th word: Drift compensation,
offset with respect to the manipulated variable.
Drift compensation compensates for the steady-state control deviation resulting from non-linearities of the drive amplifier and axis board (temperature drift, aging etc.).

7th word: Maximum permissible feed rate,
limit of the drive system's component capable of withstanding the least mechanical load (exceeding of this feed rate/speed value shortens the lifetime).
Example: VSBMAX = 8000_H = 32768_{DEC} means a maximum feed rate of 32768 mm/min

MDSD DOUBLE WORD
List start, machine data record, double word operands

The list of the double word operands contains:

1st word: software limit switch, positive (SWEP)

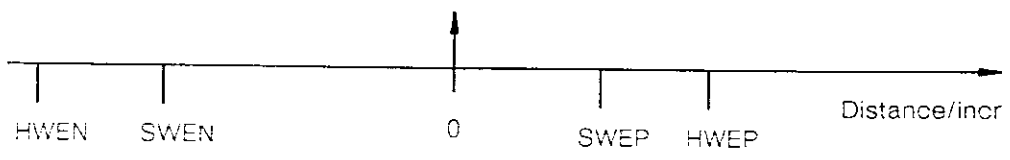
2nd word: software limit switch, negative (SWEN)

Example: SWEP = 3FFF $FFFE_H$ = +1 073 741 822_{DEC} means that the positive software limit switch is located at 1 073 741 822 incr.

SWEN = FFFF $FFFE_H$ = -2_{DEC} means that the negative software limit switch is located at -2 incr.

The G function is used to define (D13) how the software limit switches are handled. The two software limit switches must not overlap (this is evaluated during initialization as a fatal error). The two software limit switches may certainly have the same sign (e.g. both negative), but the following must apply in all cases: SWEN < SWEP.

Example for the geometric arrangement of software and hardware limit switches



Datum is determined by reference point travel

Restrictions/limits for the machine data record and positioning sentences

Data item	Limits	Multiplier for the numerical value of the operand
Distance resolution	0.25 µm/incr ≤ Dist. res. ≤ 25 µm/incr	1/100 µm/incr
KV value	1.0 ≤ KV value ≤ 200	1/16
Following error	0 incr ≤ Fol. err. max ≤ 16000 incr	1 incr
Incr control window	1 incr ≤ Window ≤ 65535	1 incr
Backlash compensation	0 ≤ Backl. comp. ≤ 255 incr	1 incr
Drift compensation	−127 incr ≤ Driftk c. ≤ 128 incr	1 incr
Maximum feed rate	1 mm/min ≤ fd. r. ≤ 64000 mm/min	1 mm/min
Software limit switch positive and negative	SWEN < SWEP $-(2^{30}-1) \leq$ SWEN/SWEP $\leq +(2^{30}-1)$	1 incr
GFKT	---	1
VSB	1 mm/min ≤ VSB ≤ max. feed rate	1 mm/min
WEG	Software limit switches	1 incr
AMAX	1/256 m/s² ≤ AMAX ≤ 255 m/s²	1/256 m/s²
BMAX	1/256 m/s² ≤ BMAX ≤ 255 m/s²	1/256 m/s²

After initialization, an error message appears at ERR and STAT if these limits of the machine data record and positioning data sentences are not obeyed. The data containing errors is overwritten with the affiliated defaults (cf. INIT input).

RDY BINARY
Position control loop closed

Status 1: Initialization of the function block has been concluded successfully and no fatal error has occurred. In this state, the position control loop is closed, i.e. the function block is controlling to the current speed setpoint. It is ready for job assignment.
Status 0: The position control loop is open. It is not possible to assign a job to the function block (internal blocking). No error message appears.

Before initialization (0→1 edge at INIT) and after detection of a fatal error, RDY is 0.

ERR BINARY
Error message

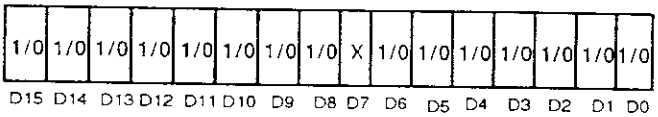
Status 0: No error has occurred.
Status 1: The function block has detected an error. The STAT output indicates what kind of error has occurred. ERR remains 1 until the error has been acknowledged (0→1 edge at QUIT).

ERR is a pure signalling output, i.e. its status does not influence any other input or output.

STAT WORD
Error status

Status 0: No error has occurred.
Status > 0: The function block has detected an error. The STAT word is an error bit pattern. If several errors occur simultaneously, the total of the individual significances results in the contents of STAT.

Meanings of the individual bits in the error bit pattern:



D15: Internal error (programming error)
D14: Feed rate is 0
D13: Error in the machine data record
D12: Job is inadmissible
D11: Programmed distance exceeds negative software limit switch
D10: Programmed distance exceeds positive software limit switch
D 9: Negative hardware limit switch has responded

D 8: Positive hardware limit switch has responded
D 7: Not used
D 6: Programmed maximum speed has been exceeded
D 5: Emergency Stop has been actuated
D 4: Following error is more than the programmed maximum following error
D 3: Axis board's watchdog has responded
D 2: UA1: broken wire
D 1: UA2: broken wire
D 0: Encoder malfunction

Fatal errors are those which result in opening of the position control loop (RDY = 0). After occurring of a fatal error new initialization is necessary.

Summary of possible fatal errors:

- Specification of software limit switches that overlap (SWEP<SWEN)
- Specification of an inadmissible distance resolution
- Specification of a 0 mm/min feed rate for positioning travel with a distance <> 0 incr
- Attainment of the maximum speed preset (VSBMAX)
- Emergency Stop actuation during a running job
- Contact with the positive or negative hardware limit switch
- Encoder malfunction (reported by the axis board)
- UA1: broken wire (reported by the axis board)
- UA2: broken wire (reported by the axis board)
- Watchdog of the axis board has responded (reported by the axis board)
- Internal program error (invalid program status pointer)

POS BINARY
Position reached message

Status 0: The axis is not in the currently programmed position setpoint, i.e. the last positioning job has not yet been executed.
Status 1: The axis has reached the position setpoint. The function block is ready for a new positioning job. (After completion or aborting of a positioning job.)

IAKT BINARY
Axis at standstill message

Status 0: The axis is travelling.
Status 1: The axis is at standstill (speed=0).

IWRT DOUBLE WORD
Actual position value

The double word operand contains the current actual position value with respect to the defined zero point determined by a reference point travel (and by zero shift).

Speed/time diagrams

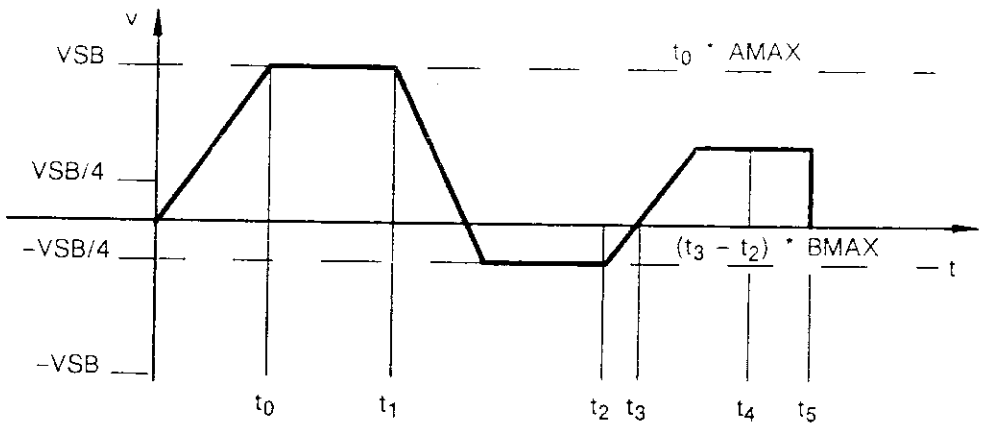
Conventions:
All diagrams have been drawn for positive distance parameters and linear ramps. For a negative distance the speeds would be negated. In the case of $\sin^2$ ramps, the diagrams are basically the same (see example 9).

VSB: programmed feed rate setpoint for running job

Note:
If, during a positioning job, the remaining distance criterion (i.e. the remaining travel distance is equal to the distance need for braking to feed rate 0 at BMAX) or the abort criterion (during reference travel, for instance, travel over the reference point cam) has already been fulfilled when the programmed feed rate is reached, the acceleration ramp is aborted and a deceleration ramp is initiated.

1 Reference point travel

1.1 Reference point travel with reversal



t<0: No job (RDY, POS and IAKT are 1)

t=0: Start job for reference point travel is assigned
(0→1 edge at STRT, IAKT and POS change to 0)

t_0: Speed setpoint has been reached

t_1: Reference point cam has been reached (REFP=1)

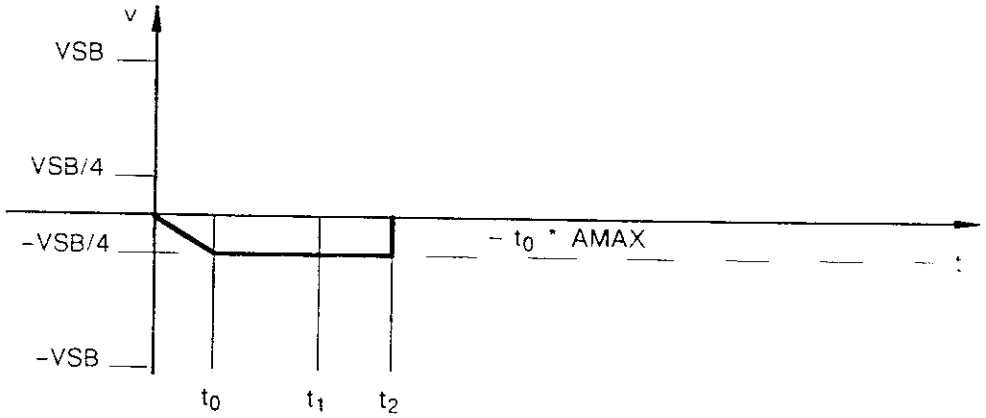
t_2: Axis leaves reference point cam in the reverse direction (REFP=0)

t_3: Speed of 0 m/s reached

t_4: Reference point cam reached again (REFP=1)

t_5: Encoder zero mark is detected and reference point cam is still 1:
speed abruptly becomes 0 (RDY, POS and IAKT are 1)

1.2 Reference point travel without reversal



t<0: No job (RDY, POS and IAKT are 1)

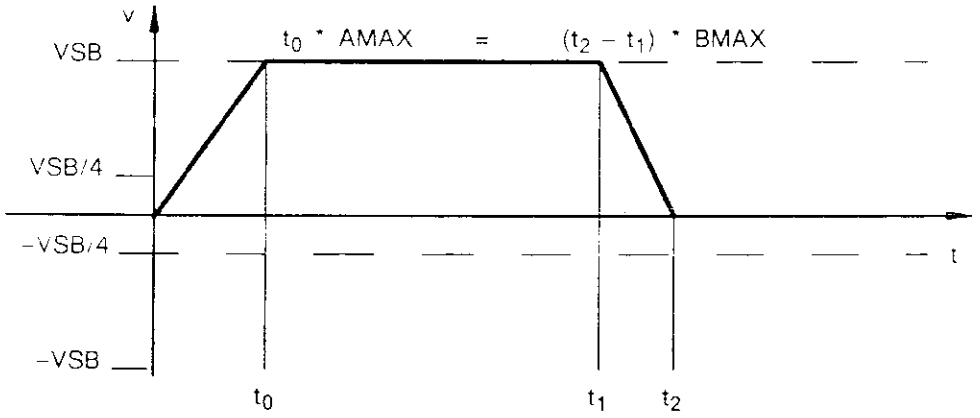
t=0: Start job for reference point travel is assigned
(0→1 edge at STRT, IAKT and POS change to 0)

t_0: -1/4 of the programmed speed setpoint has been reached

t_1: Reference point cam has been reached (REFP=1)

t_2: Encoder zero mark is detected and reference point cam is still 1:
speed abruptly becomes 0 (RDY, POS and IAKT are 1)

2 Positioning travel



t<0: No job (RDY, POS and IAKT are 1)

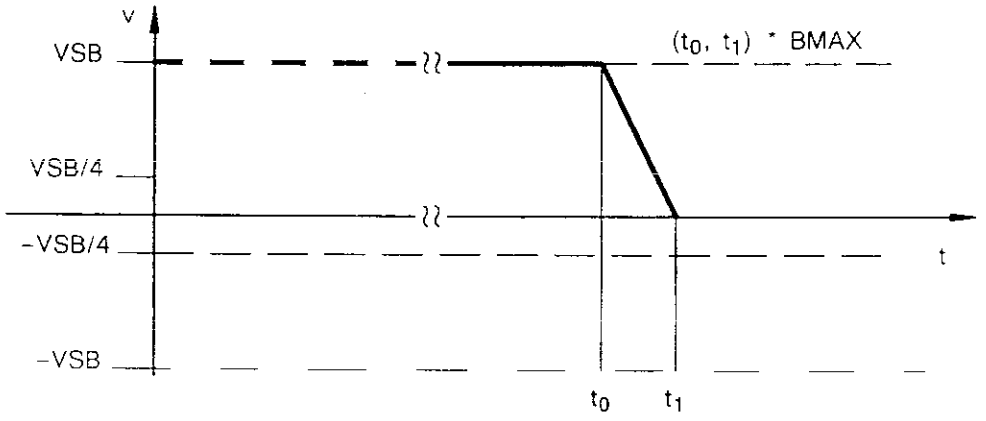
t=0: Start job for positioning travel is assigned
(0→1 edge at STRT, IAKT and POS change to 0)

t_0: Programmed speed setpoint has been reached

t_1: Remaining distance criterion is fulfilled (initiation of the braking ramp)

t_2: Feed rate = 0, position setpoint has been reached (RDY, POS and IAKT are 1)

3 Stopping a running job

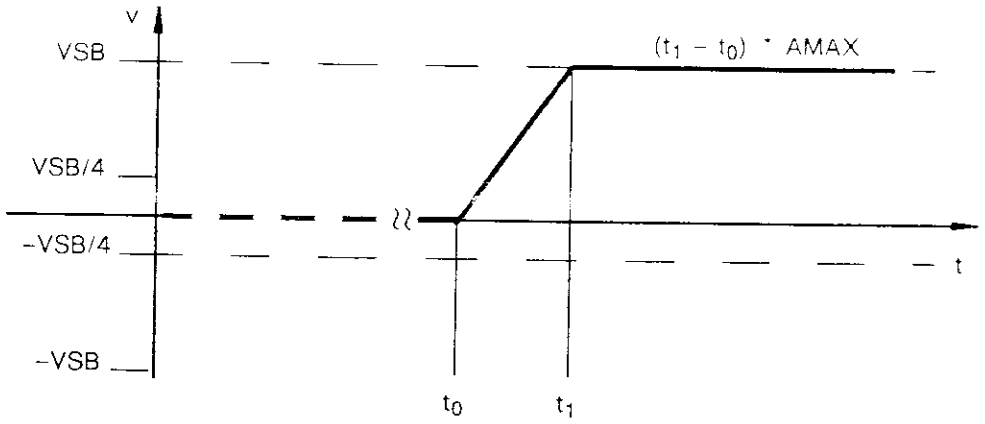


t<t_0: Positioning job (positioning and reference point travel) running
(1 at STRT; IAKT and POS are 0)

t_0: Stop job is assigned (1→0 edge at START, SNR unchanged)
Initiation of a braking ramp

t_1: Feed rate = 0 (RDY, POS and IAKT are 1)

4 Continuing a stopped job



$t<t_0$: Job is stopped (axis at standstill, RDY and IAKT are 1, POS is 0)

t_0: Continuation job is assigned (0→1 edge at START, SNR unchanged)
Start of an acceleration ramp (POS and IAKT change to 0)

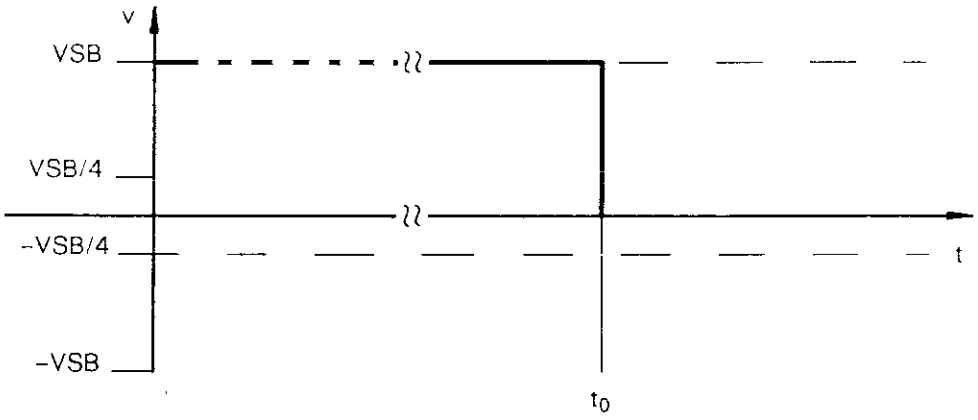
t_1: Feed rate = programmed feed rate setpoint (POS and IAKT are 0 until the position setpoint is reached)

5 Aborting a running job
This is analogous to stopping a running job:

The abort job is assigned at the time t_0. Only a start job is possible after this. Difference between abort and stop job:
If a continuation job is assigned after a stop job, the started positioning sentence is completed. If, however, a start job is assigned after an abort job and the sentence number is unchanged, the aborted sentence is begun from the start. However, this difference manifests itself only when working with relative distance parameters. When working with absolute distance parameters, the axis moves to the same end position in both cases.

6 Opening the position control loop

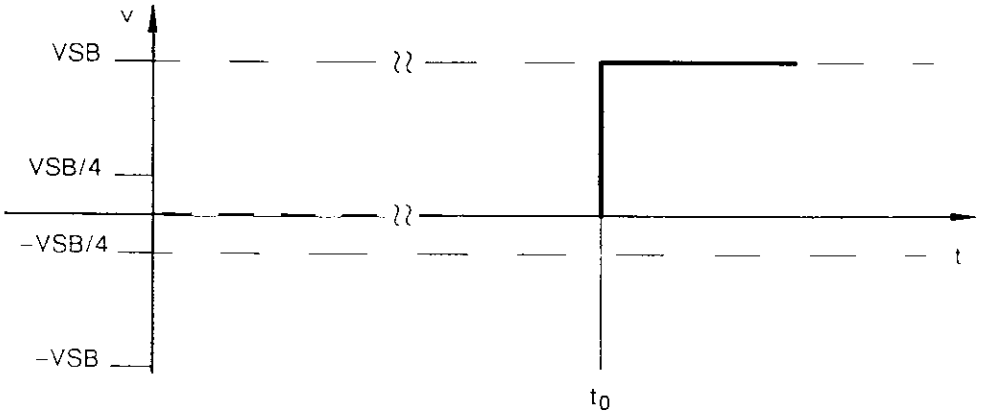
6.1 Opening the position control loop during a running job
(0→1 edge at LOFF)



$t<t_0$: Positioning job (positioning or reference point travel) running
(1 at STRT; IAKT and POS are 0)

t_0: Opening of the position control loop (1→0 edge at LOFF, SNR unchanged), abrupt braking to speed 0 (RDY, POS and IAK are unchanged)

6.2 Again closing an open position control loop
(1→0 edge at LOFF)



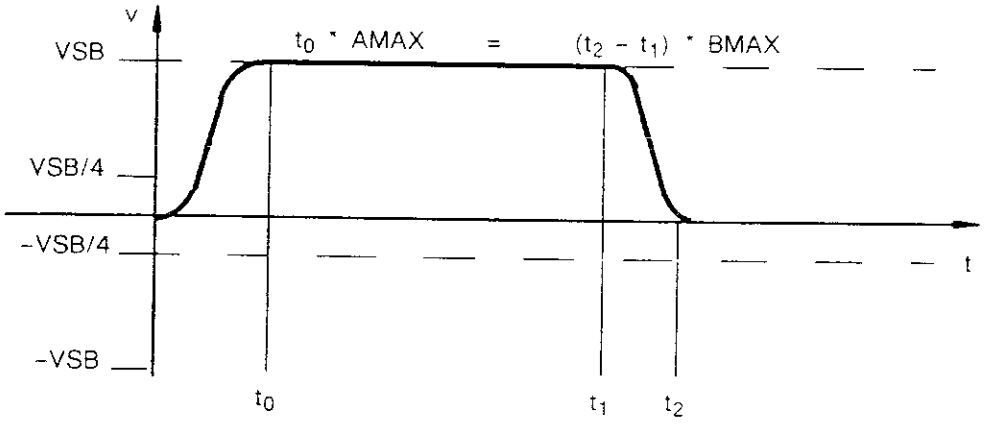
$t<t_0$: Position control loop is open (axis at standstill, RDY is 1)

t_0: Positioning job (positioning or reference point travel) continues, abrupt acceleration to the speed at which the position control loop has been opened (1 at STRT; IAKT and POS are 0)

7 Contact with a hardware limit switch analogous to opening the position control loop; abrupt braking to speed 0; RDY becomes 0 (fatal error); ERR, POS and IAKT are 1 and STAT is set to the corresponding error message.

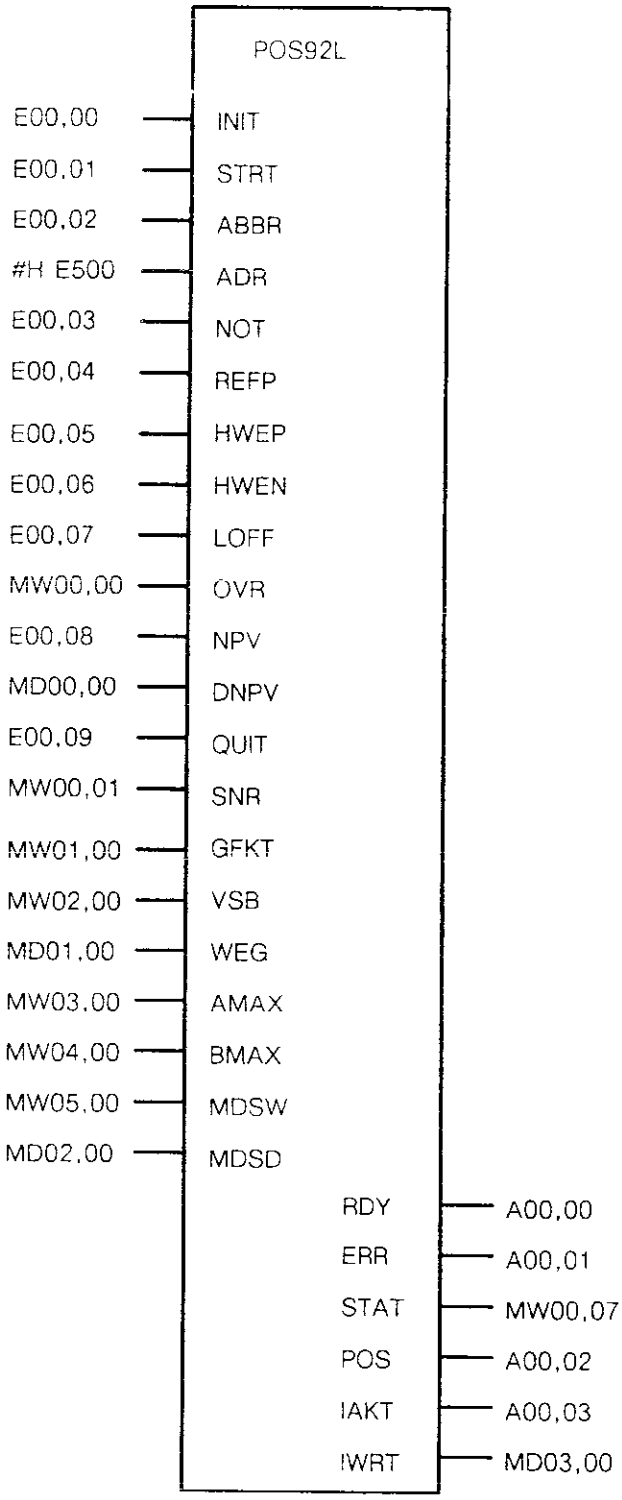
8 Actuation of Emergeny Stop during a job or occurrence of a fatal error analogous to opening the position control loop; abrupt braking to speed 0; RDY becomes 0 (fatal error); ERR, POS and IAKT are 1 and STAT is set to the corresponding error message.

9 Example of a speed/time daigram with sin^2 ramps (positioning travel)



Example

FBD/LD



IL

```
!BA     0
POS92L
        E00,00
        E00,01
        E00,02
        #H E500
        E00,03
        E00,04
        E00,05
        E00,06
        E00,07
        MW00,00
        E00,08
        MD00,00
        E00,09
        MW00,01
        MW01,00
        MW02,00
        MD01,00
        MW03,00
        MW04,00
        MW05,00
        MD02,00
        A00,00
        A00,01
        MW00,07
        A00,02
        A00,03
        MD03,00
```

The variables for the positioning sentences can be entered in the variable or symbol editor. Besides containing the operands for the function block's inputs/outputs, the following variable list also contains the absolute and symbolic variable names for reference point travel and for positioning travel movements:

Variable list

Variable	Symbol	Long text
E00,00	INIT	Enable of initialization
E00,01	STRT	Start, job assignment to the function block
E00,02	ABBR	Abort a running job
E00,03	NOT	Emergency stop switch (active low)
E00,04	REFP	Reference point cam (active high)
E00,05	HWEP	Hardware limit switch, positive (active low)
E00,06	HWEN	Hardware limit switch, negative (active low)
E00,07	LOFF	Open position control loop
E00,08	NPV	Datum offset
E00,09	QUIT	Error acknowledgement input
A00,00	RDY	Position control loop closed
A00,01	ERR	Error message
A00,02	POS	Position reached message
A00,03	IAKT	Axis standstill message
MW00,00	OVR	Feed rate override
MW00,01	SNR	Sentence number input
MW00,07	STAT	Error status output
MW01,00	GFKT	G function for reference point travel
MW01,01	GFKT1	G function for sentence number 1
MW01,02	GFKT2	G function for sentence number 2
MW01,03	GFKT3	G function for sentence number 3
MW01,04	GFKT4	G function for sentence number 4
MW02,00	VSB	Feed rate for reference point travel
MW02,01	VSB1	Feed rate for sentence number 1
MW02,02	VSB2	Feed rate for sentence number 2
MW02,03	VSB3	Feed rate for sentence number 3
MW02,04	VSB4	Feed rate for sentence number 4
MW03,00	AMAX	Maximum acceleration for reference point travel
MW03,01	AMAX1	Maximum acceleration for sentence number 1
MW03,02	AMAX2	Maximum acceleration for sentence number 2
MW03,03	AMAX3	Maximum acceleration for sentence number 3
MW03,04	AMAX4	Maximum acceleration for sentence number 4
MW04,00	BMAX	Maximum deceleration for reference point travel
MW04,01	BMAX1	Maximum deceleration for sentence number 1
MW04,02	BMAX2	Maximum deceleration for sentence number 2
MW04,03	BMAX3	Maximum deceleration for sentence number 3
MW04,04	BMAX4	Maximum deceleration for sentence number 4
MW05,00	MDSW	Distance resolution
MW05,01		Gain (KV value)
MW05,02		Following error
MW05,03		Control window
MW05,04		Backlash compensation
MW05,05		Drift compensation
MW05,06		Maximum permissible feed rate
#H E500	ADR	Channel address of axis board 35 AE 92
MD00,00	DNPV	Delta datum offset
MD01,00	WEG	Distance for reference point travel
MD01,01	WEG1	Distance for sentence number 1
MD01,02	WEG2	Distance for sentence number 2
MD01,03	WEG3	Distance for sentence number 3
MD01,04	WEG4	Distance for sentence number 4
MD02,00	MDSD	Software limit switch, positive
MD02,01		Software limit switch, negative
MD03,00	IWRT	Actual position value output

Input of variable values

There are basically **two** possibilities of entering the variable values in the PLC:

1. WRITE VARIABLES

Using the ABB Procontic programming system, values can be assigned to the variables in ONLINE mode. To do this, the variable names and the affiliated values are entered in the *short list* and transferred to the PLC.

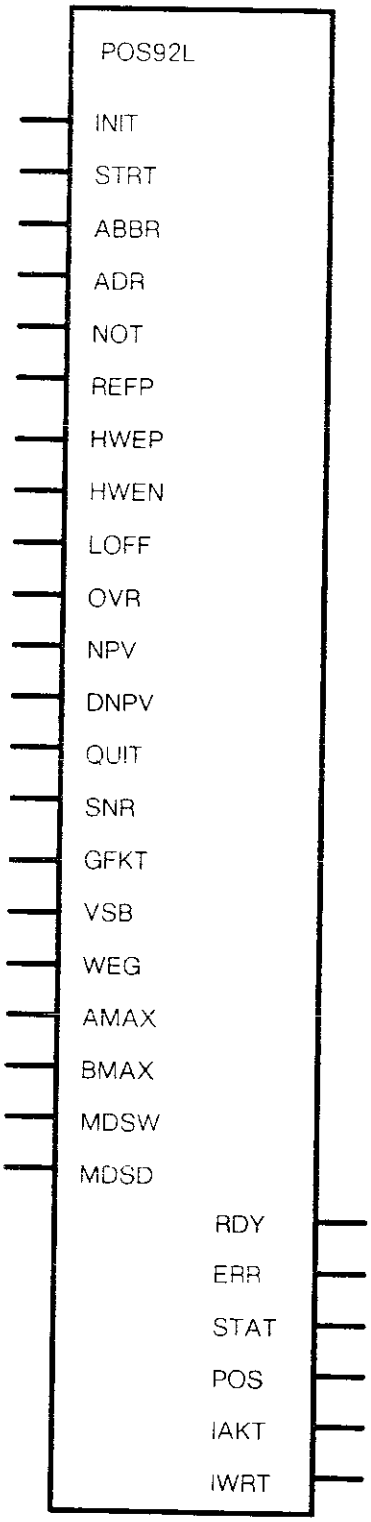
The short list's contents can additionally be stored in a file, thus being reproducible at all times. It is therefore possible to enter any number of positioning sentences and of storing them in corresponding short lists on disk.

This possibility of specifying variable values was used in the above example.

2. USE OF INDIRECT CONSTANTS

A further possibility of entering variable values consists of using direct constants in the variable or symbol list instead of flags. For indirect constants, the values can already be entered in the LONG TEXT column in the variable or symbol editor.

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
INIT	E	L	N	P	Y	0	0
STRT	E	L	N	P	Y	0	0
ABBR	E	L	N	P	Y	0	0
ADR	K	W	N	P	Y	0	0
NOT	E	L	N	P	Y	0	0
REFP	E	L	N	P	Y	0	0
HWEP	E	L	N	P	Y	0	0
HWEN	E	L	N	P	Y	0	0
LOFF	E	L	N	P	Y	0	0
OVR	E	W	N	P	Y	0	0
NPV	E	L	N	P	Y	0	0
DNPV	E	D	N	P	Y	0	0
QUIT	E	L	N	P	Y	0	0
SNR	E	W	N	P	Y	0	0
GFK	E	W	N	P	Y	0	0
VSB	E	W	N	P	Y	0	0
WEG	E	D	N	P	Y	0	0
AMAX	E	W	N	P	Y	0	0
BMAX	E	W	N	P	Y	0	0
MDSW	E	W	N	P	Y	0	0
MDSD	E	D	N	P	Y	0	0
RDY	A	L	N	P	Y	0	0
ERR	A	L	N	P	Y	0	0
STAT	A	W	N	P	Y	0	0
POS	A	L	N	P	Y	0	0
IAKT	A	L	N	P	Y	0	0
IWRT	A	D	N	P	Y	0	0

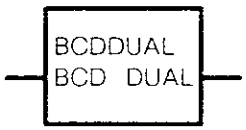
CE IL Definition

```
00000   !BA     0
00001   POS92L
00002           PP 0
00003           PP 0

00004           PP 0
00005           PP 0
00006           PP 0
00007           PP 0
00008           PP 0
00009           PP 0
00010           PP 0
00011           PP 0
00012           PP 0
00013           PP 0
00014           PP 0
00015           PP 0
00016           PP 0
00017           PP 0
00018           PP 0
00019           PP 0
00020           PP 0
00021           PP 0
00022           PP 0

00023           PP 0
00024           PP 0
00025           PP 0
00026           PP 0
00027           PP 0
00028           PP 0
```

Nr	Function block number (default 0)
INIT	Enable of initialization (BINARY)
STRT	Start, job assignment to the function block (BINARY)
ABBR	Abort a running job (BINARY)
ADR	Address of the axis board (DIR. CONST.)
NOT	Emergency stop switch (BINARY)
REFP	Reference point cam (BINARY)
HWEP	Hardware limit switch positive (BINARY)
HWEN	Hardware limit switch negative (BINARY)
LOFF	Open position control loop (BINARY)
OVR	Feed override (WORD)
NPV	Datum offset (BINARY)
DNPV	Delta datum offset (DOUBLE WORD)
QUIT	Error acknowledgement input (BINARY)
SNR	Sentence number (WORD)
GFKT	List start for G functions (WORD)
VSB	List start for feed rates (WORD)
WEG	List start for travel distances (DOUBLE WORD)
AMAX	List start for max. acceleration (WORD)
BMAX	List start for max. deceleration (WORD)
MDSW	List start for machine data record (WORD)
MDSD	List start for machine data record (DOUBLE WORD)
RDY	Position control loop closed (BINARY)
ERR	Error has occurred (BINARY)
STAT	Error status (WORD)
POS	Position reached (BINARY)
IAKT	Axis inactive (BINARY)
IWRT	Actual position value (DOUBLE WORD)

The positive BCD coded number at the input BCD is converted to a binary number and is allocated to the operand at the DUAL output.

FBD/LD



IL

```
!BA     0
BCDBIN
        BCD
        DUAL
```

Parameters

BCD	WORD	EW, MW, AW, KW	BCD-coded number
DUAL	WORD	AW,MW	Binary number

CE Data

Runtime:	
Basic runtime:	88 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

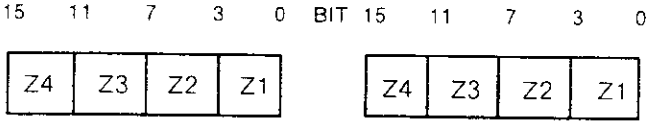
The positive BCD coded number at the input BCD is converted to a binary number and is allocated to the operand at the DUAL output.

The input and the output can neither be duplicated nor negated.

Definition:

The significance of the digits in a BCD coded number and a hexadecimal number is defined as follows:

BCD-NUMBER HEXDEC-NUMBER



Numerical value:

Z1 * 1
Z2 * 10
Z3 * 100
Z4 * 1000

$0 \leq Zi \leq 9$

Numerical value:

Z1 * 1
Z2 * 16
Z3 * 256
Z4 * 4096

$0 \leq Zi \leq F$

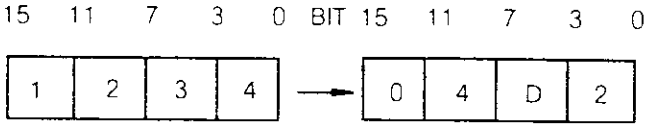
Note:

At the BCD input, the block additionally also accepts digits to which the following applies:

$0 \leq Zi \leq F$

Example 1

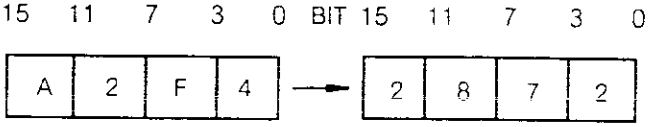
BCD NUMBER HEXDEC NUMBER



Z1 = 4 * 1 = 4	Z1 = 2 * 1 = 2
Z2 = 3 * 10 = 30	Z2 = 13 * 16 = 208
Z3 = 2 * 100 = 200	Z3 = 4 * 256 = 1024
Z4 = 1 * 1000 = 1000	Z4 = – * – =
+ 1234	+ 1234

Example 2

BCD NUMBER HEXDEC NUMBER



Z1 = 4 * 1 = 4	Z1 = 2 * 1 = 2
Z2 = 15 * 10 = 150	Z2 = 7 * 16 = 112
Z3 = 2 * 100 = 200	Z3 = 8 * 256 = 2048
Z4 = 10 * 1000 = 10000	Z4 = 2 * 4096 = 8192
+ 10354	+ 10354

Representation of a negative BCD number

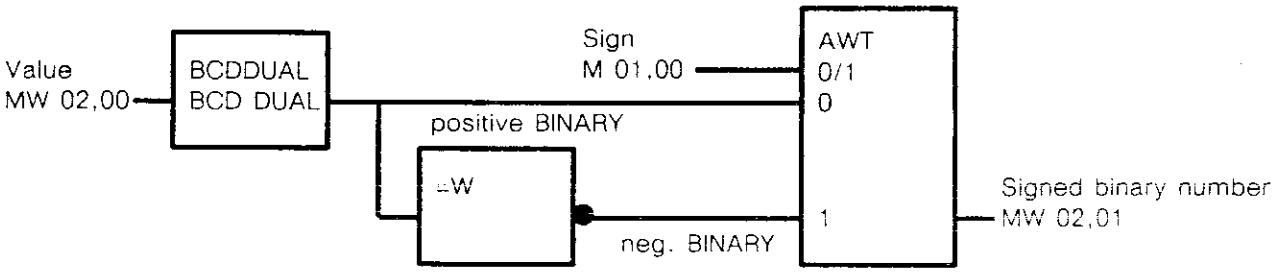
A negative BCD number can be represented in the PLC by *separate* representation of the value and the sign. In doing so, the value of the BCD number is stored in a word variable and the information about the sign is stored in a binary variable.

Example:

Representation of the BCD number -7:
MW 02,00 = 7 Value of the number, BCD coded
M 01,00 = 1 Sign of the number
- Positive: 0
- Negative: 1

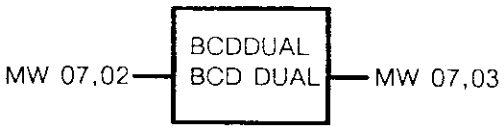
Conversion of a negative BCD number to a negative binary number

Example:



Example

FBD/LD



IL

```
!BA    0
BCDBIN
       MW 07,02
       MW 07,03
```

CE FBD Definition

Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
BCD	E	W	N	P	Y	0	0
DUAL	A	W	N	P	Y	0	0

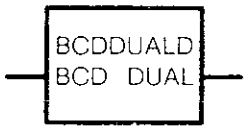
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	BCDBIN			
00002		PP 0	BCD	Input WORD (BCD number)
00003		PP 0	DUAL	Output WORD (binary number)

BCD TO BINARY CONVERSION, DOUBLE WORD — BCDDUALD

The positive BCD coded number at the input BCD is converted to a binary number and is allocated to the operand at the DUAL output.

FBD/LD



IL

```
!BA     0
BCDDW
        BCD
        DUAL
```

Parameters

BCD	DOUBLE WORD	MD, KD	BCD coded number
DUAL	DOUBLE WORD	MD	Binary number

CE Data

Runtime:	
Basic runtime:	310 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

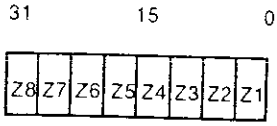
The positive BCD coded number at the input BCD is converted to a binary number and is allocated to the operand at the DUAL output.

The input and the output can neither be duplicated nor negated.

Definition

The significance of the digits in a BCD coded number and a hexadecimal number is defined as follows:

BINARY NUMBER

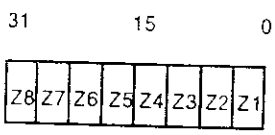


Numerical value:

Z1 * 1
Z2 * 10
Z3 * 100
Z4 * 1000
Z5 * 10000
Z6 * 100000
Z7 * 1000000
Z8 * 10000000

$0 \leq Zi \leq 9$

HEXDEC NUMBER



Numerical value:

Z1 * 1
Z2 * 16
Z3 * 256
Z4 * 4096
Z5 * 65536
Z6 * 1048576
Z7 * 16777216
Z8 * 268435456

$0 \leq Zi \leq F$

Note:

The block also accepts digits at the BCD input to which the following applies:

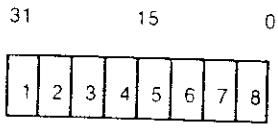
$0 \leq Zi \leq F$

Representation of a negative BCD number
See function block BCDDUAL

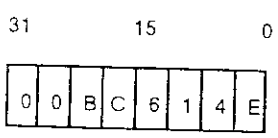
Conversion of a negative BCD number to a negative binary number
See function block BCDDUAL

Example 1:

BCD NUMBER



HEXDEC NUMBER

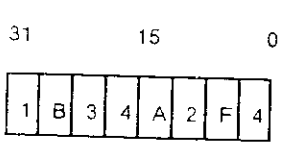


Z1 = 8 * 1	=	8
Z2 = 7 * 10	=	70
Z3 = 6 * 100	=	600
Z4 = 5 * 1000	=	5000
Z5 = 4 * 10000	=	40000
Z6 = 3 * 100000	=	300000
Z7 = 2 * 1000000	=	2000000
Z8 = 1 * 10000000	=	10000000
		- 12345678

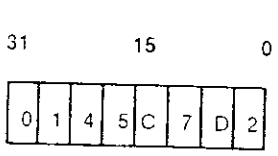
Z1 = 14 * 1	=	14
Z2 = 4 * 16	=	64
Z3 = 1 * 256	=	256
Z4 = 6 * 4096	=	24576
Z5 = 12 * 65536	=	786432
Z6 = 11 * 1048576	=	11534336
Z7 = 0 * 16777216	=	0
Z8 = 0 * 268435456	=	0
		+ 12345678

Example 2:

BCD NUMBER



HEXDEC NUMBER

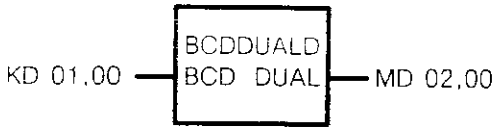


Z1 = 4 * 1	=	4
Z2 = 15 * 10	=	150
Z3 = 2 * 100	=	200
Z4 = 10 * 1000	=	10000
Z5 = 4 * 10000	=	40000
Z6 = 3 * 100000	=	300000
Z7 = 11 * 1000000	=	11000000
Z8 = 1 * 10000000	=	10000000
		+ 21350354

Z1 = 2 * 1	=	2
Z2 = 13 * 16	=	208
Z3 = 7 * 256	=	1792
Z4 = 12 * 4096	=	49152
Z5 = 5 * 65536	=	327680
Z6 = 4 * 1048576	=	4194304
Z7 = 1 * 16777216	=	16777216
Z8 = 0 * 268435456	=	0
		+ 21350354

Example

FBD/LD



IL

```
!BA     0
BCDDW
        KD 01,00
        MD 02,00
```

CE FBD Definition

Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
BCD	E	D	N	P	Y	0	0
DUAL	A	D	N	P	Y	0	0

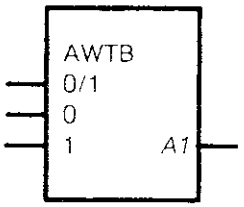
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	BCDDW			
00002		PP 0	BCD	Input DOUBLE WORD (BCD number)
00003		PP 0	DUAL	Output DOUBLE WORD (binary number)

A 0 signal at the binary input 0/1 allocates the status of the operand at the input 0 to the operand at the output A1.

A 1 signal at the binary input 0/1 allocates the status of the operand at the input 1 to the operand at the output A1.

FBD/LD



IL

```
!BA   0
AWTB
      0/1
      0
      1
      A1
```

Parameters

0/1	BINARY	E, M, A, S, K	Switchover input
0	BINARY	E, M, A, S, K	Binary input for 0/1 = 0
1	BINARY	E, M, A, S, K	Binary input for 0/1 = 1
A1	BINARY	A, M	Binary output

CE Data

Runtime:	
Basic runtime:	32 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

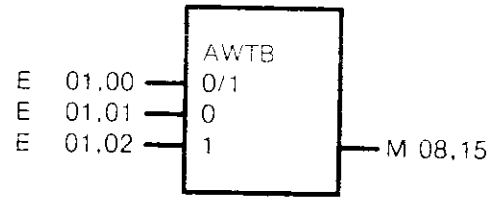
A 0 signal at the binary input 0/1 allocates the status of the operand at the input 0 to the operand at the output A1.

A 1 signal at the binary input 0/1 allocates the status of the operand at the input 1 to the operand at the output A1.

The inputs and the output can neither be duplicated nor inverted.

Example

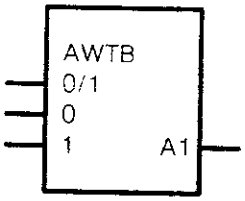
FBD/LD



IL

```
!BA    0
AWTB
       E  01.00
       E  01.01
       E  01.02
       M  08.15
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0/1	E	L	N	P	Y	0	0
0	E	L	N	P	Y	0	0
1	E	L	N	P	Y	0	0
A1	A	L	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	AWTB			
00002		PP 0	0/1	Input BINARY (Selector)
00003		PP 0	0	Input BINARY
00004		PP 0	1	Input BINARY
00005		PP 0	A1	Output BINARY

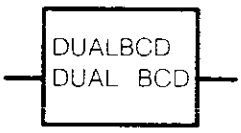
The binary number at the DUAL input is converted to a BCD coded number and is allocated to the operand at the output BCD.

The binary number is represented in 16 bits and must lie within the range

$0 \leq DUAL \leq 270FH$ (corresponding to BCD 9999).

The BCD number is limited to 9999 if it lies outside of this range. The BCD number is stored in a 16 bit word.

FBD/LD



IL

```
!BA     0
BINBCD
        DUAL
        BCD
```

Parameters

DUAL	WORD	EW, MW, AW, KW	Binary number
BCD	WORD	AW,MW	BCD coded number

CE Data

Runtime:	
Basic runtime:	Depending on the value: 28 µs (value = 0) ... 162 µs (value = 9999)
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The binary number at the DUAL input is converted to a BCD coded number and is allocated to the operand at the output BCD.

The binary number is represented in 16 bits and must lie within the range

$0 \leq$ DUAL $\leq$ 270FH (corresponding to BCD 9999).

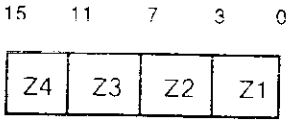
The BCD number is limited to 9999 if it lies outside this range. The BCD number is stored in a 16-bit word.

The input and the output can neither be duplicated nor negated.

Definition:

The significance of the digits in a hexidecimal number and a BCD coded number is defined as follows:

HEXDEC NUMBER

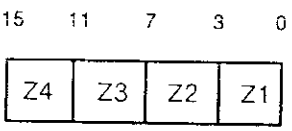


Numerical value:

Z1 · 1
Z2 · 16
Z3 · 256
Z4 · 4096

$0 \leq Zi \leq F$

BCD NUMBER



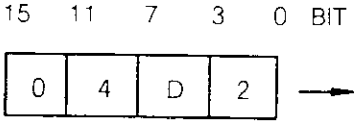
Numerical value:

Z1 · 1
Z2 · 10
Z3 · 100
Z4 · 1000

$0 \leq Zi \leq 9$

Example:

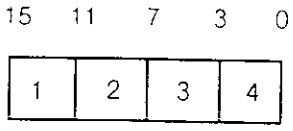
HEXDEC NUMBER



Z1 = 2 · 1 = 2
Z2 = 13 · 16 = 208
Z3 = 4 · 256 = 1024
Z4 = 0 · 4096 = 0

\+ 1234

BCD NUIMBER



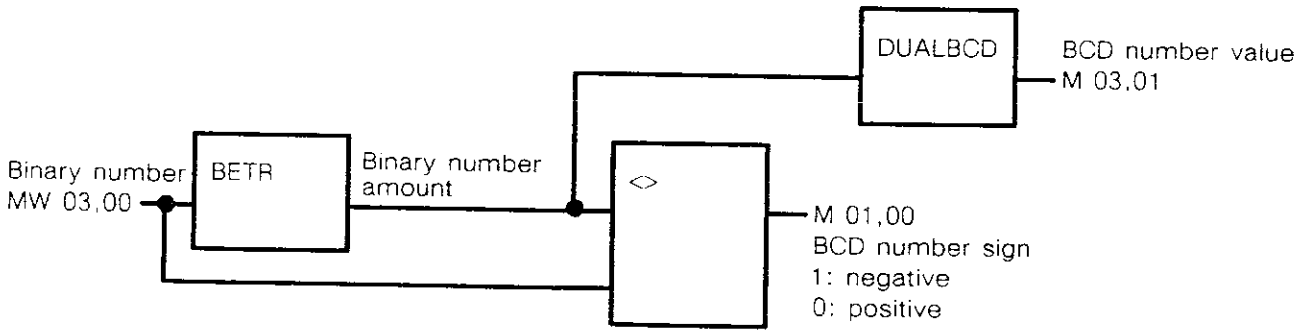
Z1 = 4 · 1 = 4
Z2 = 3 · 10 = 30
Z3 = 2 · 100 = 200
Z4 = 1 · 1000 = 1000

\+ 1234

Conversion of a negative binary number to a BCD number

A negative binary number with an amount less than 270FH can be converted to a BCD number, whereby the value and the sign of the BCD number are each stored in one flag.

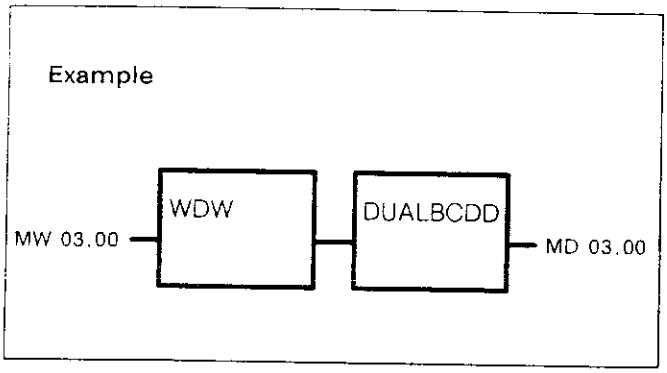
Example: Converting positive and negative binary numbers



Converting a binary number with an amount higher than 270FH

Binary numbers with an amount higher than 270FH are first of all converted to a double word (function block WDW). They are then converted from BINARY to BCD by means of the DUALBCDD function block.

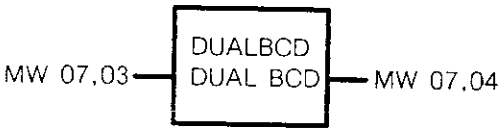
Example



The same procedure as above applies if a sign has to be taken into account.

Example

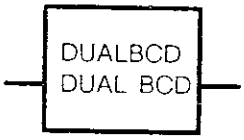
FBD/LD



IL

```
!BA    0
BINBCD
       MW 07,03
       MW 07,04
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
DUAL	E	W	N	P	Y	0	0
BCD	A	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	BINBCD			
00002		PP 0	DUAL	Input WORD (binary number)
00003		PP 0	BCD	Output WORD (BCD coded number)

BINARY TO BCD CONVERSION, DOUBLE WORD — DUALBCDD

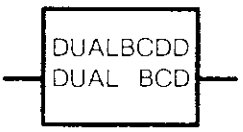
The binary number at the input DUAL is converted to a BCD coded number and is allocated to the operand at the output BCD.

The binary number is represented in 32 bits and must lie within the range

$0 \leq$ DUAL $\leq$ 5F5E0FF H
(corresponding to BCD 99 999 999).

The BCD number is limited to 99 999 999 if it lies outside of this range. The BCD number is stored in a 32-bit word.

FBD/LD



IL

```
!BA     0
DWBCD
        DUAL
        BCD
```

Parameters

DUAL	DOUBLE WORD	MD, KD	Binary number
BCD	DOUBLE WORD	MD	BCD coded number

CE Data

Runtime:	
Basic runtime:	36 – 288 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The binary number at the input DUAL is converted to a BCD coded number and is allocated to the operand at the output BCD.

The binary number is represented in 32 bits and must lie within the range

$0 \leq DUAL \leq 5F5E0FF\ H$
(corresponding to BCD 99 999 999).

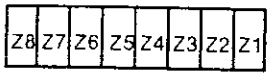
The BCD number is limited to 99 999 999 if it lies outside of this range. The BCD number is stored in a 32-bit word.

The input and output can neither be duplicated nor negated.

Definition

The significance of the digits in a hexadecimal number and a BCD coded number is defined as follows:

HEXDEC NUMBER

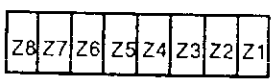


Numerical value:

Z1 * 1
Z2 * 16
Z3 * 256
Z4 * 4096
Z5 * 65536
Z6 * 1048576
Z7 * 16777216
Z8 * 268435456

$0 \leq Zi \leq F$

BCD NUMBER



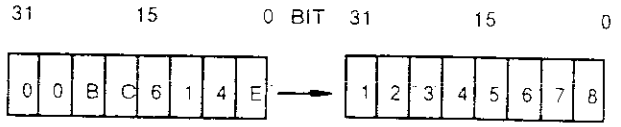
Numerical value:

Z1 * 1
Z2 * 10
Z3 * 100
Z4 * 1000
Z5 * 10000
Z6 * 100000
Z7 * 1000000
Z8 * 10000000

$0 \leq Zi \leq 9$

Example:

HEXDEC NUMBER BCD NUMBER

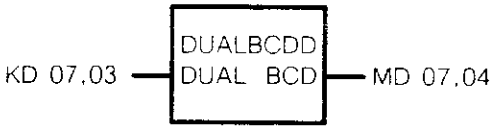


Z1 = 14 * 1	=	14	Z1 = 8 * 1	=	8
Z2 = 4 * 16	=	64	Z2 = 7 * 10	=	70
Z3 = 1 * 256	=	256	Z3 = 6 * 100	=	600
Z4 = 6 * 4096	=	24576	Z4 = 5 * 1000	=	5000
Z5 = 12 * 65536	=	786432	Z5 = 4 * 10000	=	40000
Z6 = 11 * 1048576	=	11534336	Z6 = 3 * 100000	=	300000
Z7 = 0 * 16777216	=	0	Z7 = 2 * 1000000	=	2000000
Z8 = 0 * 268435456	=	0	Z8 = 1 * 10000000	=	10000000
		+ 12345678			+ 12345678

Converting a negative binary number to a BCD number
See function block DUALBCD

Example

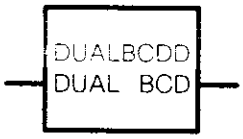
FBD/LD



IL

```
!BA     0
DWBCD
        KD 07,03
        MD 07,04
```

CE FBD Definition



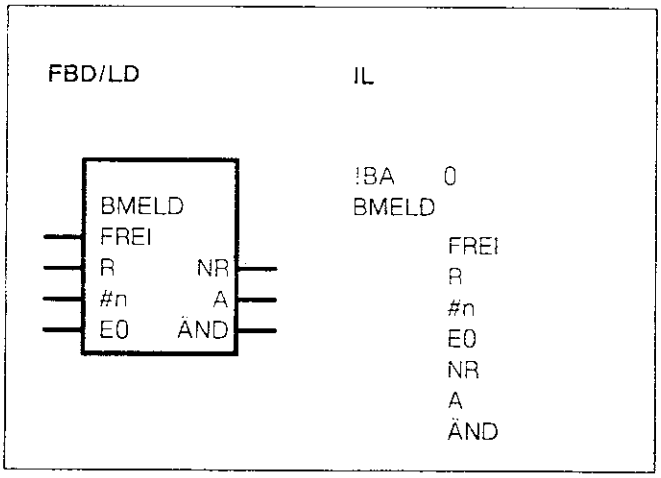
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
DUAL	E	D	N	P	Y	0	0
BCD	A	D	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	DWBCD			
00002		PP 0	DUAL	Input DOUBLE WORD (binary number)
00003		PP 0	BCD	Output DOUBLE WORD (BCD coded number)

BINARY VALUE CHANGE ANNUNCIATOR — BMELD

This function block monitors the binary values present at the input E0 capable of duplication for changes.



Parameters

FREI	BINARY	A, E, M, S, K	Block enable
R	BINARY	A, E, M, S, K	Reset
#n	DIRECT CONSTANT	#, #H	Number of input values
E0	BINARY	A, E, M, S, K	Input values; input can be duplicated
NR	WORD	AW, MW	Number of the input value
A	BINARY	A, M	Current input value
ÄND	BINARY	A, M	Change detected

CE Data

Runtime:	
Basic runtime:	61 µs
Additional runtime:	31 µs per input E0 ... E_{n-1}
Output updating:	yes if FREI=1
Number of historical values:	even number of binary values: (2 + #n/2) words odd number of binary values: (2 + (#n + 1)/2) words
Available as of:	ABB Procontic T320 V4 / 935 PC 82 R101 / 35 ZE 93 R101

Description

This function block monitors the binary values present at the input E0 capable of duplication for changes.

The inputs and outputs cannot be negated/inverted. The input E0 is capable of duplication.

Detection of a change

Each time the block is processed, the current input values at the inputs E0...En-1 are compared to the historical values (input values from the previous time the block was processed). If a change is detected at one of the inputs E0...En-1:

- this is signalled at the output ÄND
- the number of the input at which the change was discovered is output through the output NR
- the input value that has changed is output through the output A.

Each time the block is processed, a change at only *one* input is recognized. When a change is recognized, the inputs following the input where the change was previously discovered are monitored the next time the block is processed.

Initialization of historical values

The first time the block is processed after PLC initialization (Frei = 1) or processing is enabled after it had been disabled (FREI changes from 0 to 1), all current input values are taken over once as historical values and all outputs are set to the value 0. These initialized historical values now represent the starting basis for recognition of changes.

FREI BINARY
Processing of the block is enabled with the FREI input.
FREI = 0 -> Block is not processed
FREI = 1 -> Processing of the block is enabled

If FREI = 0, the block's outputs are also no longer updated.

R BINARY
The block can be reset with the R input.
R = 0 -> No reset
R = 1 -> Reset of the block
Reset signifies:
- Adoption of the current values at the inputs E0...En-1 as historical values.
- All outputs are set to the value 0.

#n DIRECT CONSTANT
The number of values to be monitored at the inputs E0...En-1 are specified at the #n input. This is specified as a direct constant.
Range for #n: $\leq$ #n $\leq$ 127

E0...En-1 BINARY
The input E0 is capable of duplication (E0...En-1).
The operands to be monitored for a change are specified at the inputs E0 ... En-1

NR WORD
The serial number of the input E0...En-1 where a change has been discovered is output through the output NR.

If *no* change is discovered during processing of the block, the number of the input that changed last continues to be output through the output NR.
The following affiliations apply:
Change discovered at E0 -> NR = 0
Change discovered at E1 -> NR = 1
Change discovered at En-1 -> NR = n-1

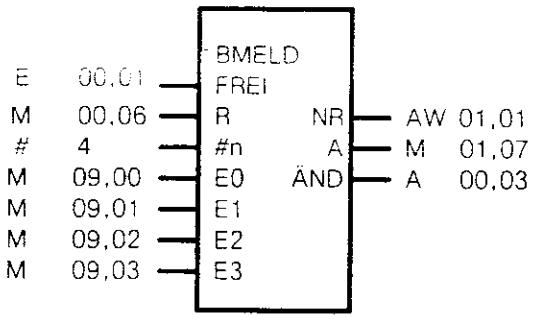
A BINARY
If a change is discovered at one of the inputs E0...En-1, the input value that has changed is allocated to the output A.

If *no* change is discovered at the inputs E0...En-1 during processing of the block, the value of the input that has changed last continues to be output through the output A.

ÄND BINARY
The output ÄND indicates whether or not a change has been discovered at the inputs E0...En-1.
ÄND = 0 -> No change discovered
ÄND = 1 -> Change discovered

Example

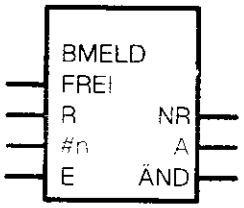
FBD/LD



IL

```
!BA     0
BMELD
        E     00,01
        M     00,06
        #     4
        M     09,00
        M     09,01
        M     09,02
        M     09,03
        AW    01,01
        M     01,07
        A     00,03
```

CE FBD Definition



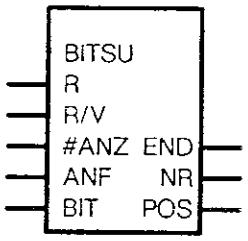
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
R	E	L	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
E	E	L	N	P	Y	1	0
NR	A	W	N	P	Y	0	0
A	A	L	N	P	Y	0	0
ÄND	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	BMELD			
00002		PP 0	FREI	Input BINARY (block enable)
00003		PP 0	R	Input BINARY (reset)
00004		PP 0	#n	# CONSTANT (number of input values)
	[	1		
00005		PP 1	E	Input BINARY (input values)
	]	1		
00006		PP 0	NR	Output WORD (No.)
00007		PP 0	A	Output BINARY (current input value)
00008		PP 0	ÄND	Output BINARY (change detected)

This function block searches through a bit field for a set bit. The bit field consists of successive word operands. If a set bit is found, this is indicated at the block's outputs.

FBD/LD



IL

```
!BA   0
BITSU
      R
      R/V
      #ANZ
      ANF
      BIT
      END
      NR
      POS
```

Parameters

R	BINARY	A, E, M, S, K	Reset of the block and new search
R/V	BINARY	A, E, M, S, K	Search direction, up/down
#ANZ	DIRECT CONSTANT	#, #H	Field length, number of word operands
ANF	WORD	EW, AW, MW, KW	Field start, 1st word operand
BIT	WORD	EW, AW, MW, KW	Number of set bits to be skipped
END	BINARY	A, M	Field end reached
NR	WORD	AW, MW	Number of the word operand containing the set bit
POS	WORD	AW, MW	Bit position within the word operand

CE Data

Runtime:	
Basic runtime:	136 µs
Additional runtime:	26 µs per word variable
Ouput updating:	yes
Number of historical values:	9 words
Available as of:	ABB Procontic T320 V4 / 935 PC 82 R101 / 35 ZE 93 R101

Description

This function block searches through a bit field for a set bit. The bit field consists of successive word operands. If a set bit is found, this is indicated at the block's outputs.

The inputs and outputs can neither be duplicated nor negated/inverted.

The bit field consists of successive word operands. The first word operand in the bit field is specified at the input ANF.

After the start of the PLC program (1st program cycle), the block sets all outputs to the value 0 and immediately begins searching for set bits. Therefore, the search is already active in the first program cycle.

If the block finds a set bit, its position is indicated at the block's outputs and the search is ended. The next time the block is called in the next program cycle, it continues the search, doing so at the bit position directly following the bit found last. If the end of the bit field is reached during the search, this is indicated and a new search can be started again by means of a reset signal. The new search again begins at the start of the bit field.

R BINARY
The input R serves to reset the block and start a new search from the beginning of the bit field.
R = 0 -> No reset
R = 1 -> Reset is triggered
A 1 signal at the input R resets all of the block's outputs to the value 0 *and* results in immediate starting of a new search from the beginning of the bit field. Therefore, the new search begins in the same processing cycle in which the block's outputs are reset.

The search within the reset cycle ends either at the first set bit or, if no bit is set, at the end of the field. The next time the block is called, the reset input must be set to 0 so that the block will be able to continue searching through the bit field for set bits. The reset input is dominant with respect to the other inputs.

R/V BINARY
The searching direction is specified at the input R/V.
R/V = 0 -> Search down
R/V = 1 -> Search up

R/V = 0 means: The *start* of the bit field within the meaning of the search is identical with the *physical start* of the bit field.

R/V = 1 means: The *start* of the bit field within the meaning of the search is identical with the *physical end* of the bit field.

#ANZ DIRECT CONSTANT
The number of word operands of which the bit field consists is specified at the input #ANZ. This is specified as a direct constant.

ANF WORD
The word operand with which the bit field physically begins is specified at the input ANF. The entire bit field consists of the operand at the input ANF and the subsequent operands corresponding to the operand numbering. The total number of word operands is specified at the input #ANZ.

BIT WORD
The way in which the function block is to indicate the bits set in the bit field is specified at the input BIT.

BIT = 1 -> *Each* set bit in the bit field is indicated at the outputs NR and POS.
BIT = 2 -> The *first* set bit in the bit field is indicated; of the other set bits in the bit field, only every *second* set bit is now indicated.
BIT = 3 -> The *first* set bit in the bit field is indicated; of the other set bits in the bit field, now only every *third* set bit is indicated.
BIT = n -> The *first* set bit in the bit field is indicated; of the other set bits in the bit field, now only every n-th set bit is indicated.

- Special function: BIT = 0
  If the block finds a set bit, in the next program cycle the search is *not automatically* continued at the subsequent bit position. The search is interrupted at this point until the bit found has assumed the significance 0. During interruption of the search, the position of the bit found last continues to be indicated at the outputs. If the bit found last assumes the value 0, the search is continued at the next bit position. If a further bit is set in the selected search direction, this bit's position is indicated and the search is interrupted again.
  If no more bits are set in the search direction, the position of the bit found last is indicated. The output END (field end reached) is set to 1.

- If no bits are set after a reset, the block runs through the bit field up to its end and stops there. The output END (field end reached) is set to one. The outputs NR and POS are set to 0.

- If bits are set after a reset, the block indicates the first set bit in the search direction and stops at this bit.

END BINARY
Whether or not the end of the bit field has been reached during the search is indicated at the END output.
END = 0 -> End of the bit field not reached
END = 1 -> End of the bit field reached

Search down: The end of the bit field is defined by bit position 15 in the *last* word operand of the physical bit field.

Search up: The end of the bit field is defined by bit position 15 in the *first* word operand of the physical bit field.

If the bit is set at the last bit position of the field and it is found and indicated during a search, the output END is not yet set to the value 1. This is not done until the next processing cycle of the block. Therefore, the prerequisite for setting the output END is that *no* set bit has been found during the current search.

NR, POS WORD
If the block finds a set bit in the planned bit field, its position is indicated at the outputs NR and POS.
Meanings:
NR: Current number of the word operand in which the set bit has been found.
POS: Position of the set bit within the word operand.

Current number of the word operand
NR = 0 -> 1st word operand of the bit field
NR = 1 -> 2nd word operand of the bit field
. . .
. . .
NR = n-1 -> nth word operand of the bit field

The word operands are numbered in the sense of the search.
Therefore, the following applies to down searching:
First word operand ≙ first word operand of the physical bit field
Last word operand ≙ last word operand of the physical bit field

Therefore, the following applies to up searching:
First word operand ≙ last word operand of the physical bit field
Last word operand ≙ first word operand of the physical bit field

Position of the set bit within the word operand
Numbering within a word operand is from 0...15. At the same time, position 0 corresponds to the least significant bit and position 15 to the most significant bit within the word operand.
If the end of the field is reached during a search without a set bit having been found, the position of the bit found last continues to be output through the output NR and the output POS. This takes place until a new search is compelled from the start of the bit field by a reset at the input R.

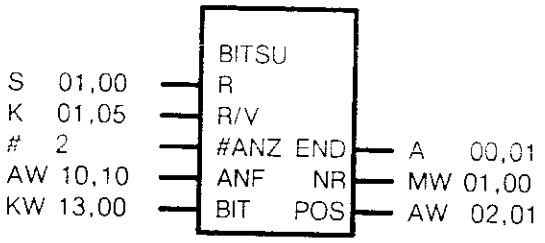
If no bits are set at all within the entire bit field, the outputs assume the following values at the end of the first search:

END = 1
NR = 0
POS = 0

This state can be terminated again by a 1 signal at the input R (reset).

Example

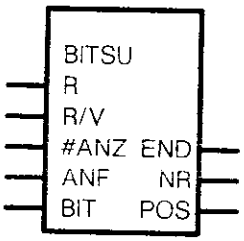
FBD/LD



IL

```
!BA    0
BITSU
       S   01,00
       K   01.05
       #   2
       AW  10,10
       KW  13,00
       A   00,01
       MW  01,00
       AW  02,01
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
R	E	L	N	P	Y	0	0
R/V	E	L	N	P	Y	0	0
#ANZ	K	W	N	P	Y	0	0
ANF	E	W	N	P	Y	0	0
BIT	E	W	N	P	Y	0	0
END	A	L	N	P	Y	0	0
NR	A	W	N	P	Y	0	0
POS	A	W	N	P	Y	0	0

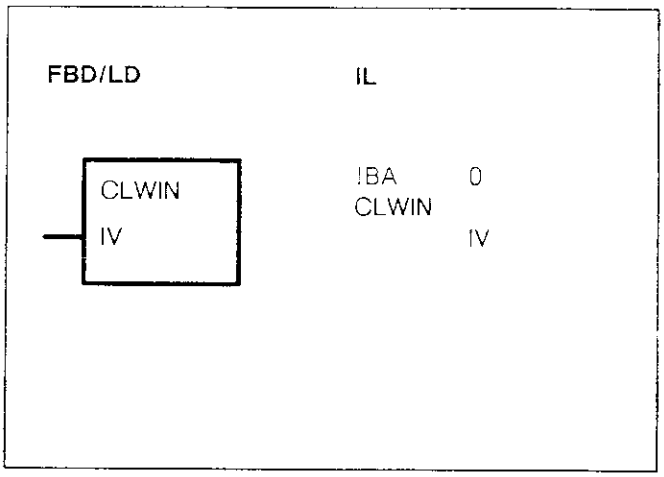
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	BITSU			
00002		PP 0	R	Input BINARY (reset)
00003		PP 0	R/V	Input BINARY (search direction)
00004		PP 0	#ANZ	DIRECT CONSTANT (#words field length)
00005		PP 0	ANF	Input WORD (field start)
00006		PP 0	BIT	Input WORD (number of set bits to be skipped)
00007		PP 0	END	Output BINARY (field end reached)
00008		PP 0	NR	Output WORD (No.)
00009		PP 0	POS	Output WORD (bit position)

CLEAR ALL MEASUREMENT WINDOWS (OMS-F) CLWIN

This function block clears all measurement windows on the iconic processor 35 IV 90.

This block is needed when using the video sensor OMS-F.



Parameters

IV	WORD	EW, AW, MW, KW	Number of the 35 IV 90 module

CE Data

Runtime:	
Basic runtime:	27.2 µs
Additional runtime:	no
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block clears all measurement windows on the iconic processor 35 IV 90.

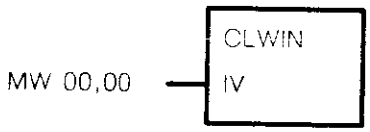
IV WORD

Number of the 35 IV 90 module.

The number of the required 35 IV 90 module with which measurement is to take place is specified with the operand at this input.
Value range: 0...5

Example

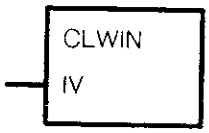
FBD/LD



IL

```
!BA     0
CLWIN
        MW 00,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
IV	E	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	CLWIN			
00002		PP 0	IV	Input WORD (IV No.)

This function allows users to set and output the current time and the current date.

FBD/LD

IL

```
IBA    0
UHR
       FREI
       S
       SEC
       MIN
       H
       TAG
       MON
       JHR
       WTG
       AKT
       FEHL
       ASEC
       AMIN
       AH
       ATAG
       AMON
       AJHR
       AWTG
```

Parameters

FREI	BINARY	E, A, M, K, S	Enable block processing
S	BINARY	E, A, M, K, S	0/1 edge sets the time and date
SEC	WORD	EW, AW, MW, KW	Set input for the seconds
MIN	WORD	EW, AW, MW, KW	Set inputs for the minutes
H	WORD	EW, AW, MW, KW	Set inputs for the hours
TAG	WORD	EW, AW, MW, KW	Set inputs for the days
MON	WORD	EW, AW, MW, KW	Set inputs for the months
JHR	WORD	EW, AW, MW, KW	Set inputs for the years
WTG	WORD	EW, AW, MW, KW	Set inputs for the weekdays
AKT	BINARY	A, M	Topicality (usefulness) of the data at the outputs
FEHL	WORD	AW, MW	Error identifier
ASEC	WORD	AW, MW	Seconds output
AMIN	WORD	AW, MW	Minutes output
AH	WORD	AW, MW	Hours output
ATAG	WORD	AW, MW	Days output
AMON	WORD	AW, MW	Months output
AJHR	WORD	AW, MW	Years output
AWTG	WORD	AW, MW	Weekday No. output

CE Data

Runtime:	
Basic runtime:	240 µs – Display of time and date 300 µs – Setting and displaying time and date
Additional runtime:	---
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function allows users to set and output the current time and the current date.

The inputs and outputs can neither be duplicated nor inverted nor negated.

Note:

The function block UHR operates together with the hardware component MM 58274A, which must be installed on the interface board. A jumper on connector X17 on the interface board must also be set to pins 5-6.

The clock is set by means of the set inputs for the time and date. The values present at the set inputs are adopted by a 0/1 edge at the input S. As long as a 1 signal is present at the FREI input, the current date and time are indicated at the block's outputs.

The time and date can also be displayed and set by means of the PLC commands UHR <CR> and UHRS <CR>.

FREI BINARY

Block enable

FREI = 0: The block is not processed. The AKT and FEHL outputs are set to 0. The time and date outputs are no longer changed by the block.

FREI = 1: Block is processed

S BINARY

0/1 edge –> the clock is set to the values present at the time and date inputs.

After setting, the time and date are displayed at the corresponding outputs and still in the same cycle.

Set inputs for date and time

In the event of a 0/1 edge at the input S, the clock is set to the values present at the set inputs. If the specified set values are inadmissible, the AKT output is set to 0 and an error message appears at the FEHL output. The values present at the time and date outputs are invalid in this case. The clock has to be set again.

SEC WORD

Set input for the seconds.
Value range: 0...59.

MIN WORD

Set input for the minutes.
Value range: 0...59.

H WORD

Set input for the hours.

The clock operates in 24 hour mode, i.e. it changes from 23:59:59 h to 0:0:0 h.
Value range: 0...23.

TAG WORD

Set input for the days (which day of the month)

The clock knows the number of days depending on the months and leap years. For the clock, a leap year exists when the year number is an integral multiple of 4. The maximum value for the days (28, 29, 30, 31) depends on the month.
Value range: 1...28, 29, 30, 31.

MON WORD

Set input for the month.
Value range: 1...12.

JHR WORD

Set input for the years.

The clock only indicates the years and decades.
Value range: 0...99.

WTG WORD

Set input for the number of the weekday.

This input specifies which day of the week the day is on which input is made. That is to say, it is possible to determine which day of the week the day with the number 1 is to be (e.g. Sunday or Monday).
Value range: 1...7.

Example:
The clock is set on Friday, 01.07.88. If the value 6 is entered for the week day number, Friday is now the 6th day of the week and Sunday is defined as the 1st day of the week.

AKT BINARY

Indication of the topicality (usefulness) of the outputs.

AKT is 1 if:

- The date and time outputs were updated in the current cycle;
- The values at the outputs are consistent, i.e. none of the values at the date or time outputs has changed during updating. They all originate from the same clock pulse;
- The clock was set correctly;
- The clock chip MM 58274A exists on the interface board.

AKT = 1 -> FEHL = 0: The above conditions are fulfulled

AKT = 0 -> FEHL > 0: An error has occurred; FEHL contains the affiliated error identifier.

FEHL WORD

In the event of an error, the relevant error identifier is available at the output FEHL.

Meanings of the error identifiers:

- No error has occurred:
  FEHL = 0: No error has occurred or FREI = 0, i.e. block disabled
- Error when setting the clock:
  FEHL = 1: $0 \leq SEC \leq 59$ has not been obeyed
  FEHL = 2: $0 \leq MIN \leq 59$ has not been obeyed
  FEHL = 3: $0 \leq H \leq 23$ has not been obeyed
  FEHL = 4: $1 \leq TAG \leq 28, 29, 30, 31$ (depending on the month) has not been obeyed
  FEHL = 5: $1 \leq MON \leq 12$ has not been obeyed
  FEHL = 6: $0 \leq JHR \leq 99$ has not been obeyed
  FEHL = 7: $1 \leq WTG \leq 7$ has not been obeyed
- Error addressing the hardware component MM 58274A:
  FEHL = 8: The hardware component MM 58274A does not exist or cannot be addressed.
- Error when updating the date and time:
  FEHL = 9: The time data is not coherent, i.e. it does not originate from the same clock pulse and may therefore be wrong. The data is always wrong when a new clock pulse occurs during updating and therefore one of the outputs switches from the maximum to the minimum, e. g. SEC from 59 -> 0.

Outputs for date and time

The outputs are updated whenever a 1 signal is present at the FREI input *and* the clock has been set once.

If the AKT output is equal to 1, the outputs for the date and time are valid. In the event of an error, an error identifier is output through the output FEHL.

ASEC WORD

Seconds output.
Value range: 0...59

AMIN WORD

Minutes output.
Value range: 0...59.

AH WORD

Hours output.
Value range: 0...23.

ATAG WORD

Days output.
Value range: 0...28, 29, 30, 31.

AMON WORD

Months output.
Value range: 1...12.

AJHR WORD

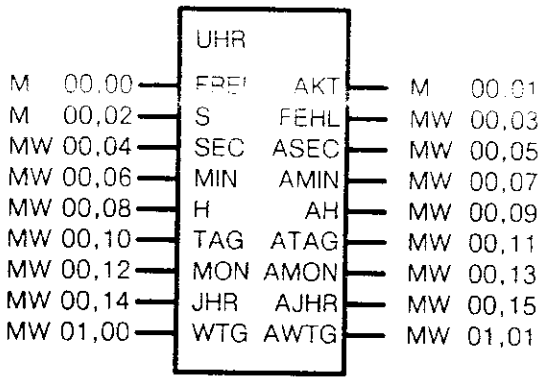
Years output.
Value range: 0...99.

AWTG WORD

Weekday output.
Value range: 1...7.

Example

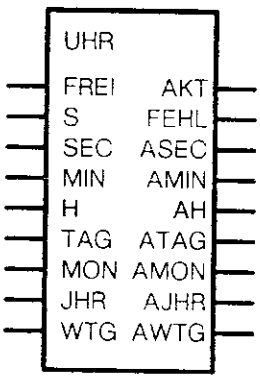
FBD/LD



IL

```
IBA   0
UHR
      M    00.00
      M    00,02
      MW   00,04
      MW   00,06
      MW   00,08
      MW   00,10
      MW   00,12
      MW   00,14
      MW   01,00
      M    00,01
      MW   00,03
      MW   00,05
      MW   00,07
      MW   00,09
      MW   00,11
      MW   00,13
      MW   00,15
      MW   01,01
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
AKT	A	L	N	P	Y	0	0
S	E	L	N	P	Y	0	0
FEHL	A	W	N	P	Y	0	0
SEC	E	W	N	P	Y	0	0
ASEC	A	W	N	P	Y	0	0
MIN	E	W	N	P	Y	0	0
AMIN	A	W	N	P	Y	0	0
H	E	W	N	P	Y	0	0
AH	A	W	N	P	Y	0	0
TAG	E	W	N	P	Y	0	0
ATAG	A	W	N	P	Y	0	0
MON	E	W	N	P	Y	0	0
AMON	A	W	N	P	Y	0	0
JHR	E	W	N	P	J	0	0
AJHR	A	W	N	P	Y	0	0
WTG	E	W	N	P	Y	0	0
AWTG	A	W	N	P	Y	0	0

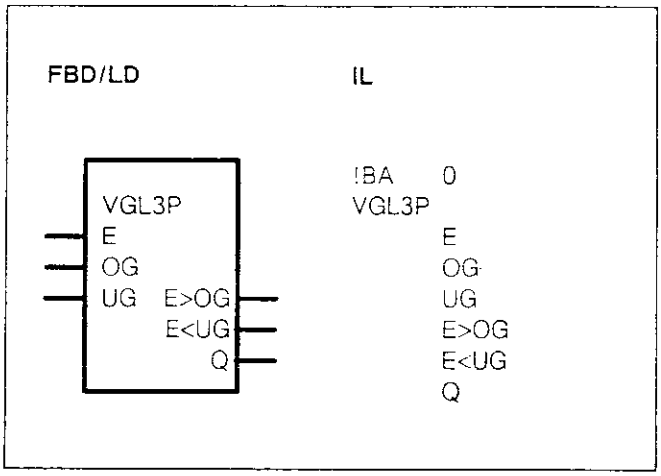
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	UHR			
00002		PP 0	FREI	
00003		PP 0	S	
00004		PP 0	SEC	
00005		PP 0	MIN	
00006		PP 0	H	
00007		PP 0	TAG	
00008		PP 0	MON	
00009		PP 0	JHR	
00010		PP 0	WTG	
00011		PP 0	AKT	
00012		PP 0	FEHL	
00013		PP 0	ASEC	
00014		PP 0	AMIN	
00015		PP 0	AH	
00016		PP 0	ATAG	
00017		PP 0	AMON	
00018		PP 0	AJHR	
00019		PP 0	AWTG	

The value of the operand at the input E is compared to the values of the operands at the inputs OG and UG.

The possible results are signalled at the outputs E>OG, E<UG and Q.
The following applies:

E < UG	---> E>OG = 0, E<UG = 1, Q = 0
UG ≤ E ≤ OG	---> E>OG = 0, E<UG = 0, Q = 1
E > OG	---> E>OG = 1, E<UG = 0, Q = 0



Parameters

E	WORD	EW, MW, AW, KW	Input value
OG	WORD	EW, MW, AW, KW	High limit
UG	WORD	EW, MW, AW, KW	Low limit
E>OG	BINARY	A, M	Value > high limit
E<UG	BINARY	A,M	Value < low limit
Q	BINARY	A,M	Low limit ≤ input value ≤ high limit

CE Data

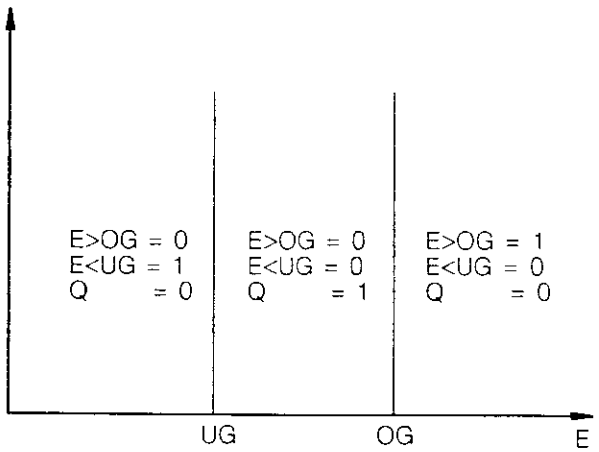
Runtime:	
Basic runtime:	48 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, 801 / 35 ZE 93 R101

Description

The value of the operand at the input E is compared to the values of the operands at the inputs OG and UG.

The possible results are signalled at the outputs E>OG, E<UG and Q.
The following applies:

E < UG	---> E>OG = 0, E<UG = 1, Q = 0
UG ≤ E ≤ OG	---> E>OG = 0, E<UG = 0, Q = 1
E > OG	---> E>OG = 1, E<UG = 0, Q = 0



The inputs can neither be duplicated nor negated. The outputs can neither be duplicated nor inverted.

Number range

Integer word (16 bits)

The following especially applies here to the non-negated inputs:

- Low level 8000 H -32768
- High level 7FFF H +32767

The following generally applies:

- Low limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside* of the number range and is neither generated nor processed correctly by the PLC. If this forbidden value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

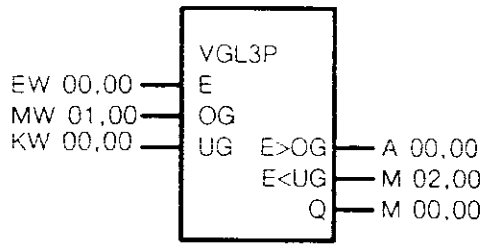
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

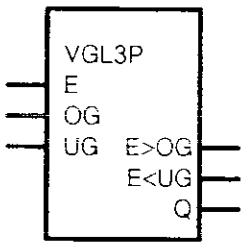
FBD/LD



IL

```
!BA    0
VGL3P
       EW   00.00
       MW   01.00
       KW   00.00
       A    00.00
       M    02.00
       M    00.00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	W	N	P	Y	0	0
OG	E	W	N	P	Y	0	0
UG	E	W	N	P	Y	0	0
E>OG	A	L	N	P	Y	0	0
E<UG	A	L	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	VGL3P			
00002		PP 0	E	Input WORD (input value)
00003		PP 0	OG	Input WORD (high limit)
00004		PP 0	UG	Input WORD (low limit)
00005		PP 0	E>OG	Output BINARY (value > high limit)
00006		PP 0	E<UG	Output BINARY (value < low limit)
00007		PP 0	Q	Output BINARY (UG ≤ E ≤ OG)

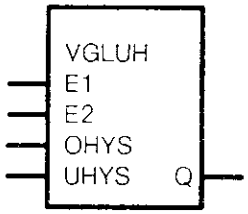
COMPARATOR WITH ASYMMETRICAL HYSTERESIS — VGLUH

The values of the operands at the inputs E1 and E2 are compared to each other. Taking the hystereses at the inputs OHYS (high hysteresis) and UHYS (low hysteresis) into account, the result is signalled at the output Q.

The following applies:

E1 < E2 − UHYS	---> Q = 0
E1 ≥ E2 + OHYS	---> Q = 1
E2−UHYS ≤ E1 ≤ E2+OHYS	---> Q as in the previous cycle

FBD/LD



IL

```
!BA   0
VGLUH
      E1
      E2
      OHYS
      UHYS
      Q
```

Parameters

E1	WORD	EW, MW, AW, KW	Input value 1
E2	WORD	EW, MW, AW, KW	Input value 2
OHYS	WORD	EW, MW, AW, KW	High hysteresis
UHYS	WORD	EW, MW, AW, KW	Low hysteresis
Q	BINARY	A, M	Output

CE Data

Runtime:	
Basic runtime:	58 ... 63 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, 801 / 35 ZE 93 R101

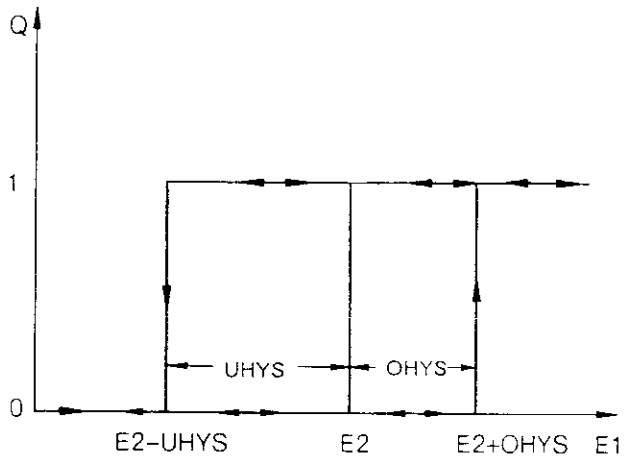
Description

The values of the operands at the inputs E1 and E2 are compared to each other. Taking the hystereses at the inputs OHYS (high hysteresis) and UHYS (low hysteresis) into account, the result is signalled at the output Q.

The following applies:

E1 < E2 − UHYS	---> Q = 0
E1 ≥ E2 + OHYS	---> Q = 1
E2−UHYS ≤ E1 ≤ E2+OHYS	---> Q as in the previous cycle

The inputs can neither be duplicated nor negated.
The output can neither be duplicated nor inverted.



Number range

Integer word (16 bits)

The following especially applies here:
E1 ≥ 8000 H (-32768)
E2 - UHYS ≥ 8001 H (-32767)

the following generally applies:
- Low limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

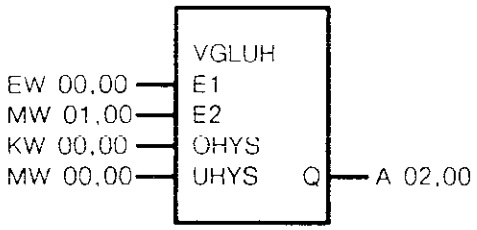
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

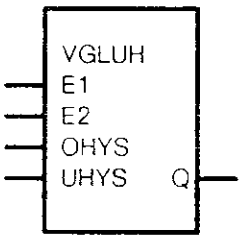
FBD/LD



IL

```
!BA    0
VGLUH
       EW 00,00
       MW 01,00
       KW 00,00
       MW 00,00
       A  02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	Y	0	0
E2	E	W	N	P	Y	0	0
OHYS	E	W	N	P	Y	0	0
UHYS	E	W	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

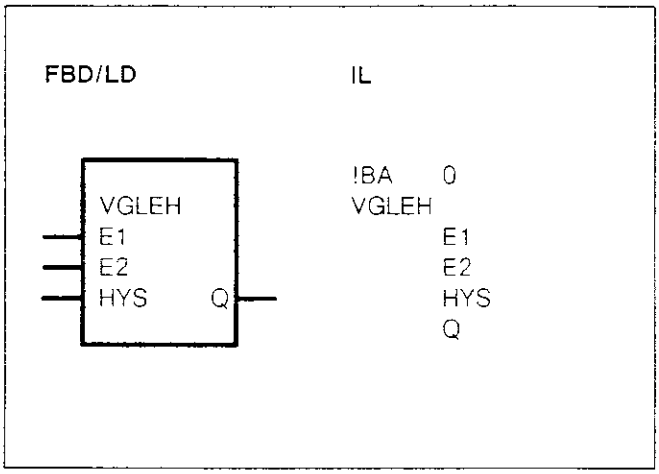
00000	!BA	0	Nr	Block No. (preset to 0)
00001	VGLUH			
00002		PP 0	E1	Input WORD
00003		PP 0	E2	Input WORD
00004		PP 0	OHYS	Input WORD (high hysteresis)
00005		PP 0	UHYS	Input WORD (low hysteresis)
00006		PP 0	Q	Output BINARY

COMPARATOR WITH UNILATERAL HYSTERESIS VGLEH

The values of the operands at the inputs E1 and E2 are compared to each other. Taking the hysteresis at the input HSY into account, the result is signalled at the output Q.

The following applies:

E1 ≥ E2 ---> Q = 1
E1 < E2 - HYS ---> Q = 0
E2 - HYS ≤ E1 < E2 ---> Q as in the previous cycle



Parameters

E1	WORD	EW, MW, AW, KW	Input value 1
E2	WORD	EW, MW, AW, KW	Input value 2
HYS	WORD	EW, MW, AW, KW	Hysteresis
Q	BINARY	A, M	Result of the comparison

CE Data

Runtime:	
Basic runtime:	56 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, 801 / 35 ZE 93 R101

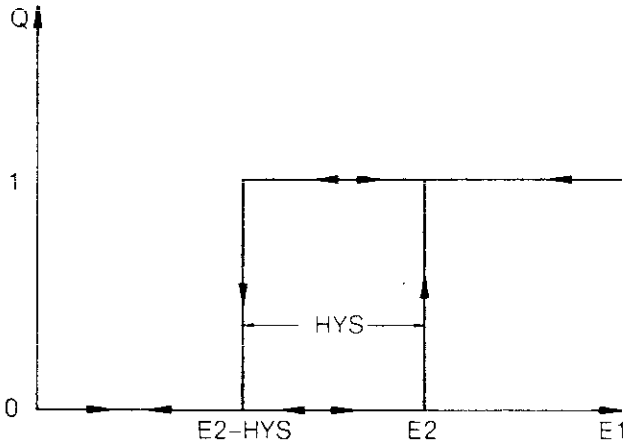
Description

The values of the operands at the inputs E1 and E2 are compared to each other. Taking the hysteresis at the input HSY into account, the result is signalled at the output Q.

The following applies:

E1 ≥ E2 ---> Q = 1
E1 < E2 - HYS ---> Q = 0
E2 - HYS ≤ E1 < E2 ---> Q as in the previous cycle

The inputs can neither be duplicated nor negated.
The output can neither be duplicated nor inverted.



HYS = Hysteresis

Number range

Integer word (16 bits)

• Low limit:	8001 H	-32767
• High limit:	7FFF H	+32767
• Inadmissible value:	8000 H	---

The following especially applies here to the specification for the left edge of the hysteresis:
E2 – HYS ≥ –32 767 (8001 H)

In the two's complement arithmetic, the value 8000H (–32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

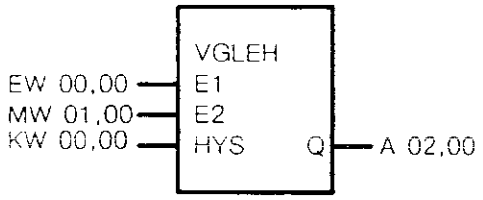
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (–32767).

Example

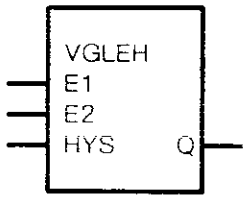
FBD/LD



IL

```
!BA    0
VGLEH
       EW 00,00
       MW 01,00
       KW 00,00
       A  02,00
```

CE FBD Definition

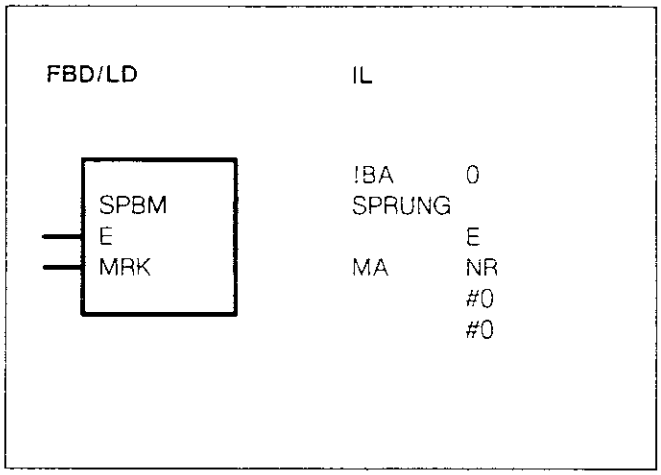


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	Y	0	0
E2	E	W	N	P	Y	0	0
HYS	E	W	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	VGLEH			
00002		PP 0	E1	Input WORD
00003		PP 0	E2	Input WORD
00004		PP 0	HYS	Input WORD (hysteresis)
00005		PP 0	Q	Output BINARY

This function block produces a jump to the target label specified at the MRK input if the jump condition at the input E is fulfilled.



Parameters

E	BINARY	E, M, A, K, S	Jump condition
MRK	SPECIAL	MRK	Target label

CE Data

Runtime:	
Basic runtime:	26 µs
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block produces a jump to the target label specified at the MRK input if the jump condition at the input E is fulfilled.

To avoid endless loops as the result of incorrect planning, the PLC accepts FORWARD JUMPS only.

The target label's designation consists of a key word and a number.

FBD/LD

Designation of the target label: MRK No.

Where:	MRK	Key word
	0 ≤ No. ≤ 999	Number of the label

IL

Designation of the target label: MA No.

Where:	MA	Key word
	0 ≤ No. ≤ 999	Number of the label

The label specified in the block is also specified at the position in the user program to where the jump is to take place. It is possible for several jump blocks to jump to one and the same label.

A label may occur several times in the program. If this is case, the jump block and the next one of these identically named labels always belong together.

On the basis of the labels, the PLC calculates the following:

- The number of historical values to be skipped
- The address of the target label

The PLC enters the two computed values at the intended points in the jump block. When planning the jump block in instruction list, these two program words may basically be written with any contents. However, it is advisable to enter the number 0 here.

Caution:

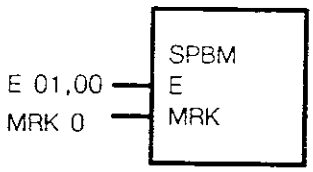
However, the two program words should not be written with NOP because NOP values are removed when optimizing the user program.

Example of a jump block in IL:

```
 IE0,0
 &E0,1
 /M0,0
 =M0,1
 IBA 123
 SPRUNG    Jump block
 M0,1      Jump condition
 MA 99     Target label's name
 #0        Free space for number of historical
           values to be skipped
 #0        Free space for target address
 IE1,5
 &E4,9
 =A6,8
 =M9,7
 .
 .
 .
 MA 99     Target label, jump target for the jump
           block
 IE10,12
 /E10,13
 &E10,14
 =A14,15
 IPE
```

Example

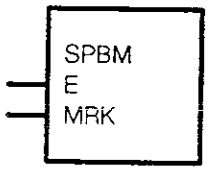
FBD/LD



IL

```
!BA      0
SPRUNG
         E 01,00
MA       0
         # 0
         # 0
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	L	N	P	Y	0	0
MRK	S	S	N	P	Y	0	0

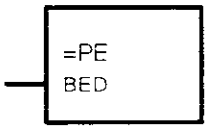
CE IL Definition

```
00000   !BA      0        Nr      Block No. (preset to 0)
00001   SPRUNG
00002            PP  0    E       Jump condition BINARY
00003   MA       PP  0    MRK     Target lable
00004            #   0
00005            #   0
```

CONDITIONAL PROGRAM END =PE

The conditional program end may be planned within a PLC program. Depending on the status of the operand at the input BED, processing of the PLC program is ended or is not ended here.

FBD/LD



IL

```
!   BED
=   PE
```

Parameters

BED	BINARY	E, M, A, S, K	Condition for program end

CE Data

Runtime:	
Basic runtime:	2 µs
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V8 / 35 ZE 94 R101

Description

The conditional program end may be planned within a PLC program. Depending on the status of the operand at the input BED, processing of the PLC program is ended or is not ended here.

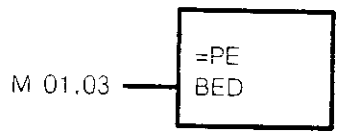
The following applies to the condition at the BED input:

BED = 0: Processing of the PLC program is continued
BED = 1: The PLC program is processed only up to this point. The subsequent part of the PLC program is *not* processed.

The input can neither be duplicated nor inverted.

Example

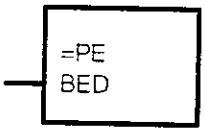
FBD/LD



IL

```
!     M 01.03
=     PE
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
BED	E	L	N	P	Y	0	0

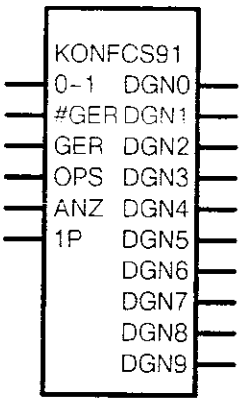
CE IL Definition

00000	!	PP 0	BED	Condition for program end (BINARY)
00002	=PE			

The connection element KONFCS91 serves to configure a 35 CS 91 module (coupler ABB Procontic T300 with ABB Procontic CS31).

The CE outputs the diagnosis data of 35 CS 91.

FBD/LD



IL

See CE IL in the programming system

Parameters

0-1	BINARY	E, M	Input for 0→1 edge
#GER	DIRECT CONSTANT	#, #H	Module address of 35 CS 91 module
GER	WORD	EW, MW, KW	Module address of 35 CS 91 module
OPS	WORD	EW, MW, KW	Operand segment
ANZ	WORD	EW, MW, KW	Number of the CS31 modules on the 35 CS 91 module
1P	WORD	MW, KW	First parameter of the configuration data list
DGN0	WORD	MW, AW	Diagnosis word 0 of the 35 CS 91
DGN1	WORD	MW, AW	Diagnosis word 1 of the 35 CS 91
DGN2	WORD	MW, AW	Diagnosis word 2 of the 35 CS 91
DGN3	WORD	MW, AW	Diagnosis word 3 of the 35 CS 91
DGN4	WORD	MW, AW	Diagnosis word 4 of the 35 CS 91
DGN5	WORD	MW, AW	Diagnosis word 5 of the 35 CS 91
DGN6	WORD	MW, AW	Diagnosis word 6 of the 35 CS 91
DGN7	WORD	MW, AW	Diagnosis word 7 of the 35 CS 91
DGN8	WORD	MW, AW	Diagnosis word 8 of the 35 CS 91
DGN9	WORD	MW, AW	Diagnosis word 9 of the 35 CS 91

CE Data

Runtime:	
Basic runtime:	not available
Additional runtime:	not available
Output updating:	yes
Number of historical values:	none
Available as of:	35 ZE 94 R101

Description

The function block serves to configure a 35 CS 91 module (coupler ABB Procontic T300 with ABB Procontic CS31).

Configuration of the 35 CS 91 defines data exchange between the 35 CS 91 module and the CS31 modules (which?/how many?). To do that, a configuration data list is transferred to the 35 CS 91 module after a 0→1 edge at the input 0-1.

The CE supplies the 35 CS 91 diagnosis data at its outputs cyclically (independent of edge input 0-1).

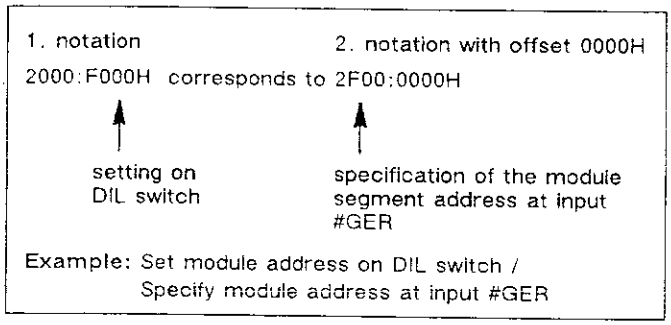
The diagnosis word 9 of the 35 CS 91 module (configuration error,...) is aknowledged (set to 0) before transfer of a configuration to the 35 CS 91 module.

0-1 BINARY
The configuration is transferred to the 35 CS 91 module with a 0→1 edge at input 0-1.

#GER DIRECT CONSTANT
The module address of the 35 CS 91 module is specified at the input #GER.

Structure of the module address:
module segment address : module offset address

Attention: Consider the module address 2000:F000H. The module offset address F000H of the first notation is set on the DIL switch. The module address 2000:F000H corresponds to the address 2F00:0000H of the second notation. At input #GER you have to specify the module segment address 2F00 of the second notation.



GER WORD
The module address of the 35 CS 91 module is specified at the input GER (same value as at input #GER). For further explanations: see input #GER.

OPS WORD
Operand segment, depending on the PLC program in which the CE has been inserted:
Input for N1: $8800_H = -30720_D$
Input for N2: $8F40_H = -28864_D$

ANZ WORD
Number of the CS31 modules which are connected to the 35 CS 91 module. This parameter also defines the length of the configuration data list.

The number of the CS31 modules (max. 31) in the configuration data list must agree with the CE input ANZ.

1P WORD
First parameter of the configuration data list. The configuration data list consists of individual parameters which have to be specified in uninterrupted ascending order in the word constant or word flag area. It has to have the following structure:

1st CS31 module, CS31 module type
1st CS31 module, CS31 module address
2nd CS31 module, CS31 module type
2nd CS31 module, CS31 module address
:
last CS31 module, CS31 module type
last CS31 module, CS31 module address

DGN0 WORD
Diagnosis word 0 of the 35 CS 91.

DGN1 WORD
Diagnosis word 1 of the 35 CS 91.

DGN2 WORD
Diagnosis word 2 of the 35 CS 91.

DGN3 WORD
Diagnosis word 3 of the 35 CS 91.

DGN4 WORD
Diagnosis word 4 of the 35 CS 91.

DGN5 WORD
Diagnosis word 5 of the 35 CS 91.

DGN6 WORD
Diagnosis word 6 of the 35 CS 91.

DGN7 WORD
Diagnosis word 7 of the 35 CS 91.

DGN8 WORD
Diagnosis word 8 of the 35 CS 91.

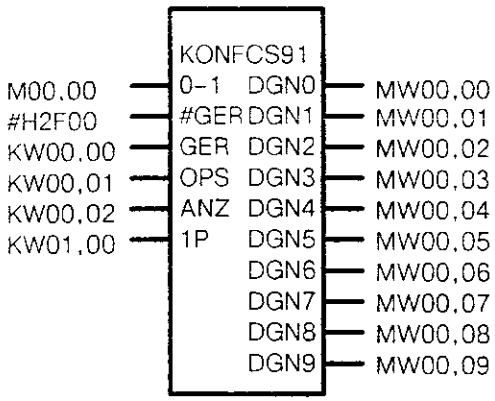
DGN9 WORD
Diagnosis word 9 of the 35 CS 91.

Note:

Refer to the 35 CS 91 module description for the function of the diagnosis words.

Example

FBD/LD

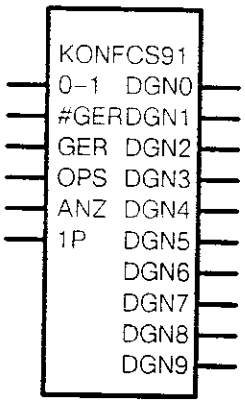


Module address 2000:F000H corresponding to 2F00:0000H has been selected for an example.
The following applies:
KW00,00 = $12032_D = 2F00_H$
KW00,01 for program N1: $-30720_D = 8800_H$
KW00,01 for program N2: $-28864_D = 8F40_H$

In case of KW00,02 = 5 the following applies:
KW01,00 contains: module type of 1st CS31 module
KW01,01 contains: module address of 1st CS31 module
. .
. .
. .
KW01,08 contains: module type of 5th CS31 module
KW01,09 contains: module address of 5th CS31 module

You have to enter the variables KW01,01 to KW01,09 with the variable editor into the variable list; they don't appear in FBD and IL.
Attention: When using the function 'Delete all unused variables', the constants KW01,01 to KW01,09 are deleted.
Remedy: Store these variables via 'Store block'; then you can read them in at any time.

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0-1	E	L	N	P	Y	0	0
#GER	E	W	N	P	Y	0	0
GER	E	W	N	P	Y	0	0
OPS	E	W	N	P	Y	0	0
ANZ	E	W	N	P	Y	0	0
1P	E	W	N	P	Y	0	0
DGN0	A	W	N	P	Y	0	0
DGN1	A	W	N	P	Y	0	0
DGN2	A	W	N	P	Y	0	0
DGN3	A	W	N	P	Y	0	0
DGN4	A	W	N	P	Y	0	0
DGN5	A	W	N	P	Y	0	0
DGN6	A	W	N	P	Y	0	0
DGN7	A	W	N	P	Y	0	0
DGN8	A	W	N	P	Y	0	0
DGN9	A	W	N	P	Y	0	0

CONVERSION OF PIXELS TO REAL UNITS OF MEASURE (OMS-F)

UMIMA

For length and area measurement, this function block converts the number of measured pixels to real units of measure.

This block is needed when using the video sensor OMS-F.

FBD/LD

IL

```
!BA    0
UMIMA
       WART
       PIWE
       MAWE
```

Parameters

WART	WORD	EW, AW, MW, KW	Dimension type to be converted
PIWE	WORD	EW, AW, MW, KW	Pixel quantity to be converted
MAWE	WORD	EW, AW, MW	Result of conversion in the real unit of measure

CE-Data

Runtime:	
Basic runtime:	121 µs
Additional runtime:	no
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

For length and area measurement, this function block converts the number of measured pixels to real units of measure.

This block is needed when using the video sensor OMS-F.

A length or area measured on a measured object is represented by the number of measured pixels. The block converts the number of these measured pixels to real units of measure (e. g. to mm).

A distinction must be made between conversion of a length dimension in X direction, a length dimension in Y direction or an area dimension. The applicable type of dimension is adjusted with WART.

When output, the result of conversion (MAWE) is rounded to a whole positive number. If the valid positive number range of 0...32767 is exceeded, 0 is output as the result. To convert the number of pixels to the real unit of measure, the block uses a scaling factor stored in the PLC. This scaling factor defines how many real units of measure correspond to 1 pixel length or width. The scaling factor is entered in the PLC beforehand by means of the X scaling PLC command (XS) in "teach-in mode".

Note:
Before the UMIMA block is called for the first time the scaling factor must be entered in the PLC in any case.

WART WORD

Dimension type to be converted

The type of dimension to be converted is specified with the operand specified at the WART input.
The following applies:

WART = 0: Length dimension in X direction
WART = 1: Length dimension in Y direction
WART = 2: Area dimension

CONVERSION OF PIXELS TO REAL UNITS OF MEASURE (OMS-F)

PIWE WORD
Pixel quantity to be converted.

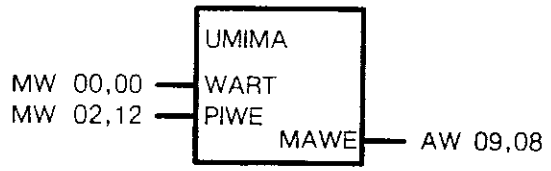
The number of pixels to be converted to the real unit of measure is specified with the operand specified at the PIWE input.

MAWE WORD
Result of conversion to the real unit of measure.

The result of conversion to the real unit of measure is available at this output.

Example

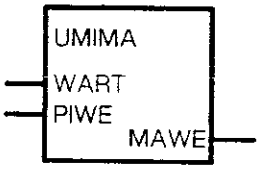
FBD/LD



IL

```
IBA    0
UMIMA
       MW  00,00
       MW  02,12
       AW  09,08
```

CE FBD Definition

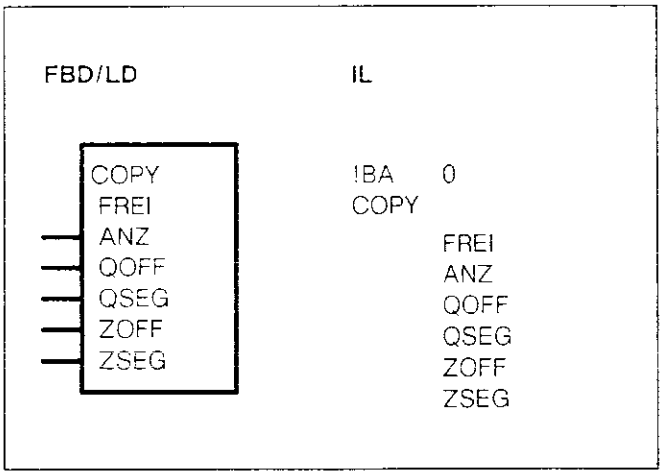


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
WART	E	W	N	P	Y	0	0
PIWE	E	W	N	P	Y	0	0
MAWE	E	W	N	P	Y	0	0

CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)
00001	UMIMA			
00002		PP 0	WART	Input WORD (X/Y direction, area)
00003		PP 0	PIWE	Input WORD (pixel value)
00004		PP 0	MAWE	Output WORD (result of conversion)

This function block copies n words from a source memory area into a target memory area.



Parameters

FREI	BINARY	E,A,M,K,S	Block enable
ANZ	WORD	EW,AW,MW,KW	Quantity (n) of words to be copied
QOFF	WORD	EW,AW,MW,KW	Offset address of the start of the source area
QSEG	WORD	EW,AW,MW,KW	Segment address of the start of the source area
ZOFF	WORD	EW,AW,MW,KW	Offset address of the start of the target area
ZSEG	WORD	EW,AW,MW,KW	Segment address of the start of the target area

CE Data

Runtime:
- Basic runtime: 59 µs
- Additional runtime: 2 µs per copied word

Output updating: not applicable
Number of historical values: none
Available as of: ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block copies n words from a source memory area into a target memory area.

The contents of the source memory area are not changed.

In each case, the start of the source and target memory areas is specified at the block's inputs by means of the offset and segment addresses. These are planned as variables or as indirect constants.

The inputs and outputs can neither be duplicated nor inverted/negated.

FREI BINARY
Block enable
FREI = 0 -> The block is not processed
FREI = 1 -> The block is processed

ANZ WORD
Quantity n of words to be copied.
The following applies: $0 \leq n \leq +8000H$
n = 0: No copying
n = 8000H: A whole segment (64 kbytes) is copied

QOFF WORD
Offset address of the start of the source area

QSEG WORD
Segment address of the start of the source area

ZOFF WORD
Offset address of the start of the target area

ZSEG WORD
Segment address of the start of the target area

Example:
128 words are to be copied from address 4000H:6 to address 5000H:4.

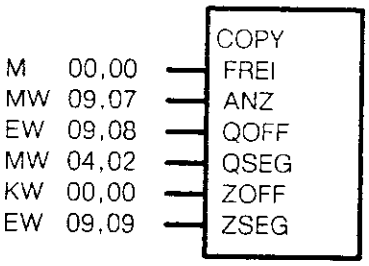
Number of words to be copied:	128
Offset address of the source memory:	6
Segment address of the source memory:	4000H
Offset address of the target memory:	4
Segment address of the target memory:	5000H

Note:

If the COPY function block is located in the PLC program with the lower priority, programming can be interrupted by the program with the higher priority. An interrupt can be prevented by planning the COPY block in the PLC program with the higher priority.

Example

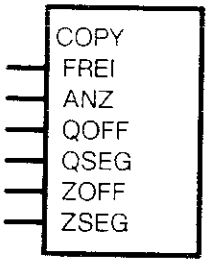
FBD/LD



IL

```
!BA    0
COPY
       M    00,00
       MW   09,07
       EW   09,08
       MW   04,02
       KW   00,00
       EW   09,09
```

CE FBD Definition



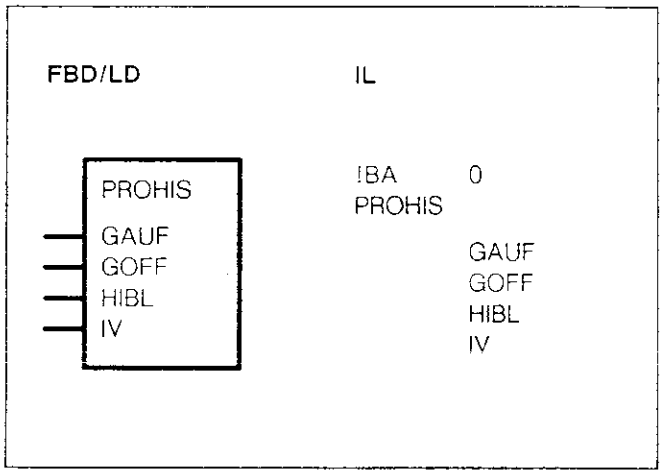
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
ANZ	E	W	N	P	Y	0	0
QOFF	E	W	N	P	Y	0	0
QSEG	E	W	N	P	Y	0	0
ZOFF	E	W	N	P	Y	0	0
ZSEG	E	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)	
00001	COPY				
00002		PP 0	FREI	Enable	BINARY
00003		PP 0	ANZ	Number of words	WORD
00004		PP 0	QOFF	OFFSET source	WORD
00005		PP 0	QSEG	SEGMENT source	WORD
00006		PP 0	ZOFF	OFFSET target	WORD
00007		PP 0	ZSEG	SEGMENT target	WORD

This function block serves to select a gray scale range of which a histogram analyis (gray scale distribution) is to be made.

This function block is needed when using the video sensor OMS-F.



Parameters

GAUF	WORD	EW, AW, MW, KW	Gray scale resolution, 64 or 128 steps
GOFF	WORD	EW, AW, MW, KW	Lowest gray scale value of the gray scale range to be analysed
HIBL	WORD	EW, AW, MW, KW	Length of the gray scale range to be analysed
IV	WORD	EW, AW, MW, KW	Number of the 35 IV 90 module

CE Data

Runtime:	
Basic runtime:	1.11 ms
Additional runtime:	no
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block serves to select a gray scale range of which a histogram analyis (gray scale distribution) is to be made.

The gray scale range to be analysed is defined by its lowest gray scale value (GOFF) and by the length of the range (HIBL).

The block subdivides the gray scale range to be analysed into 8 subranges of an identical size. On the 35 IV 90, a counter is allocated to each subrange in the following order:

Subrange 1 -> counter 1
Subrange 2 -> counter 2
.
.
Subrange 8 -> counter 8

Any remainder resulting from splitting of the analysed range into the 8 subranges is ignored. That is to say, the actually analysed range is smaller by the remainder than the one specified by GOFF and HIBL.

Example:

Analysed gray scale range:
Gray scale 20 ... gray scale 56

This results in:
GOFF = 20
HIBL = 37

Subdivision into 8 subranges: 37 : 8 = 4 remainder 5
4 gray tones are assigned to each of the 8 counters:
gray tones 20,21 22 and 23 are assigned to counter 1;
gray tones 24,25,26 and 27 are assigned to counter 2;
.
.
.
.
gray tones 48,49, 50 and 51 are assigned to counter 8.
The last 5 gray tones in the range (52, 53, 54, 55 and 56) are ignored during the course of measurement.

Note:
It is therefore expedient only to chose values for the length of the gray tone (HIBL) which are integral multiples of 8.

During picture recording, the counters count how often the gray tones assigned to them occur.

Note: The window required for the histogram analysis must be defined beforehand as measurement window 1 on the 35 IV 90 module (PROWi).

If the counters have been preset beforehand with event values (offsets), the status register contains information about which counters have elapsed (event occurrence). See also the PROOFF block.

GAUF WORD
Gray tone resolution in 64 or 128 gray tone steps.

When selecting a gray tone resolution (GAUF) between 64 or 128 steps, in relation to the former (64) make sure that the jumper X11 is set accordingly on the 35 KI 90 (see also OMS-F description of functions, Chapter 4.1.1).

GOFF WORD
Value for the low gray tone limit.

The value for the lowest gray tone of the analysed range is specified with the operand at this input.

HIBL WORD
Gray tone range length.

The value for the length (number of gray tone values) of the analysed range is specified with the operand at this input.

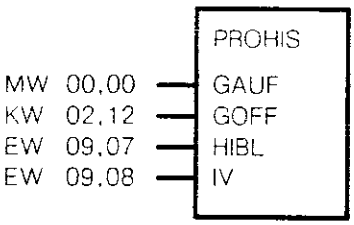
GOFF + HIBL – 1 specifies the top limit of the gray tone range.
Value range: HIBL $\geq$ 8

IV WORD
Number of the 35 IV 90 module.

The number of the required 35 IV 90 module with which measurement is to take place is specified with the operand at this input.
Value range: 0...5

Example

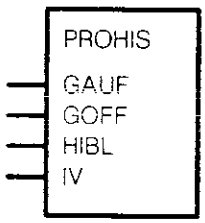
FBD/LD



IL

```
!BA     0
PROHIS
        MW 00,00
        KW 02,12
        EW 09,07
        EW 09,08
```

CE FBD Definition

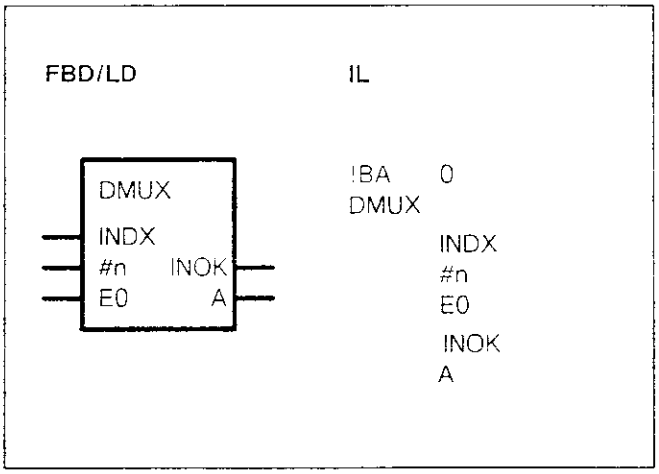


Param.	Param. Gorup	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
GAUF	E	W	N	P	Y	0	0
GOFF	E	W	N	P	Y	0	0
HIBL	E	W	N	P	Y	0	0
IV	E	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	PROHIS			
00002		PP 0	GAUF	Input WORD (gray scale resolution)
00003		PP 0	GOFF	Input WORD(gray tone offset)
00004		PP 0	HIBL	Input WORD (histogram length)
00005		PP 0	IV	Input WORD (No. of the 35 IV 90)

This function block connects one of the inputs E0...En-1 to the output A depending on the input INDX.



Parameters

INDX	WORD	EW, AW, MW, KW	Index input
#n	DIRECT CONSTANT	#, #H	Quantity n of word inputs E0 ... En-1
E0	WORD	EW, AW, MW, KW	Word input capable of duplication
INOK	BINARY	A, M	Range monitoring of the index input
A	WORD	AW, MW	Word output to which one of the inputs E0...E_{n-1} is switched through.

CE Data

Runtime:	
Basic runtime:	36 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block connects one of the inputs E0...En-1 to the output A depending on the input INDX.

The value at the input INDX is monitored for validity. The output A is set to 0 if the word input INDX is not within the valid range.

Relationship between E0...En-1, INDX and A:

The input INDX is used to define which of the inputs E0...En-1 is connected to the output A.

The following applies:

INDX = 1: E0 -> A
INDX = 2: E1 -> A
INDX = 3: E2 -> A
: : : :
INDX = n: E_{n-1} -> A

where: $1 \leq INDX \leq n \leq 32767$ (theoretically)

The output A is set to 0 if the input INDX = 0.

INDX WORD

Index input for selection of one of the inputs E0...En-1.

The following applies:
$1 \leq INDX \leq n$ (Number of inputs E0...En-1)

Note:
INDX = 0 -> Initialization of the output A with 0.

#n DIRECT CONSTANT
Quantity n of word inputs E0...En-1

E WORD
Input E0 ... E_{n-1} capable of duplication

One of the inputs E0 ... E_{n-1} is connected to the output A.

INOK BINARY
Range monitoring for the input INDX

The output INOK indicates whether or not the input INDX is within the valid range. The outputs INOK and A are set to 0 if the word input INDX is not within the valid range.

The following applies:
INOK = 1 -> INDX in the valid range
INOK = 0 -> INDX in the invalid range -> A = 0

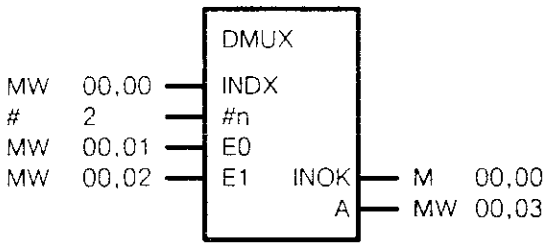
Valid range for the index: $1 \leq INDX \leq n$

A WORD
Output to which one of the inputs E0...En-1 is switched through.

By means of the input INDX, one of the inputs E0...En-1 is selected and its value is allocated to the output A. The output A is set to 0 if the word input INDX is not within the valid range.

Example

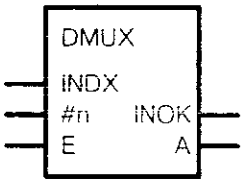
FBD/LD



IL

```
!BA    0
DMUX
       MW    00,00
       #     2
       MW    00,01
       MW    00,02
       M     00,00
       MW    00,03
```

CE FBD Definition

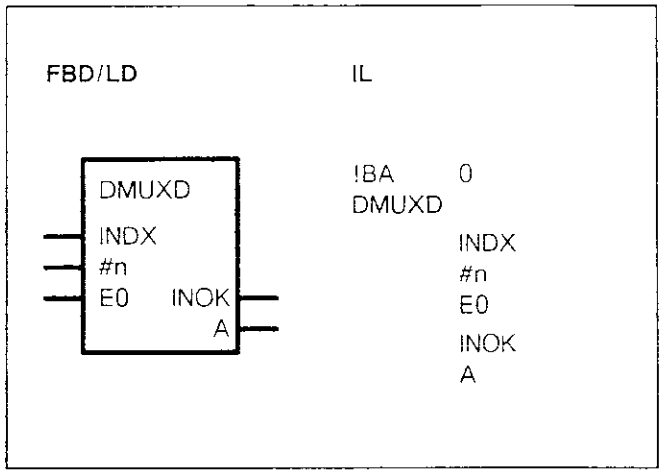


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
INDX	E	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
E	E	W	N	P	Y	1	0
INOK	A	L	N	P	Y	0	0
A	A	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)	
00001	DMUX				
00002		PP 0	INDX	Index	(WORD)
00003		PP 0	#n	# Quantity	(#)
	[	1			
00004		PP 1	E	Input	(WORD)
	]	1			
00005		PP 0	INOK	Output 'index valid'	(BINARY)
00006		PP 0	A	Output	(WORD)

This function block connects one of the inputs E0...En-1 to the output A depending on the input INDX.



Parameters

INDX	WORD	EW, AW, MW, KW	Index input
#n	DIRECT	#, #H	Quantity n of the double word inputs E0 ... E_{n-1}
E0	DOUBLE WORD	MD, KD	Double word inputs E0 ... E_{n-1}; the input E0 is capable of duplication.
NOK	BINARY	A, M	Range monitoring of the index input
A	DOUBLE WORD	MD	Double word output to which one of the inputs E0 ... E_{n-1} is switched through.

CE Data

Runtime:	
Basic runtime:	38 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block connects one of the inputs E0...En-1 to the output A depending on the input INDX.

The value at the input INDX is checked for validity. The output A is set to 0 if the word input INDX is not within the valid range.

Relationship between E0 ... E_{n-1}, INDX and A:

The input INDX is used to define which of the inputs E0 ... E_{n-1} is connected to the output A.

The following applies:

INDX = 1 : E0 -> A
INDX = 2 : E1 -> A
INDX = 3 : E2 -> A
: : : :
INDX = n : E_{n-1} -> A

where: $1 \leq INDX \leq n \leq 32767$ (theoretically)

Note: The output A is is set to 0 if the input INDX = 0.

INDX WORD

Index input for selection of one of the inputs E0 ... E_{n-1}.

The following applies:
$1 \leq INDX \leq n$ (number of inputs E0 ... E_{n-1})

Note:
INDX = 0 -> Initialization of the output A with 0.

#n DIRECT CONSTANT

Quantity n of double word inputs E0 ... E_{n-1}.

E DOUBLE WORD

Input E0 ... E_{n-1} capable of duplication.

One of the n inputs E0 ... E_{n-1} is connected to the output A.

INOK BINARY

Range monitoring for the input INDX

The output INOK indicates whether or not the input INDX is within the valid range. The outputs INOK and A are set to 0 if the word input INDX is not within the valid range.

The following applies:
INOK = 1 -> INDX within the valid range
INOK = 0 -> INDX in the invalid range -> A = 0

Valid range for the index: $1 \leq INDX \leq n$

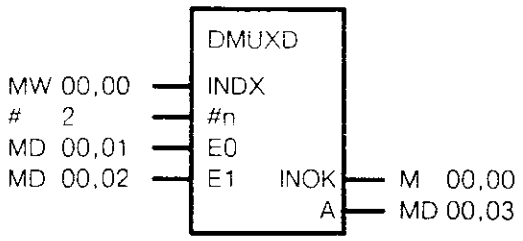
A DOUBLE WORD

Output to which one of the n inputs E0...En-1 is switched through.

By means of the input INDX, one of the inputs E0...En-1 is selected and its value is allocated to the output A. The output A is set to 0 if the input INDX is not within the valid range.

Example

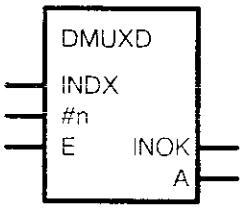
FBD/LD



IL

```
!BA     0
DMUXD
        MW 00,00
        #  2
        MD 00,01
        MD 00,02
        M  00,00
        MD 00,03
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
INDX	E	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
E	E	D	N	P	Y	1	0
INOK	A	L	N	P	Y	0	0
A	A	D	N	P	Y	0	0

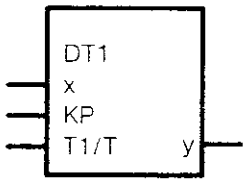
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	DMUXD			
00002		PP 0	INDX	Index (WORD)
00003		PP 0	#n	Quantity (#)
	[	1		
00004		PP 1	E	Input (DOUBLE WORD)
	]	1		
00005		PP 0	INOK	Output 'index valid' (BINARY)
00006		PP 0	A	Output (DOUBLE WORD)

DIFFERENTIATOR WITH DELAY OF THE 1ST ORDER — DT1

The controlled variable x is multiplied by the proportional coefficient KP. The proportional coefficient is specified as a percentage. If the controlled variable no longer changes, the manipulated variable y moves towards the value 0 in an e function. The time constant T1 specifies the time when the step response has dropped to approximately 37% of its initial value. The value has dropped below 1% after 5*T1.

FBD/LD



IL

```
!BA   0
DT1
      x
      KP
      T1/T
      y
```

Parameters

x	WORD	EW, AW, MW, KW	Controlled variable
KP	WORD	EW, AW, MW, KW	Proportional coefficient, specified as a percentage
T1/T	WORD	EW, AW, MW, KW	Time constant scaled to cycle time
y	WORD	AW, MW	Manipulated variable

CE Data

Runtime:	
Basic runtime:	200 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	4 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

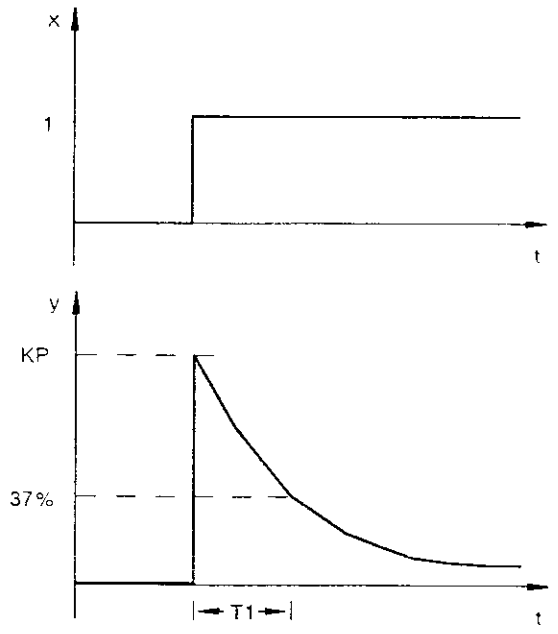
The controlled variable x is multiplied by the proportional coefficient KP. The proportional coefficient is specified as a percentage. If the controlled variable no longer changes, the manipulated variable y moves towards the value 0 in an e function. The time constant T1 specifies the time when the step response has dropped to approximately 37% of its initial value. The value has dropped below 1% after 5*T1.

The inputs and the output can neither be duplicated nor negated.

Transfer function:

$$F(s)=\frac{KP * s * T1}{s * T1 + 1}$$

Transfer function:



x WORD
The operand for the controlled variable (input value for the DT1 function) is specified at the input x.

KP WORD
The proportional coefficient is specified at the input KP. This value is specified as a percentage (since version V6; GJP5122900 R301) and can be positive or negative.

Example:

1	=	1	Percent
55	=	55	Percent
100	=	100	Percent
1000	=	1000	Percent
−100	=	−100	Percent

100 percent means that the input x is *not* influenced by the proportional coefficient.

T1/T WORD
The time constant of the DT1 function is specified at the input T1/T. To do this, the time constant T1 must be scaled to the PLC cycle time T.

The following applies to T1/TZ:

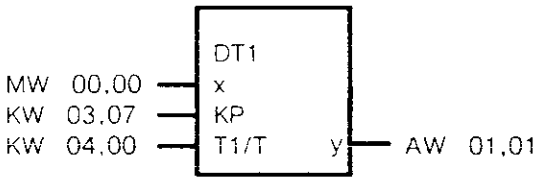
$10 \leq T1/TZ \leq 32767 \rightarrow T1 > 10 * TZ$

If a negative value is specified for T1/T, then the PLC automatically sets T1/T to the maximum positive value +32767.

y WORD
The manipulated variable (output value of the DT1 function) is output through the output y.

Example

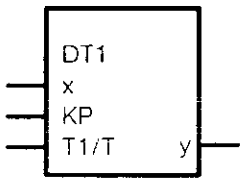
FBD/LD



IL

```
!BA   0
DT1
      MW  00.00
      KW  03.07
      KW  04.00
      AW  01.01
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
x	E	W	N	P	Y	0	0
KP	E	W	N	P	Y	0	0
T1/T	E	W	N	P	Y	0	0
y	A	W	N	P	Y	0	0

CE IL Definition

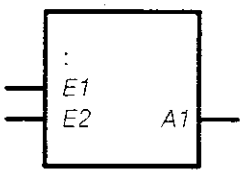
00000	!BA	0	Nr	Block No. (preset to 0)
00001	DT1			
00002		PP 0	x	Input WORD (controlled variable)
00003		PP 0	KP	Input WORD (proportional coefficient)
00004		PP 0	T1/T	Input WORD (time constant)
00005		PP 0	y	Output WORD (manipulated variable)

DIVISION :

The value of the operand at the input E1 is divided by the value of the operand at the input E2 and the result is allocated to the operand at the output A1.

The result of division is always a whole number. Rounding (inclusion of the digits after the decimal point) does not take place.

FBD/LD



IL

```
!   E1
:   E2
=   A1
```

Parameters

E1	WORD	EW, MW, AW, KW	Dividend
E2	WORD	EW, MW, AW, KW	Divisor
A1	WORD	MW, AW	Result (quotient)

CE Data

Runtime:	
Basic runtime:	< 31 µs
Additional runtime:	24 µs per additional input (E3 ... En).
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The value of the operand at the input E1 is divided by the value of the operand at the input E2 and the result is allocated to the operand at the output A1.

The result of division is always a whole number. Rounding (inclusion of the digits after the decimal point) does not take place.

The input E2 can be duplicated (E2...En). The following applies if it is duplicated:

E1 : E2 : E3 ... : En = A1.

All inputs and the output can be negated.

Number Range

Integer word (16 bits)

The following applies here particularly to the *non-negated* inputs E1 and E2:
Lower limit: 8000 H (-32768)

The following generally applies:

- Lower limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In twos complement arithmetic, the value 8000H (-32768) lies *outside* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations on the part of the user or
- by reading in from outside the PLC or
- by an indirect word constant

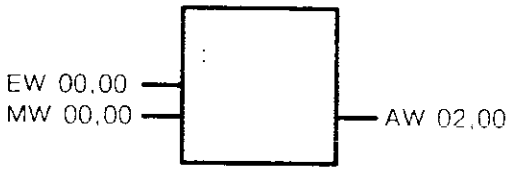
under no circumstances may a negation or substraction be done on this value.

A permissible value is generated again by an allocation (=), addition (+), multiplication (*) or division (:).

In the event of allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

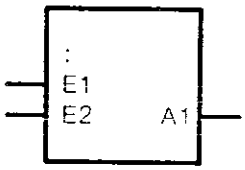
FBD/LD



IL

```
!   EW 00,00
    MW 00,00
=   AW 02,00
```

CE FBD Definition



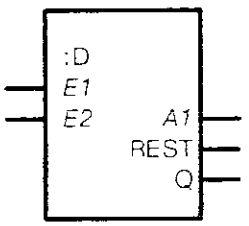
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	Y	P	N	0	0
E2	E	W	Y	P	N	1	0
A1	A	W	Y	P	N	0	0

CE IL Definition

00000	!	PP 0	E1	Input WORD
	[	1		
00002	:	PP 1	E2	Input WORD (capable of duplication)
	]	1		
00005	=	PP 0	A1	Output WORD

The value of the operand at the input E1 is divided by the value of the operand at the input E2 and the result is allocated to the operand at the output A1, the remainder being allocated to the operand at the output REST. If a remainder is produced, the result will always be rounded down. If the result lies outside of the permissible number range, it will be limited to the maximum or minimum value of the number range. If limiting has taken place, a 1 signal is allocated to the binary operand at the output Q and the value 0 is allocated to the output REST. If no limiting has taken place, a 0 signal is allocated to the binary operand at the output Q.

FBD/LD



IL

```
!BA   0
DIVD
      E1
      E2
      A1
      REST
      Q
```

Parameters

E1	DOUBLE WORD	MD, KD	Dividend
E2	DOUBLE WORD	MD, KD	Divisor
A1	DOUBLE WORD	MD	Result (quotient)
REST	DOUBLE WORD	MD	Rest
Q	BINARY	A, M	Result limited

CE Data

Runtime:	
Basic runtime:	300 ... 324 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the operand at the input E1 is divided by the value of the operand at the input E2 and the result is allocated to the operand at the output A1, the remainder being allocated to the operand at the output REST. If a remainder is produced, the result will always be rounded down. If the result lies outside of the permissible number range, it will be limited to the maximum or minimum value of the number range. If limiting has taken place, a 1 signal is allocated to the binary operand at the output Q and the value 0 is allocated to the output REST. If no limiting has taken place, a 0 signal is allocated to the binary operand at the output Q.

Division by "zero" is therfore also signalled at the binary output Q.

The inputs and outputs can neither be duplicated nor negated.

Remainder handling

If division results in a remainder, this is available at the double word output REST. The result of division is always rounded down if a remainder occurs.

Example:
3 : 3 = 1 Remainder 0
4 : 3 = 1 Remainder 1
5 : 3 = 1 Remainder 2
6 : 3 = 2 Remainder 0

As the remainder is available at the output REST, the user can compare this to the divisor and can round the result at the output A1 to suit his own requirements.

Example:

Remainder ≥ divisor/2 -> round up the result at A1.

Division by "zero"

If the divisor has the value "zero", the positive or negative limit of the number range is allocated to the output A1.

The following applies to division by "zero":

A1 = -2 147 483 647 (8000 0001 H) if the dividend is negative.

A1 = +2 147 483 647 (7FFF FFFF H) if the dividend is positive.

REST = 0	Output for the remainder
Q = 1	Output to signal that the value at the output A1 has been limited

Invalid result value

If the invalid value 8000 0000 H is the result of division, this will be corrected to the permissible limit 8000 0001 H (-2 147 483 647), the binary output Q will be set to the value 1 and the output REST will be set to the value 0.

Number range

Integer double word (32 bits)

Here, the following particularly applies to the inputs E1 and E2:

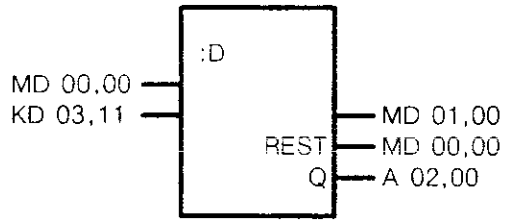
- Low limit: 8000 0000 H -2 147 483 648

The following generally applies:

• Low limit:	8000 0001 H	-2 147 483 647
• High limit:	7FFF FFFF H	+2 147 483 647
• Inadmissible value:	8000 0000 H	---

Example

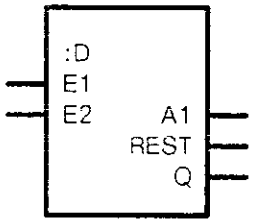
FBD/LD



IL

```
!BA    0
DIVD
       MD 00,00
       KD 03,11
       MD 01,00
       MD 00,00
       A  02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
E2	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0
REST	A	D	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

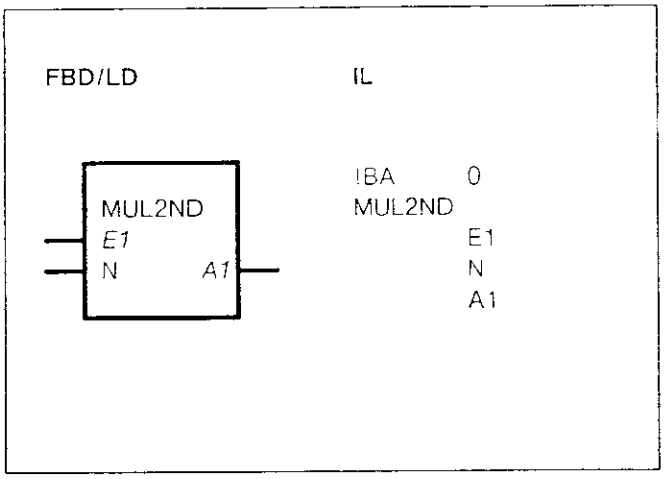
00000	!BA	0	Nr	Block No. (preset to 0)
00001	DIVD			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	E2	Input DOUBLE WORD
00004		PP 0	A1	Output DOUBLE WORD
00005		PP 0	REST	Output DOUBLE WORD
00005		PP 0	Q	Output BINARY

The value of the operand at the input E1 is shifted N times and bit-by-bit.

If the value at the input N is positive, shifting takes place to the left. For each shift by 1 bit position, this corresponds to multiplying the current value by 2.

If the value at the input N is negative, shifting takes place to the right. For each shift, this corresponds to division of the current value by 2.

The result is allocated to the operand at the output A1.



Parameters

E1	DOUBLE WORD	MD, KD	Input operand
N	WORD	EW, AW, MW, KW	Quantity
A1	DOUBLE WORD	MD	Result

CE Data

Runtime:	
Basic runtime:	34 µs
Additional runtime:	6 µs per shift
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R101

Description

The value of the operand at the input E1 is shifted N times and bit-by-bit.

If the value at the input N is positive, shifting takes place to the left. For each shift by 1 bit position, this corresponds to multiplying the current value by 2.

If the value at the input N is negative, shifting takes place to the right. For each shift, this corresponds to division of the current value by 2.

The result is allocated to the operand at the output A1.

Meaningful range for N: $-30 \leq N \leq +30$

If N = 0, the value at the input E1 is passed directly to the output A1.

Sign of the value at the input E1:

The sign of the value E1 is not influenced by the shift operation. That is to say, the sign of the output value is always identical to that of the input value.

Left shifting (Multiplication):

When the value at the input is shifted to the left, the bit 0 released in each case is filled up with 0. The sign bit (bit 31) is not changed because setting to the limit of the number range takes place beforehand.

Limiting of the value at the output A1 during left shifting:

- The following applies to positive values at the input E1:

If bit 30 is assigned a "1" and if shift operations still have to be carried out as the result of the value at the input N, these will no longer be executed. Instead, the output will be set to the limit of the postivie number range. That is to say, the limit has been reached in any case at the latest after shifting 30 times.
Output A1 = +2 147 483 647 (7FFF FFFF H).

- The following applies to negative values at the input E1:

If bit 30 is assigned a "0" and if shift operations still have to be carried out as the result of the value at the input N, these will no longer be executed. Instead, the output will be set to the limit of the postivie number range. That is to say, the limit has been reached in any case at the latest after shifting 30 times.
Output A1 = -2 147 483 647 (8000 0001 H).

Shifting to the right (division):

When shifting to the right, each bit is moved to the right by one position. At the same time, the sign bit (bit 31) always retains its value. The bit released (bit 30) is filled up with the value of the sign bit in each case.

Limiting of the value at the output when shifting to the right:

- The following applies to positive values at the input:

If now only bit 0 has a "1" and if shift operations still have to be executed as the result of the value at the input N, the output will be set to the value 0. That is to say, the value 0 is reached in any case at the latest after shifting 30 times.
Output A1 = 0.

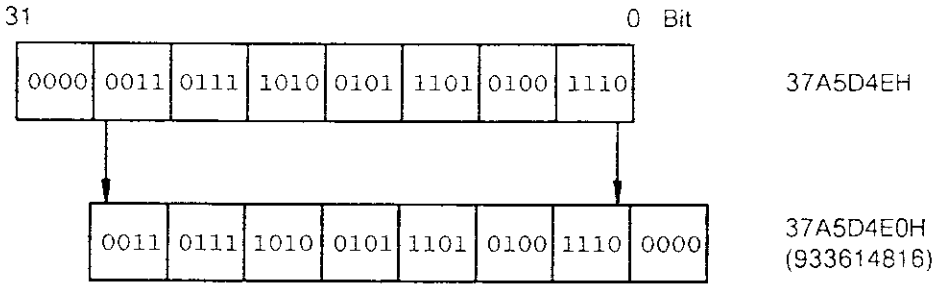
- The following applies to negative values at the input E1:

If bits 0...31 have a "1" as the result of shifting, the limit (-1) has been reached. Further shifts have no effect. That is to say, the value -1 has been reached in any case at the latest after shifting 31 times.
Output A1 = -1 (FFFF FFFF H).

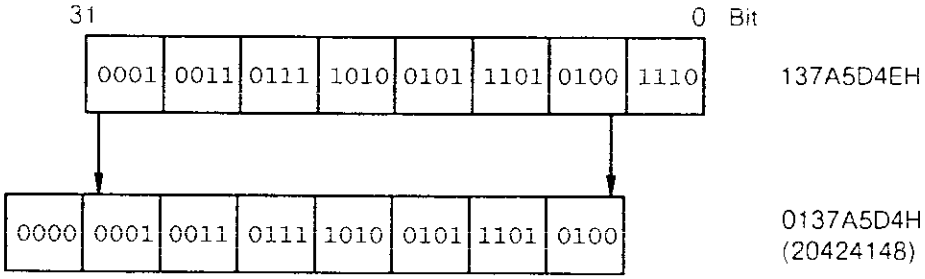
The inputs and the output can neither be duplicated nor negated

Examples

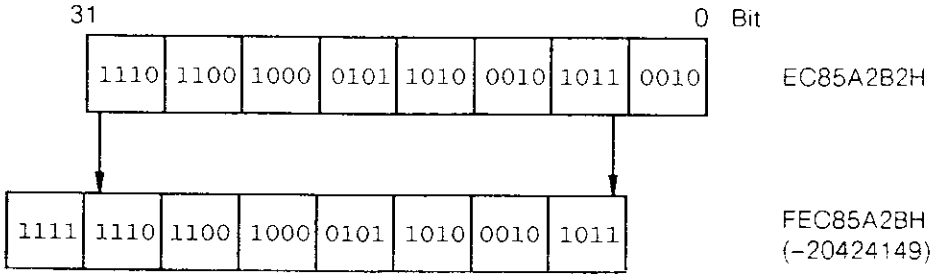
1. Input value E1 = 58350926 (37A5D4EH)
   Exponent N = 4 --> 4 * Left shift



2. Input value E1 = 326786382 (137A5D4EH)
   Exponent N = - 4 --> 4 * Right shift



3. Input value E1 = -326786382 (EC85A2B2H)
   Exponent N = - 4 --> 4 * Right shift



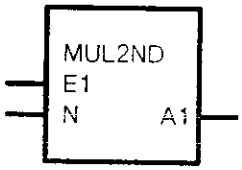
Example

FBD/LD

IL

```
!BA      0
MUL2ND
         MD 00,00
         KW 03,11
         MD 01,00
```

CE FBD Definition



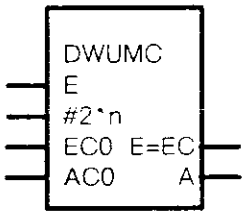
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
N	E	W	N	P	Y	0	0
A1	A	D	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MUL2ND			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	N	Input WORD (number of shifts)
00004		PP 0	A1	Output DOUBLE WORD

This function block compares the value of the operand at the input E to the reference values of the operands at the inputs EC0...ECn-1. If the input E agrees with at least one of the reference values EC, the output E=EC is set to 1. The output A receives the value of the output code AC, which is allocated to the reference value EC found.

FBD/LD



IL

```
!BA     0
DWUMC
        E
        #2*n
        EC
        AC
        E=EC
        A
```

Parametes

E	DOUBLE WORD	MD, KD	Input
#2*n	DIRECT CONSTANT	#, #H, KW	Quantity n of reference values (multiplied by 2)
EC	DOUBLE WORD	MD, KD	Reference value; the input can be duplicated
AC	DOUBLE WORD	MD, KD	Output code; the input can be duplicated
E=EC	BINARY	A, M	Coincidence indication
A	DOUBLE WORD	MD	Output of the output code's value

CE Data

Runtime:	
Basic runtime:	53 µs
Additional runtime:	10 µs per entered reference value
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block compares the value of the operand at the input E to the reference values of the operands at the inputs EC0...ECn-1. If the input E agrees with at least one of the reference values EC, the output E=EC is set to 1. The output A receives the value of the output code AC, which is allocated to the reference value EC found.

An operand for the output code ACi is allocated to each reference value at the inputs ECi. Affiliation of EC to AC is recognizable by the index i. The index begins with 0 and is generated automatically in the event of duplication.

The number of inputs EC and AC must be specified as a direct constant at the input #2*n.

The inputs and outputs cannot be negated/inverted.

E DOUBLE WORD
The operand whose value is to be compared to the values of the n reference values (EC0...ECn-1) is specified at the input E.

#2*n DIRECT CONSTANT (#,#H)
The total number (2*n) of the reference values (EC0...ECn-1) and output codes (AC0...ACn-1) is specified at the input #2*n. This is specified as an indirect constant.

EC0...ECn-1 DOUBLE WORD
The input EC0 must be duplicated according to the number of reference values needed. The operands for the reference values are specified at the inputs EC0...ECn-1. The value of the operand at the input E1 is compared to the reference values.

AC0...ACn-1 DOUBLE WORD
By duplication of the input EC0, the input AC0 is automatically also duplicated.
The output codes are specified at the inputs AC0...ACn-1. The output code ACi is output through the output A if the input E agrees with one of the reference values ECi.

Affiliation between reference values and output codes:
EC0 <-> AC0
EC1 <-> AC1
. .
. .
ECn-1 <-> ACn-1

E=EC BINARY
Agreement between the operand value of the input E and one of the reference values is signalled at the output E=EC.

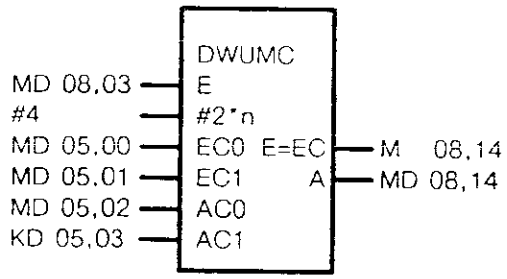
The following applies: E=EC = 0 -> No agreement
E=EC = 1 -> Agreement

A DOUBLE WORD
The output code ACi is output through the output A if the input E agrees with one of the reference values ECi.

The following applies: A = 0 -> No agreement
A = ACi -> Agreement

Example

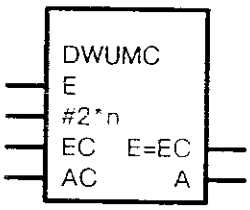
FBD/LD



IL

```
!BA    0
DWUMC
       MD 08,03
       #4
       MD 05,00
       MD 05,01
       MD 05,02
       KD 05,03
       M  08,14
       MD 08,14
```

CE FBD Definition



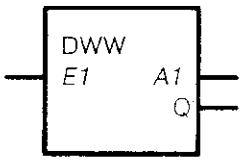
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	D	N	P	Y	0	0
#2*n	K	W	N	P	Y	0	0
EC	E	D	N	P	Y	1	0
AC	E	D	N	P	Y	1	0
E=EC	A	L	N	P	Y	0	0
A	A	D	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	DWUMC			
00002		PP 0	E	Input DOUBLE WORD (input value)
00003		PP 0	#2*n	# CONSTANT (number of EC and AC)
	[	1		
00004		PP 1	EC	Input DOUBLE WORD (reference value)
	]	1		
	[	1		
00005		PP 1	AC	Input DOUBLE WORD (output code)
	]	1		
00006		PP 0	E=EC	Output BINARY (input value=Ecode)
00007		PP 0	A	Output DOUBLE WORD (output value)

The value of the double word operand at the input E1 is converted to a word variable and the result is allocated to the word operand at the output A1.

FBD/LD



IL

```
!BA   0
DWW
      E1
      A1
      Q
```

Parameters

E1	DOUBLE WORD	MD, KD	Double word variable to be converted
A1	WORD	AW, MW	Result of conversion, word variable
Q	BINARY	A, M	Result limited

CE Data

Runtime:	
Basic runtime:	36 ... 38 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the double word operand at the input E1 is converted to a word variable and the result is allocated to the word operand at the output A1.

The result is limited to the maximum or minimum number range.
Max. number range: +32 767 (7FFFH)
Min. number range: -32 767 (8001H)

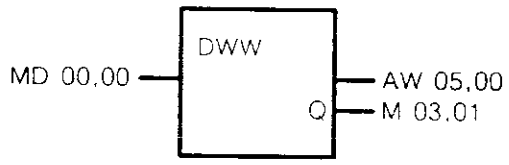
If limiting has taken place, a 1 signal is allocated to the binary operand at the output Q. If no limiting has taken place, a 0 signal is allocated to the binary operand at the output Q.

If the value of the operand at the input E1 lies outside of the permissible number range (value 8000 0000 H), the converted value is set to -32 767 (8001H).

The input and the outputs can neither be duplicated nor negated.

Example

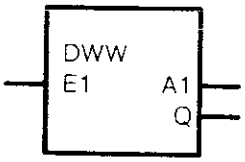
FBD/LD



IL

```
!BA    0
DWW
       MD 00,00
       AW 05,00
       M  03,01
```

CE FBD Definition



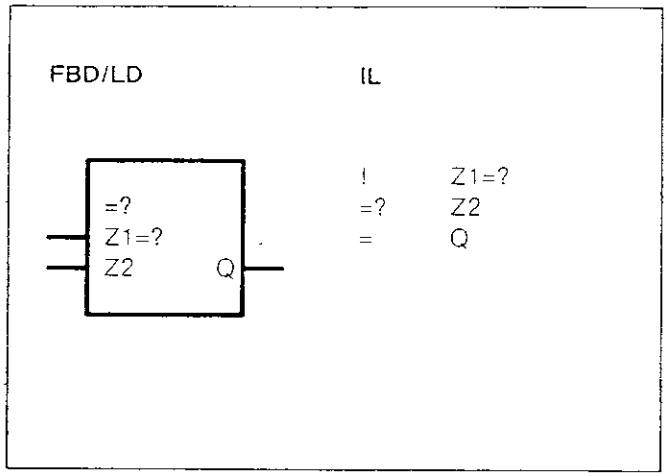
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
A1	A	W	N	P	N	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	DWW			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	A1	Output WORD
00004		PP 0	Q	Output BINARY

EQUAL =?

The value of the operand at the input Z1=? is compared to the value of the operand at the input Z2. If the value at Z1=? is equal to the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1=? is unequal to Z2.



Parameters

Z1=?	WORD	EW, MW, AW, KW	Value to be compared
Z2	WORD	EW, MW, AW, KW	Comparison value
Q	BINARY	A, M, S	Result of the comparison

CE Data

Runtime:	
Basic runtime:	< 12 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The value of the operand at the input Z1=? is compared to the value of the operand at the input Z2. If the value at Z1=? is equal to the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1=? is unequal to Z2.

The inputs can be negated but cannot be duplicated. The output can be inverted but cannot be duplicated.

Number range

Integer word (16 Bit)

The following specially applies here to the non-negated inputs:

- low limit 8000 H -32768
- high limit 7FFF H +32767

The following generally applies:

- low limit: 8001 H -32767
- high limit: 7FFF H +32767
- inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

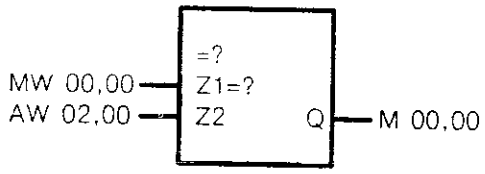
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

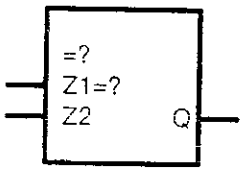
FBD/LD



IL

```
!     MW 00,00
=?    AW 02,00
=     M  00,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1=?	E	W	Y	P	Y	0	0
Z2	E	W	Y	P	Y	0	0
Q	A	L	Y	P	Y	0	0

CE IL Definition

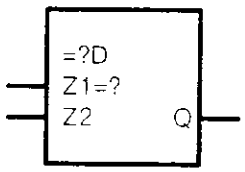
00000	!	PP 0	Z1=?	Input WORD
00002	=?	PP 0	Z2	Input WORD
00004	=	PP 0	Q	Output BINARY

EQUAL, DOUBLE WORD

=?D

The value of the operand at the input Z1=? is compared to the value of the operand at the input Z2. If the vaule at Z1=? is identical to the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1=? is unequal to Z2.

FBD/LD



IL

```
!BA   0
VGLD
      Z1=?
      Z2
      Q
```

Parametes

Z1=?	DOUBLE WORD	MD, KD	Value to be compared
Z2	DOUBLE WORD	MD, KD	Comparison value
Q	BINARY	A, M	Result of the comparison

CE Data

Runtime:	
Basic runtime:	35 ... 37 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the operand at the input Z1=? is compared to the value of the operand at the input Z2. If the vaule at Z1=? is identical to the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1=? is unequal to Z2.

The inputs can neither be duplicated nor negated. The output can neither be duplicated nor inverted.

Number range

Integer double word (32 Bit)

The following specially applies here to the inputs Z1=? and Z2:

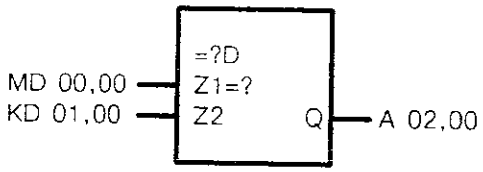
- low limit: 8000 0000 H -2 147 483 648
- high limit: 7FFF FFFF H +2 147 483 647

The following generally applies:

- low limit: 8000 0001 H -2 147 483 647
- high limit: 7FFF FFFF H +2 147 483 647
- inadmissible value: 8000 0000 H ---

Example

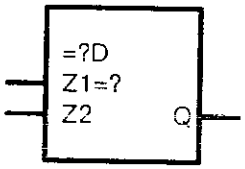
FBD/LD



IL

```
!BA   0
VGLD
      MD 00,00
      KD 01,00
      A  02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1=?	E	D	N	P	Y	0	0
Z2	E	D	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

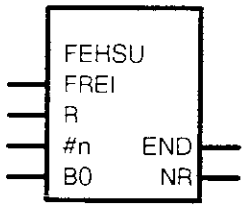
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	VGLD			
00002		PP 0	Z1=?	Input DOUBLEWORD
00003		PP 0	Z2	Input DOUBLEWORD
00004		PP 0	Q	Output BINARY

ERROR SEARCHER WITH AUTOMATIC DELETION — FEHSU

This function block searches through a list of binary variables (E, A, M, S) for set binary variables. If a set binary variable is found, its serial number referred to the list is indicated. The set binary variable at the input is deleted.

FBD/LD



IL

```
!BA     0
FEHSU
        R
        FREI
        #n
        B0
        END
        NR
```

Parameters

FREI	BINARY	E, A, M, S, K	Enable search
R	BINARY	E, A, M, S, K	Reset
#n	DIRECT CONSTANT	#, #H	Number of binary variables
B0	BINARY	E, A, M	List of binary variables; the input can be duplicated
END	BINARY	A, M	List end reached
NR	WORD	AW, MW	List number of the variables found

CE Data

Runtime:	
Basic runtime:	90 µs
Additional runtime:	10 µs per binary variable at the inputs B0...B_{n-1}
Output updating:	yes if FREI = 1
Number of historical values:	4 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block searches through a list of binary variables (E, A, M, S) for set binary variables. If a set binary variable is found, its serial number referred to the list is indicated. The set binary variable at the input is deleted.

The inputs and outputs cannot be negated/inverted. The input B0 is capable of duplication.

FREI BINARY
The search for a set binary variable is enabled with the FREI input.
FREI = 0 Search is not enabled
FREI = 1 Search is enabled.

Search:
The search is continued until
- a set binary variable is found or
- the end of the variable list is reached.

If a binary variable with the value 1 is found in the list, the search is ended, the variable is deleted and its list number is output through the output NR. The search is continued again when the block is called again (in the next program cycle). The binary variable directly following the one found is now examined first.
The search ends at the end of the list if no further set variable is found.
The value 1 is then allocated to the END output and the number of the variable found last during the search is indicated at the output NR.
If no set variable is found during a search from the start of the list, the search is automatically repeated from the start of the list when the block is called again. This takes place until a set variable is found.

R BINARY

By means of the input R, the function block is set to a defined state (reset) in order to be able to start a search from the start of the variable list.

R = 0 -> No reset of the block

R = 1 -> Reset of the block for preparation of a search from the start of the list

Note: Even when the block is called for the very first time after starting the PLC program, a reset *must* first of all be executed in order to create the correct marginal conditions for the first search.

A reset of the block results in the following:
- Output END = 0
- Output NR = 0
- The next search takes place from the start of the list

Priority:
The input R has higher priority than the input FREI, i. e. no search is executed as long as there is a 1 signal at the input R.

#n DIRECT CONSTANT

The number of binary variables planned at the inputs B0...Bn-1 is specified at the input #n. This is specified as a direct constant.

B0..Bn-1 BINARY

The input B0 can be duplicated (B0...Bn-1).
The list of binary variables to be examined is specified at the inputs B0...Bn-1. Owing to the reset mechanism, step operands (S) and indirect constants (K) are not allowed.

END BINARY

Whether or not the end of the specified variable list has been reached without a set binary variable having been found in the search just carried out is signalled at the END output. That is to say, if the last binary variable of the list is recognized and reported as being "set" during a search, the output END is not yet set to the value 1. Instead, this does not take place until the next processing cycle.

END = 0 -> List end not reached

END = 1 -> List end reached without a set binary variable having been found in the current search

NR WORD

The list number of the binary variable found last is output through the NR output.
The following affiliations apply:

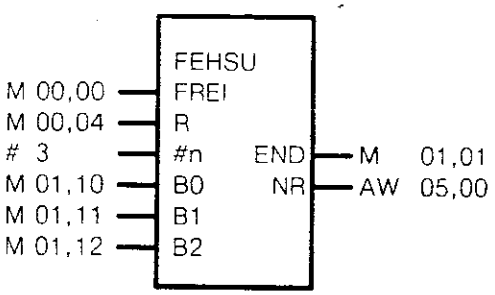
Variable at the input	List number
B0	1
B1	2
.	.
.	.
.	.
B_{n-1}	n

If the end of the list is reached during the search without a set variable having been found, the number of the variable last found continues to be output through the output NR.

Searching from the start of the list without a variable being found:
If no set binary variable is found during a search from the start of the list, the state 1 is assigned to the output END when the end of the list is reached and the value 0 is assigned to the output NR. When the block is called again in the next program cycle, the search automatically takes place again from the start of the list. Automatic searching from the start of the list continues until a set binary variable is found once.

Example

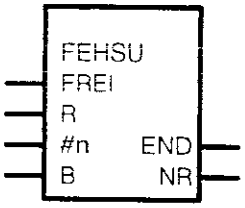
FBD/LD



IL

```
!BA   0
FEHSU
      M  00,04
      M  00,00
      #  3
      M  01,10
      M  01,11
      M  01,12
      M  01,01
      AW 05,00
```

CE FBD Definition

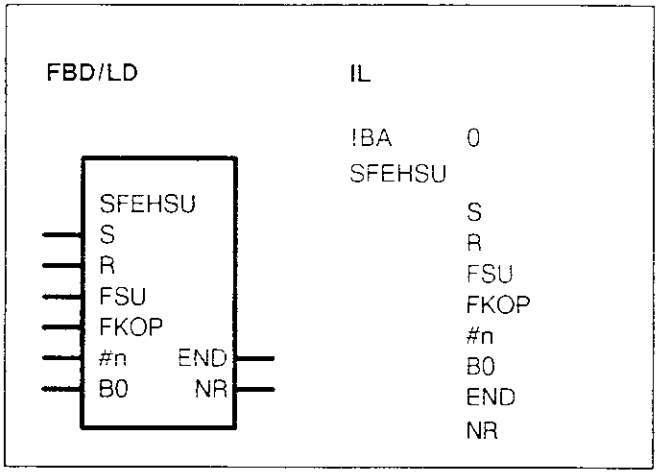


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
R	E	L	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
B	E	L	N	P	Y	1	0
END	A	L	N	P	Y	0	0
NR	A	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	FEHSU			
00002		PP 0	R	Input BINARY (Reset)
00003		PP 0	FREI	Input BINARY (enable)
00004		PP 0	#n	# CONSTANT (number of bits)
	[	1		
00005		PP 0	B	Input BINARY (element of the list of binary variables)
	]	1		
00006		PP 0	END	Ausgang BINÄR (list end reached)
00007		PP 0	NR	Ausgang WORT (No.)

This function block successively searches through a list of binary variables (E, A, M, S) for set binary variables. If a set binary variable is found, its number is output through the output NR. In doing so, the block does not directly search through the input list for the set binary variables, but through its image, which it stores in an internal list. After a set binary variable has been found in the internal list and its number has been output through the output NR, this binary variable is deleted from the internal list.



Parameters

S	BINARY	E, A, M, S, K	Set
R	BINARY	E, A, M, S, K	Reset
FSU	BINARY	E, A, M, S, K	Enable search
FKOP	BINARY	E, A, M, S, K	Enable copy
#n	DIRECT CONSTANT	#, #H	Number of binary variables
B0	BINARY	E, A, M, S, K	List of binary variables; the input can be duplicated
END	BINARY	A, M	List end reached
NR	WORD	AW, MW	List number of the variable found

CE Data

Runtime:	
Basic runtime:	83 µs
Additional runtime:	22 µs per input B0...B_{n-1}
Output updating:	yes
Number of historical values:	3 words + #n/16 words
Available as of:	ABB Procontic T320 V4 / 935 PC 82 R101 / 35 ZE 93 R101

Description

This function block successively searches through a list of binary variables (E, A, M, S) for set binary variables. If a set binary variable is found, its number is output through the output NR. In doing so, the block does not directly search through the input list for the set binary variables, but through its image, which it stores in an internal list. After a set binary variable has been found in the internal list and its number has been output through the output NR, this binary variable is deleted from the internal list.

The inputs and outputs cannot be negated. The input B0 can be duplicated (B0...Bn-1).

S BINARY

A 1 signal at the input S results in the following:

- All set binary variables of the input list are additionally entered in the internal list (ORed with the internal list).
- The block is prepared for a search from the start of the internal list, i. e. the internal list's pointer is set to its start.
- The output END (list end reached) is set to 1 and the output NR (number of the binary variable) is set to 0.
- If a set variable has been found during the previous search in the internal list, this is deleted from the internal list in order to avoid a double message.

R BINARY

- All set binary variables in the internal list are deleted when the input R has a 1 signal. The set binary variables at the block's inputs are *not* affected by this.
- The pointer to the internal list is *not* changed, i. e. a subsequent search begins as from the point to which the pointer of the internal list pointed before deletion.
- The output END is set to 1 and the output NR is set to 0.

FSU BINARY

- A 1 signal enables the search for set binary variables in the internal list.
- A 0 signal disables the search; in doing so, the old values at the outputs are reallocated during each cycle.
- If a binary variabe with the value 1 has been found in the internal list, the number of this set binary variable is output through the output NR. The binary variable is then cleared from the *internal* list. The output END (list end reached) is set to 0 if the binary variable is not the last one in the list.
- Numbering of the binary variables begins with 1.
- The output END retains the value 0 as long as the block has not reached the end of the internal list during the search.
- Each time it is called again, the block continues the search in the internal list, beginning with the next binary variable after the one found last. *Beforehand*, the binary variable found last is deleted from the internal list in order to avoid a double message.
- The following applies if the end of the internal list is reached during the search:
  - the output END (list end reached) is set to 1.
  - the number of the binary variable in the internal list found last is output through the output NR.
  - each further time the block is called, the internal list is searched through as from the point of the binary variable found last. The search ends when a set binary variable is found or when the end of the list is reached.
- If the last binary variable of the list is set and has been found during the search, the search stops there until a new search from the start of the internal list is prepared by means of the input S.
- If no set binary variable has been found during the course of a search from the start of the list, the output END (list end reached) is set to 1 and the output NR (number) retains the value 0. Each time the block is called again, the whole list is always searched through until a binary variable assumes the value 1 and is found.

FKOP BINARY

When the input FKOP has a 1 signal, all set binary variables of the input list are additionally entered in the internal list. At the same time, the binary variables already set in the internal list are retained. Updating is realized by "ORing" of the input list with the internal list.

Restart when searching after updating

Updating of the internal list has no influence on the next search. The search begins precisely at the point in the internal list where it would also have been begun if no updating had taken place.

#n DIRECT CONSTANT

The number of binary variables planned at the inputs B0...Bn-1 is specified at the input #n. This is specified as a direct constant.

Important: The quantity at the input #n *must* be an integral multiple of 16. Dummy operands may also be planned (e.g. K0.0 is specified at all inputs not needed) in order to keep to this stipulation.

B0...Bn-1 BINARY

The input B0 is capable of duplication (B0...Bn-1). The binary variables to be examined are specified at the inputs B0...Bn-1. The number of variables must always be an integral multiple of 16. To achieve this, assign K0.0 = 0 to inputs Bi that are not needed.

END BINARY

Whether or not the end of the list has been reached during the search is signalled at the output END.

END = 0 -> List end not reached
END = 1 -> List end reached

If the last variable of the list is set *and* has been found during the search just carried out, its number is output through the output NR and the value 1 is additionally allocated to the output END.

NR WORD
The list number of the variable found last is output through the output NR. The following affiliations apply:

Variable at the input	List number
B0	1
B1	2
.	.
.	.
.	.
B_{n-1}	n

If the end of the list is reached during the search without a new variable having been found, the number of the variable found at last continues to be output through the output NR.
The value 0 is output through the output NR if *no* variable is set in the list.

Priorities of the inputs S, R, FSU, FKOP

- The set input S has the highest priority. No other input is processed as long as the input S has a 1 signal.
- The reset input R has the second highest priority. The subsequent inputs are not processed as long as the input R has a 1 signal.
- The update input FKOP has priority over the enable search input FSU.

Updating of the internal list (input FKOP):

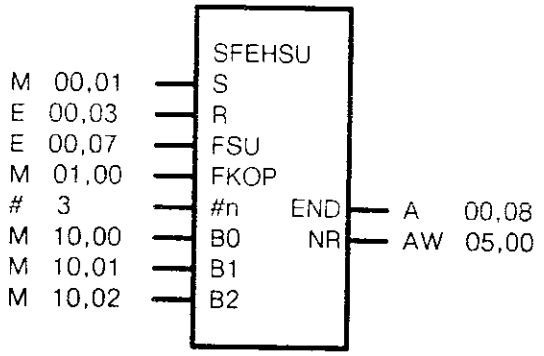
Case 1: Updating enabled (FKOP = 1)
If updating is enabled, all set binary input variables are additionally entered in the internal list. If the search is enabled at the input FSU, it is carried out immediately after updating of the internal list.

Case 2: Updating not enabled (FKOP = 0)
If updating is not enabled, the search is carried out immediately if enabled at the input FSU.

If the search is not enabled in both cases, the old values are output through the outputs END and NR.

Example

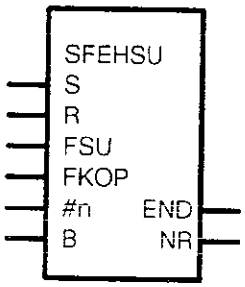
FBD/LD



IL

```
!BA      0
SFEHSU
         M  00,01
         E  00,03
         E  00,07
         M  01,00
         #  3
         M  10,00
         M  10,01
         M  10,02
         A  00,08
         AW 05,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
S	E	L	N	P	Y	0	0
R	E	L	N	P	Y	0	0
FSU	E	L	N	P	Y	0	0
FKOP	E	L	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
B0	E	L	N	P	Y	1	0
END	A	L	N	P	Y	0	0
NR	A	W	N	P	Y	0	0

CE IL Definition

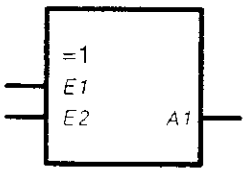
00000	!BA	0	Nr	Block No. (preset to 0)
00001	SFEHSU			
00002		PP 0	S	Input BINARY (Set)
00003		PP 0	R	Input BINARY (Reset)
00004		PP 0	FSU	Input BINARY (enable search)
00005		PP 0	FKOP	Input BINARY (enable copy)
00006		PP 0	#n	# CONSTANT (number of bits)
	[	1		
00007		PP 0	B	Input BINARY (element of the list of binary variables)
	]	1		
00008		PP 0	END	Output BINARY (list end reached)
00009		PP 0	NR	Output WORD (No.)

EXCLUSIVE OR

=1

This connection element realizes a logical EXCLUSIVE OR combination of the operands at the inputs. The result is allocated to the operand at the output.

FBD/LD



IL

```
!   E1
&N  E2
/N  E1
&   E2
=   A1
```

Parameters

E1	BINARY	E, M,A,S,K	Operand 1 of the XOR combination
E2	BINARY	E,M,A,S,K,	Operand 2 of the XOR combination
A1	BINARY	M,A,S	Result of the XOR combination

CE Data

Runtime:	
Basic runtime:	10.9 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801/ 35 ZE 93 R101

Description

This connection element realizes a logical EXCLUSIVE OR combination of the operands at the inputs. The result is allocated to the operand at the output.

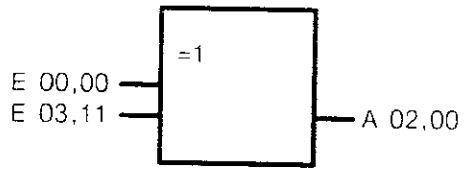
Inverting and duplication of the inputs and of the output is not possible.

Truth table:

E1	E2	A1
0	0	0
1	0	1
0	1	1
1	1	0

Example

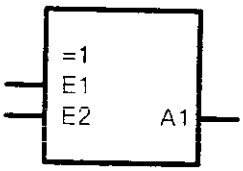
FBD/LD



IL

```
!     E 00,00
&N    E 03,11
/N    E 00,00
&     F 03,11
=     A 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	L	N	P	N	0	0
E2	E	L	N	P	N	0	0
A1	A	L	N	P	N	0	0

CE IL Definition

00000	!	PP 0	E1	Input BINARY
00002	&N	PP 0	E2	Input BINARY
00004	/N	PP 0	E1	Input BINARY
00006	&	PP 0	E2	Input BINARY
00008	=	PP 0	A1	Ouput BINARY

EXCLUSIVE OR COMBINATION, DOUBLE WORD — DWXOR

This function block generates the bit-by-bit XOR combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

FBD/LD

IL

```
!BA     0
DWXOR
        E1
        E2
        A1
```

Parameters

E1	DOUBLE WORD	MD, KD	Operand 1
E2	DOUBLE WORD	MD, KD	Operand 2
A1	DOUBLE WORD	MD	Result of the XOR combination

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block generates the bit-by-bit XOR combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

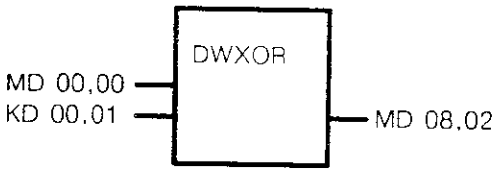
The inputs and the output can neither be duplicated nor negated nor inverted.

Example

E1	1.0.0.0	0.0.1.1	0.0.1.0	0.1.1.0	1.0.1.0	1.1.0.0	0.0.1.1	0.1.0.1
E2	1.0.0.1	0.1.1.0	0.0.1.0	1.1.1.1	1.1.1.1	0.0.0.0	0.1.1.0	1.1.0.0
A1	0.0.0.1	0.1.0.1	0.0.0.0	1.0.0.1	0.1.0.1	1.1.0.0	0.1.0.1	1.0.0.1

Example

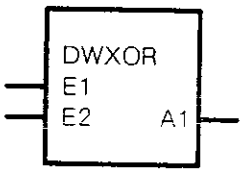
FBD/LD



IL

```
!BA     0
DWXOR
        MD 00.00
        KD 00.01
        MD 08.02
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
E2	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0

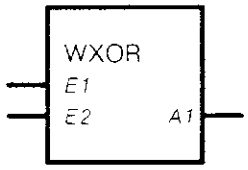
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	DWXOR			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	E2	Input DOUBLE WORD
00004		PP 0	A1	Output DOUBLE WORD

EXCLUSIVE OR COMBINATION, WORD — WXOR

This function block generates the bit-by-bit XOR combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

FBD/LD



IL

```
!BA   0
WXOR
      E1
      E2
      A1
```

Parameters

E1	WORD	EW, MW, AW, KW	Operand 1
E2	WORD	EW, MW, AW, KW	Operand 2
A1	WORD	MW, AW	Result of the XOR combination

CE Data

Runtime:
- Basic runtime: 28 µs
- Additional runtime: ---

Output updating: yes
Number of historical values: none
Available as of: ABB Procontic T320 V3 / 935 PC 81 R701, R801/ 35 ZE 93 R101

Description

This function block generates the bit-by-bit XOR combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

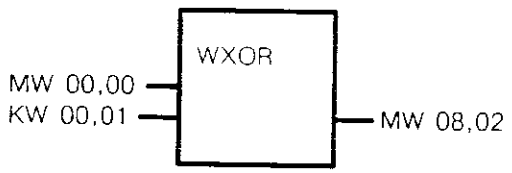
The inputs and the output can neither be duplicated nor negated nor inverted.

Example

E1	0.0.0.0	0.0.1.1	0.0.1.0	0.1.1.0
E2	1.0.0.1	0.0.0.0	0.0.1.0	1.1.1.1
A1	1.0.0.1	0.0.1.1	0.0.0.0	1.0.0.1

Example

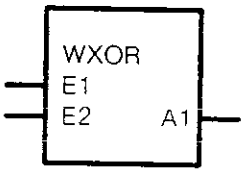
FBD/LD



IL

```
!BA     0
WXOR
        MW 00,00
        KW 00,01
        MW 08,02
```

CE FBD Definition

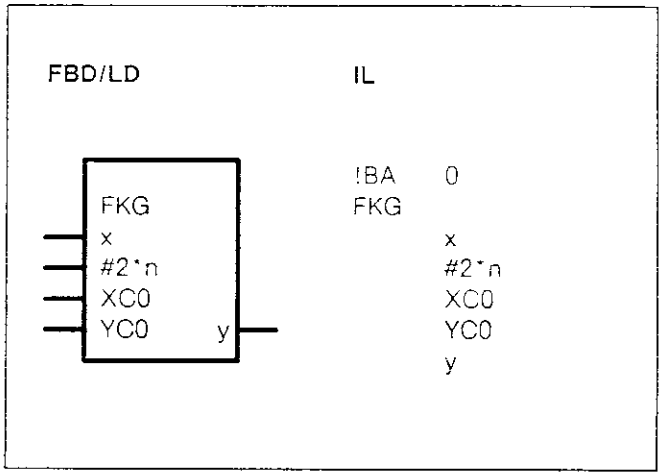


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
E2	E	W	N	P	N	0	0
A1	A	W	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WXOR			
00002		PP 0	E1	Input WORD
00003		PP 0	E2	Input WORD
00004		PP 0	A1	Output WORD

In an x/y coordinate system, a polygon is defined by n coordinate points X0/Y0...Xn-1/Yn-1. For each value at the input x, the function block outputs the affiliated y value of the polygon through the output y.



Parameters

x	WORD	EW, AW, MW, KW	Input for the x value of the polygon
#2*n	DIRECT CONSTANT	#, #H	n: number of interpolation points
XC0	WORD	EW, AW, MW, KW	Input for x values of the interpolation points; Input can be duplicated
YC0	WORD	EW, AW, MW, KW	Input for y values of the interpolation points; Input can be duplicated
y	WORD	AW, MW	Output for the y value of the polygon

CE Data

Runtime:	
Basic runtime:	136 µs
Additional runtime:	9 µs per interpolation point
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

In an x/y coordinate system, a polygon is defined by n coordinate points X0/Y0...Xn-1/Yn-1. For each value at the input x, the function block outputs the affiliated y value of the polygon through the output y.

The following applies to the x coordinates:

$X0 < X1 < X2 \ldots < X_{n-1}$
$2 \leq n \leq 32767$

This function block interpolates linearly between the interpolation points. The resulting polygon represents the relationship between the input quantity x and the output quantity y.

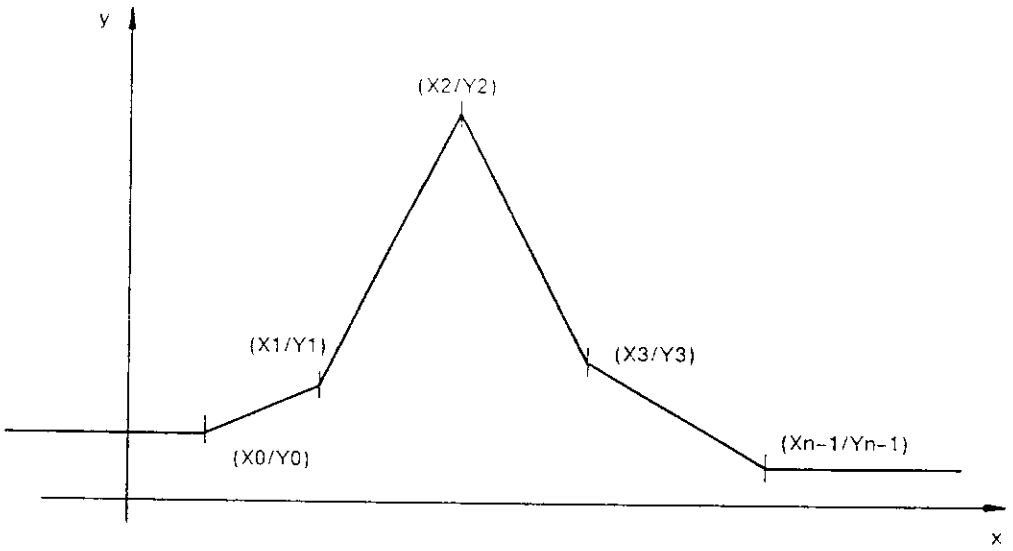
The following applies to interpolation between 2 interpolation points:

$$y = \frac{(x - X_{i-1}) \cdot (Y_i - Y_{i-1})}{X_i - X_{i-1}} + Y_{i-1}$$

Note: The result of division is always rounded down, i.e. any division remainder is ignored.

The following applies to the range outside of the interpolation points:

$y = Y0$ for $x \leq X0$
$y = Y_{n-1}$ for $x \geq X_{n-1}$



x WORD
The current x coordinate is specified at the input x. The block then defines the y coordinate affiliated by the polygon.

#2*n DIRECT CONSTANT
The number of x/y coordinates needed to define the polygon is specified at the input #2*n. This is specified as a direct constant.

n = number of interpolation points
2*n = number of x/y coordinates for n interpolation points

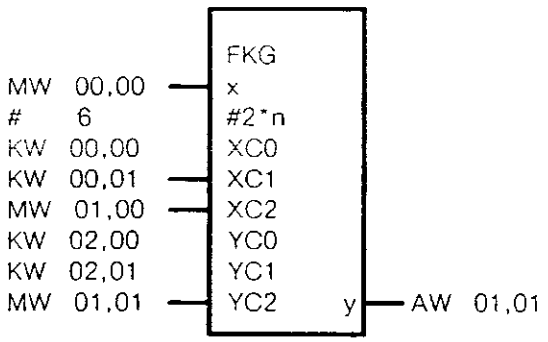
XC0...XCn-1 WORD
The x coordinates of the n interpolation points are specified are the inputs XC0...XCn-1. The input XC0 is capable of duplication. If this input is duplicated, the input YC0 is automatically also duplicated.

YC0...YCn-1 WORD
The y coordinates of the n interpolation points are specified at the inputs YC0...YCn-1. The input YC0 is capable of duplication. If this input is duplicated, the input XC0 is automatically also duplicated.

y WORD
The y coordinate affiliated by the polygon of the specified x coordinate is output through the output y.

Example

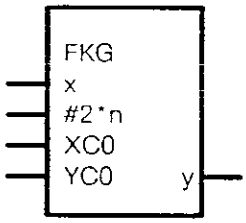
FBD/LD



IL

```
!BA    0
FKG
       MW  00,00
       #    6
       KW  00,00
       KW  00,01
       MW  01,00
       KW  02,00
       KW  02,01
       MW  01,01
       AW  01,01
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
x	E	W	N	P	Y	0	0
#2*n	K	W	N	P	Y	0	0
XC0	E	W	N	P	Y	1	0
YC0	E	W	N	P	Y	1	0
y	A	W	N	P	Y	0	0

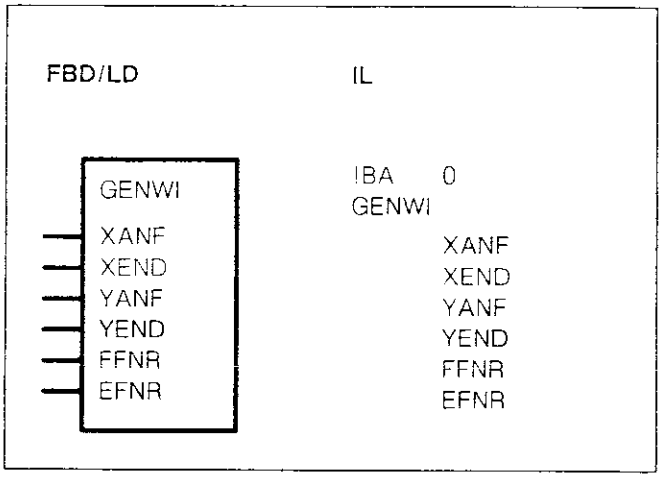
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	FKG			
00002		PP 0	x	Input WORD
00003		PP 0	#2*n	Input # CONSTANT (n = number of interpolation points)
00004	[	1		
00005		PP 1	XC0	Input WORD (x coordinate)
00006		PP 1	YC0	Input WORT (y coordinate)
00007	]	1		
00008		PP 0	y	Output WORD

GENERATION OF A WINDOW (OMS-F) GENWI

This function block generates a single window or a whole window sequence in terms of shape, size and position during the program run on the PLC.

This block is needed when using the video sensor OMS-F.



Parameters

XANF	WORD	EW, AW, MW, KW	Pixel address (column number) of the LEFT side of the window frame
XEND	WORD	EW, AW, MW, KW	Pixel address (column number) of the RIGHT side of the window frame
YANF	WORD	EW, AW, MW, KW	Pixel address (line number) of the TOP side of the window frame
YEND	WORD	EW, AW, MW, KW	Pixel address (line number) of the BOTTOM side of the window frame
FFNR	WORD	EW, AW, MW, KW	Window sequence number
EFNR	WORD	EW, AW, MW, KW	Single window number

CE Data

Runtime:	
Basic runtime:	630 ... 910 µs
Additional runtime:	no
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block generates a single window or a whole window sequence in terms of shape, size and position during the program run on the PLC.

If a whole window sequence is generated, all windows are identical in terms of their shape, size and position. As in the case of the "teach-in method", the windows are stored in the PLC's sensor constant memory, whereby previous entries are overwritten.

Using the PROWI or MODWI block, these windows defined on the PLC can then be loaded to the 35 IV 90 module (called).

The advantage in comparison with "teach-in" is that up-to-date data determined during the program run can be used to define the measurement windows.

XANF WORD

Pixel address (column number) of the LEFT side of the window frame.

The column number for the LEFT side of the window is specified with the operand at this input.
Value range:

35 KI 90, 10-MHz-clock	35 KI 90, 8-MHz-clock
43 ... 558 = 515 columns	31 ... 446 = 415 columns

XEND WORD
Pixel address (column number) of the RIGHT side of the window frame.

The column number for the RIGHT side of the window is specified with the operand at this input.
Value range:

35 KI 90, 10 MHz clock	35 KI 90, 8 MHz clock
43...558 = 515 columns	31...446 = 415 columns

YANF WORD
Pixel address (line number) of the TOP side of the window frame.

The line number for the TOP side of the window is specified with the operand at this input.
Value range:

35 KI 90, 10 MHz clock	35 KI 90, 8 MHz clock
16...300 = 284 lines	16 ...300 = 284 lines

YEND WORD
Pixel address (line number) of the BOTTOM side of the window frame.

The line number for the BOTTOM side of the window is specified with the operand at this input.
Value range:

35 KI 90, 10 MHz clock	35 KI 90, 8 MHz clock
16...300 = 284 lines	16...300 = 284 lines

FFNR WORD
Window sequence number.

The number of the required window sequence is specified with the operand at this input.
Value range: 1...512.
A window sequence consists of maximum of 8 single windows.

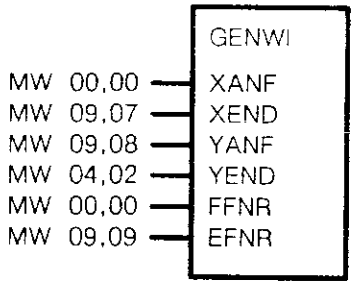
EFNR WORD
Single window number.

The number of the required single window is specified with the operand at this input.
Value range: 0...8

1...8:	Number of the window selected
0:	All windows of the window sequence are selected.

Example

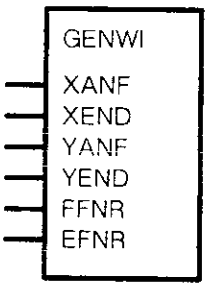
FBD/LD



IL

```
!BA    0
GENWI
       MW 00,00
       MW 09,07
       MW 09,08
       MW 04,02
       MW 00,00
       MW 09,09
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
XANF	E	W	N	P	Y	0	0
XEND	E	W	N	P	Y	0	0
YANF	E	W	N	P	Y	0	0
YEND	E	W	N	P	Y	0	0
FFNR	E	W	N	P	Y	0	0
EFNR	E	W	N	P	Y	0	0

CE IL Definition

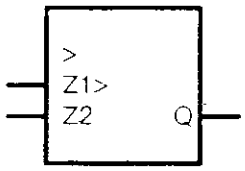
00000	!BA	0	Nr	Block No. (preset to 0)
00001	GENWI			
00002		PP 0	XANF	Input WORD (x start)
00003		PP 0	XEND	Input WORD (x end)
00004		PP 0	YANF	Input WORD (y start)
00005		PP 0	YEND	Input WORD (y end)
00006		PP 0	FFNR	Input WORD (window sequence number)
00007		PP 0	EFNR	Input WORD (single window number)

GREATER THAN >

The value of the operand at the input Z1> is compared to the value of the operand at the input Z2.

If the value at Z1> is greater than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1> is equal to or less than Z2.

FBD/LD



IL

```
!   Z1>
>   Z2
=   Q
```

Parameters

Z1>	WORD	EW, MW, AW, KW	Value to be compared
Z2	WORD	EW, MW, AW, KW	Comparison value
Q	BINARY	A, M, S	Result of the comparison

CE Data

Runtime:	
Basic runtime:	< 12 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The value of the operand at the input Z1> is compared to the value of the operand at the input Z2.

If the value at Z1> is greater than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1> is equal to or less than Z2.

The inputs can neither be negated nor duplicated. The output can be inverted, but cannot be duplicated.

Number range

Integer word (16 Bit)

The following specially applies here to the non-negated inputs:

• low limit	8000 H	-32768
• high limit	7FFF H	+32767

The following generally applies:

• low limit:	8001 H	-32767
• high limit:	7FFF H	+32767
• inadmissible value:	8000 H	---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

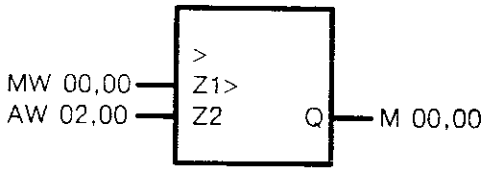
- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

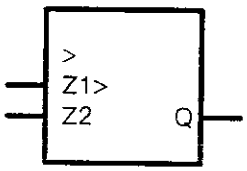
FBD/LD



IL

```
!     MW 00,00
>     AW 02,00
=     M  00,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1>	E	W	Y	P	Y	0	0
Z2	E	W	Y	P	Y	0	0
Q	A	L	Y	P	Y	0	0

CE IL Definition

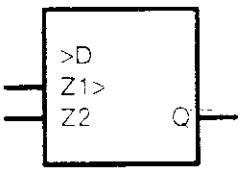
00000	!	PP 0	Z1>	Input WORD
00002	>	PP 0	Z2	Input WORD
00004	=	PP 0	Q	Output BINARY

GREATER THAN, DOUBLE WORD >D

The value of the operand at the input Z1> is compared to the value of the operand at the input Z2.

The state 1 is allocated to the operand at the output Q if the value at Z1> is greater than the one at Z2. The state 0 is allocated to Q if Z1> is equal to or less than Z2.

FBD/LD



IL

```
!BA   0
VGRD
      Z1>
      Z2
      Q
```

Parameters

Z1>	DOUBLE WORD	MD, KD	Value to be compared
Z2	DOUBLE WORD	MD, KD	Comparison value
Q	BINARY	A, M	Result of the comparison

CE Data

Runtime:	
Basic runtime:	35 - 37 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the operand at the input Z1> is compared to the value of the operand at the input Z2.

The state 1 is allocated to the operand at the output Q if the value at Z1> is greater than the one at Z2. The state 0 is allocated to Q if Z1> is equal to or less than Z2.

The inputs can neither be duplicated nor negated. The output can neither be duplicated nor inverted.

Number range

Integer double word (32 Bit)

The following specially applies here to the inputs Z1> and Z2:

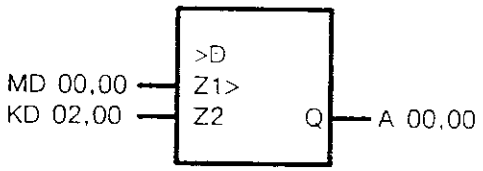
- low limit: 8000 0000 H -2 147 483 648
- high limit: 7FFF FFFF H +2 147 483 647

The following generally applies:

- low limit: 8000 0001 H -2 147 483 647
- high limit: 7FFF FFFF H +2 147 483 647
- inadmissible value: 8000 0000 H ---

Example

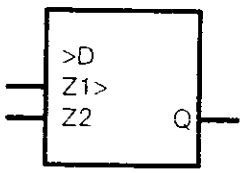
FBD/LD



IL

```
!BA    0
VGRD
       MD 00,00
       KD 02,00
       A  00,00
```

CE FBD Definition



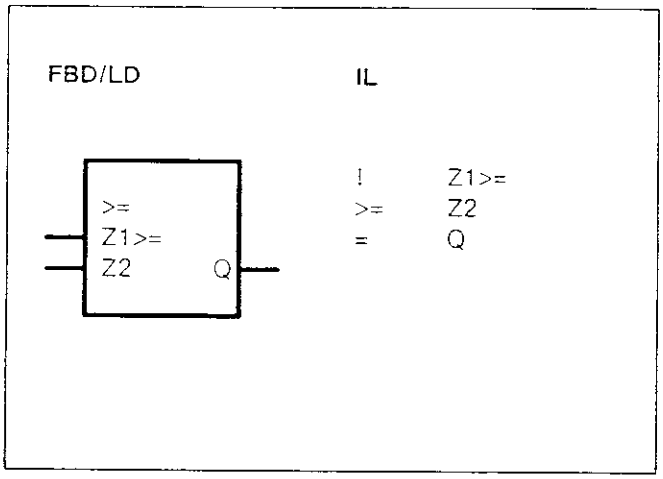
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1>	E	D	N	P	Y	0	0
Z2	E	D	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (presest to 0)
00001	VGRD			
00002		PP 0	Z1>	Input DOUBLE WORD
00003		PP 0	Z2	Input DOUBLE WORD
00004		PP 0	Q	Output BINARY

GREATER THAN OR EQUAL TO >=

The value of the operand at the input Z1>= is compared to the value of the operand at the input Z2. The state 1 is allocated to the operand at the output Q if the value at Z1>= is greater than or equal to the value at Z2. The state 0 is allocated to Q if Z1>= is less than Z2.



Parameters

Z1>=	WORD	EW, MW, AW, KW	Value to be compared
Z2	WORD	EW, MW, AW, KW	Comparison value
Q	BINARY	A, M, S	Result of the comparison

CE Data

Runtime:	
Basic runtime:	< 12 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The value of the operand at the input Z1>= is compared to the value of the operand at the input Z2. The state 1 is allocated to the operand at the output Q if the value at Z1>= is greater than or equal to the value at Z2. The state 0 is allocated to Q if Z1>= is less than Z2.

The inputs can be negated, but not duplicated. The output can be inverted, but not duplicated.

Number range

Integer word (16 Bit)

The following specially applies here to the non-negated inputs:

- low limit: 8000 H -32768
- high limit: 7FFF H +32767

The following generally applies:

- low limit: 8001 H -32767
- high limit: 7FFF H +32767
- inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

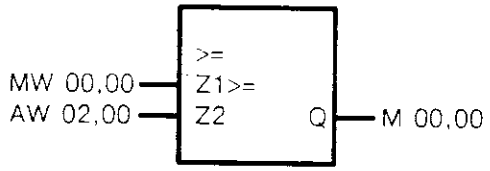
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

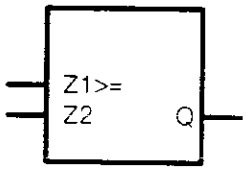
FBD/LD



IL

```
!     MW 00,00
>=    AW 02,00
=     M  00,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1>=	E	W	Y	P	Y	0	0
Z2	E	W	Y	P	Y	0	0
Q	A	L	Y	P	Y	0	0

CE IL Definition

00000	!	PP 0	Z1>=	Input WORD
00002	>=	PP 0	Z2	Input WORD
00004	=	PP 0	Q	Output BINARY

IF THEN

IF THEN is a basic function in the manufacturer library.
The element's name in the library is :=
The element cannot be called in the CE and CE IL editors.

FBD/LD	IL
Call not possible	

Parameters

CE Data

Runtime:

Basic runtime:	4 µs
Additional runtime:	---
Output updating:	---
Number of historical values:	---
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801/ 35 ZE 93 R101

Description

IF THEN is a basic function in the manufacturer library.
The element's name in the library is :=
The element cannot be called in the CE and CE IL editors.
The programming unit uses this element to realize connection lines.

Example

FBD/LD	IL
Call is not possible	

CE FBD Definition

not defined

CE IL Definition

```
00000    !      PP  0    inp
00002    =      PP  0    outp
```

IF THEN, WORD

Call not possible

IF THEN, WORD is a basic function in the manufacturer library. The element's name in the library is :=WO
This element can not be called in the CE and CE IL editors.

FBD/LD	IL
Call not possible	

Parameters

CE Data

Runtime:	
Basic runtime:	4 µs
Additional runtime:	---
Output updating:	---
Number of historical values:	---
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

IF THEN, WORD is a basic function in the manufacturer library. The element's name in the library is :=WO
This element cannot be called in the CE and CE IL editors.
The programming unit uses this element to realize connecting lines.

Example

FBD/LD	IL
Call not possible	

CE FBD Definition

not defined

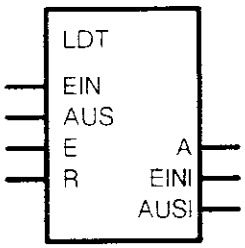
CE IL Definition

```
00000    !      PP  0    inp
00002    =      PP  0    outp
```

ILLUMINATION PUSHBUTTON CONTROL — LDT

This function block changes the state at the output A with every 0/1 edge at the input E (like a toggle flip-flop).

FBD/LD



IL

```
!BA   0
LDT
      EIN
      AUS
      E
      R
      A
      EINI
      AUSI
```

Parameters

EIN	BINARY	E, A, M, K, S	Dynamic input for activating output A
AUS	BINARY	E, A, M, K, S	Dynamic input for deactivating output A
E	BINARY	E, A, M, K, S	Dynamic input for activating and deactivating output A
R	BINARY	E, A, M, K, S	Static input for deactivating output A
A	BINARY	A, M	Output
EINI	BINARY	A, M	Pulse for the duration of one PLC program cycle when activating output A
AUSI	BINARY	A, M	Pulse for the duration of one PLC program cycle when deactivating output A

CE Data

Runtime:	
Basic runtime:	79...83 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block changes the state at the output A with every 0/1 edge at the input E (like a toggle flip-flop). However, this only applies if the input R is inactive. The output A can be influenced additionally with the inputs EIN, AUS and R.

All inputs except R are edge-controlled, i. e. they react only to a 0/1 edge. The input R is active for as long as a static 1 signal is present at it. Brief pulses are available at the output EINI and AUSI when a status change of the output A occurs.

EIN	AUS	E	R	A
X	X	X	H	L
X	X	>	L	NA
X	>	<	L	L
>	<	<	L	H

NA: Inverting of the previous state of A
X: State is ignored
>: 0/1 edge
<: no 0/1 edge
L: State 0
H: State 1

EIN BINARY

The output A is set with every 0/1 edge at the input EIN. However, this only applies when the inputs AUS, E and R are inactive. Therefore, of all inputs the input EIN has the lowest priority.

AUS BINARY

The output A is reset with every 0/1 edge at the input AUS. Therefore, the input AUS has a higher priority than the input EIN, but a lower priority than the inputs E and R.

E BINARY

With every 0/1 edge at the input E, the block changes the status at the output A. However, this only applies if the input R is inactive. Therefore, the input E has a higher priority than the inputs EIN and AUS, but a lower priority than the input R.

R BINARY

The output A is reset when a static 1 signal is present at the input R.

The input R has a higher priority than the inputs E and EIN. A priority conflict with the input AUS is not possible because both inputs have the same effect on the outputs.

A BINARY

The output A is changed corresponding to the definition of the inputs.

EINI BINARY

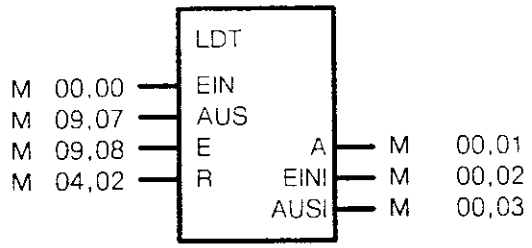
A pulse is generated at the output EINI with every 0/1 edge of the output A. The length of this pulse is 1 program cycle.

AUSI BINARY:

A pulse is generated at the output AUSI with every 1/0 edge of the output A. The length of this pulse is 1 program cycle.

Example

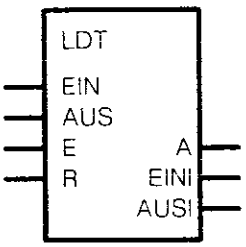
FBD/LD



IL

```
!BA   0
LDT
      M 00,00
      M 09,07
      M 09,08
      M 04,02
      M 00,01
      M 00,02
      M 00,03
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
EIN	E	L	N	P	Y	0	0
AUS	E	L	N	P	Y	0	0
E	E	L	N	P	Y	0	0
R	E	L	N	P	Y	0	0
A	A	L	N	P	Y	0	0
EINI	A	L	N	P	Y	0	0
AUSI	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	LDT			
00002		PP 0	EIN	0/1: Activating A (BINARY)
00003		PP 0	AUS	0/1: Deactivating A (BINARY)
00004		PP 0	E	0/1: Activating/deactivating A (BINARY)
00005		PP 0	R	Statically: deactivating A (BINARY)
00006		PP 0	A	Output (BINARY)
00007		PP 0	EINI	Activating pulse (BINARY)
00008		PP 0	AUSI	Deactivating pulse (BINARY)

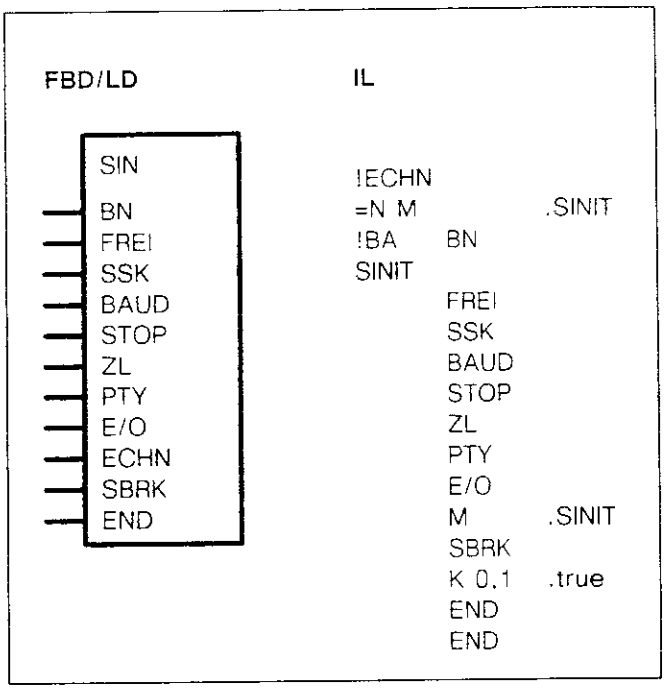
INITIALIZATION AND CONFIGURATION OF THE SERIAL INTERFACES

SIN

The connection element SIN is processed *once* with every 0->1 edge at the input FREI. It initializes the serial interface on the 35 DS 91 module specified at the input SSK.

This connection element is identical with the one in the ABB Procontic b and is based on the SINIT block of the ABB Procontic T300. The ECHN input was negated to ensure compatability with the CE of the ABB Procontic b. The user must absolutely designate the .SINIT flag in the variable or symbol list before transferring his program to the control system.

The SINIT block has more extensive functions than those of the connection element SIN.



Parameters

BN	BINARY	A, E, S, M, K	Block number
FREI	BINARY	A, E, S, M, K	Enable block, 0 -> 1 edge
SSK	WORD	AW, EW, MW, KW	Interface identifier (1, 2 or 3)
BAUD	WORD	AW, EW, MW, KW	Baud rate; input has no effect
STOP	WORD	AW, EW, MW, KW	Number of stop bits (1 or 2)
ZL	WORD	AW, EW, MW, KW	Character length 7 or 8 bits per character
PTY	BINARY	A, E, S, M, K	Parity, enable/disable
E/O	BINARY	A, E, S, M, K	Parity even/odd
ECHN	BINARY	A, E, S, M, K	no echo/echo
SBRK	BINARY	A, E, S, M, K	Send break character
END	WORD	AW, EW, MW, KW	End of text character

CE Data

Runtime:	
Basic runtime:	25 µs
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V5 / 935 PC 83 R201 / 35 ZE 93 R101

Description

The connection element SIN is processed *once* with every 0 -> 1 edge at the input FREI. It initializes the serial interface on the 35 DS 91 module specified at the input SSK.

This connection element is identical with the one in the ABB Procontic b and is based on the SINIT block of the ABB Procontic T300. The ECHN input was negated to ensure compatability with the CE of the ABB Procontic b. The user must absolutely designate the .SINIT flag in the variable or symbol list before transferring his program to the control system.

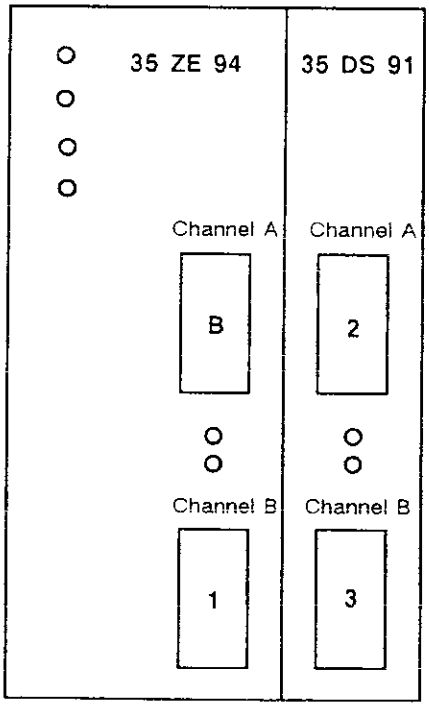
The SINIT block has more extensive functions than those of the connection element SIN.

The inputs are neither capable of duplication nor can they be negated/inverted.

An additional interface module 35 DS 91 can be connected to the PLC. In this case, the serial interfaces 1...3 are freely available to the user. These three interfaces can be operated by the PLC program (e.g. with the DRUCK and EMAS blocks). Please also refer to the chapter entitled "serial interfaces" in the PLC description. *Before* use of one of these three interfaces, it must be initialized. The connection element SIN is available for this purpose.

Interfaces of the PLC

Configuration



The PLC can operate the following serial interfaces:

- B: Operator interface B
- 1: Interface 1
- 2: Interface 2 (additional 35 DS 91 module)
- 3: Interface 3 (additional 35 DS 91 module)

Use of the interfaces

- User interface B:
Connection of a PC or terminal for programming and testing the PLC.

- Interface 1:
Either connection of a device that can be addressed by an operator control command (e.g. programmer or printer) or suitable for use for the DRUCK and EMAS blocks.

- Interface 2:
Suitable for use for the DRUCK and EMAS blocks.

- Interface 3:
Suitable for use for the DRUCK and EMAS blocks.

Synchronization between the serial interfaces 1...3 and the connected module is always achieved with the signals RTS and CTS.

BN SPECIAL INPUT
This input has been provided for reasons of compatibility with the ABB Procontic b.
In the case of the ABB Procontic T300, a block number between 0 and 999 is specified at the input BN. After the FBD/LD compiler run, this block number then appears in the IL as the block number of the SINIT block. Input of the block number at the input BN is done as follows:
B No.
where:
B: is a key word
No.: is a numeric value from 0 ... 999
The programming system reacts to this input with the following message:
Error during type test (left: S right: L)!
<ESC> nevertheless / key
Now press <ESC> for the input to be accepted.

FREI BINARY
The block is run through *once* when a 0->1 edge is specified at the FREI input. As the result of this, the serial interface whose number is specified at the SSK input is initialized and the interface is then operable.

SSK WORD
The number of the interface to be initialized is specified at the input SSK.
The following applies: $1 \leq \text{number} \leq 3$

BAUD WORD
The value for the baud rate specified at the BAUD input has no significance. In the case of ABB Procontic T300, the baud rate is set on the hardware of the 35 DS 91 module.

STOP WORD
The number of required stop bits is specified at the input STOP. One or two stop bits are possible.

ZL WORD
The required character length is specified at the input ZL. The character length signifies the number of data bits per character.
7 or 8 data bits per character are possible.

PTY BINARY
Whether a character is transferred with or without a parity bit is specified at the input PTY.
PTY = 0 -> Transfer without parity bit
PTY = 1 -> Transfer with parity bit

E/O BINARY
Whether an even or odd parity bit is required is specified at the input E/O.
E/O = 0 -> Odd parity bit
E/O = 1 -> Even parity bit

ECHN BINARY
Whether the characters received through the applicable interface are to be reflected (echoed) by the PLC is specified at the input ECHN. In this way, the sender of a character, for example, can determine whether or not it has arrived correctly in the PLC.
ECHN = 0 -> Echo, character is reflected
ECHN = 1 -> No echo, character is not reflected

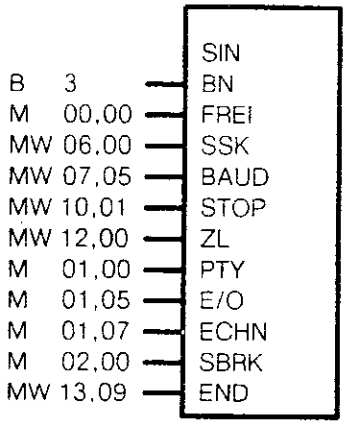
SBRK BINARY
The state of the transmit line TxD can be influenced at the input SBRK (send break character).
SBRK = 0 -> Normal state of the transmit line TxD for transfer of characters
SBRK = 1 -> Transmit line TxD set to "0"

END WORD
A freely selectable end of text character for the transmitting and receiving directions can be specified at the END input. This end character is then appended automatically to every text (telegram) which the DRUCK block sends to the outside world through the serial interface.
When a telegram is received through the serial interface, the PLC recognizes the end of the telegram by this end character.
The end character is specified as a numeric value.

Example :
3 or 03H signifies <ETX>
4 or 04H signifies <EOT>
13 or 0DH signifies <CR>
10 or 0AH signifies <LF>
32 or 20H signifies <SP>

Example

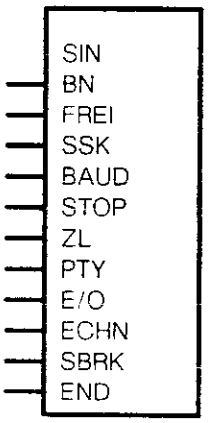
FBD/LD



IL

```
!      M 01,07
=N     M 20,00        .SINIT
!BA       3
SINIT
          M   00,00
          MW  06,00
          MW  07,05
          MW  10,01
          MW  12,00
          M   01,00
          M   01,05
          M   20,00   .SINIT
          M   02,00
          K   00,01   .true
          MW  13,09
          MW  13,09
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
BN	S	L	N	P	Y	0	0
FREI	E	L	N	P	Y	0	0
SSK	E	W	N	P	Y	0	0
BAUD	E	W	N	P	Y	0	0
STOP	E	W	N	P	Y	0	0
ZL	E	W	N	P	Y	0	0
PTY	E	L	N	P	Y	0	0
E/O	E	L	N	P	Y	0	0
ECHN	E	L	N	P	Y	0	0
SBRK	E	L	N	P	Y	0	0
END	E	W	N	P	Y	0	0

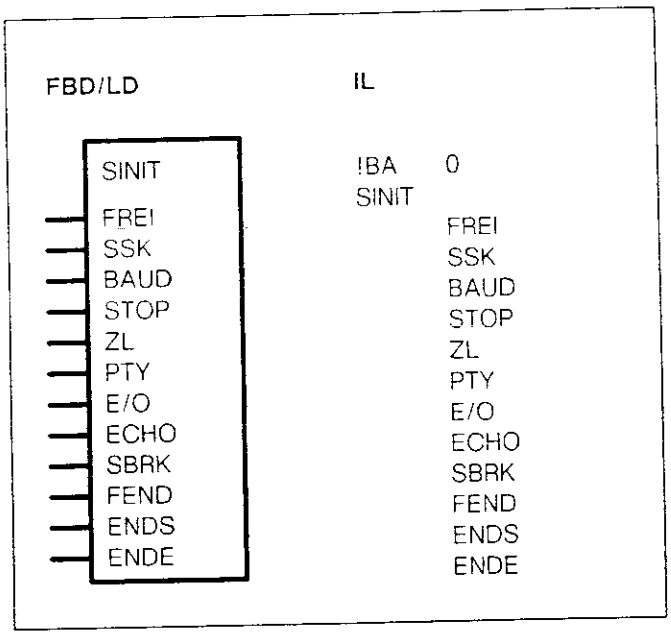
CE IL Definition

```
00000   !       PP  0     ECHN    Input BINARY (no echo)
00002   =N      M         .SINIT  ECHO (auto. gerated in SINIT)
00004   !BA     PP  0     BN      Block No.
00005   SINIT
00006           PP  0     FREI    Input BINARY (block enable)
00007           PP  0     SSK     Input WORD (interfaces)
00008           PP  0     BAUD    Input WORD (baud rate)
00009           PP  0     STOP    Input WORD (number of stop bits)
00010           PP  0     ZL      Input WORD (character length)
00011           PP  0     PTY     Input BINARY (parity)
00012           PP  0     E/O     Input BINARY (parity even/odd)
00013           M         .SINIT  ECHO (auto. generated in SINIT)
00014           PP  0     SBRK    Input BINARY (send break)
00015           K   0,1   .true   1
00016           PP  0     END     Input WORD (text end character)
00017           PP  0     END     Input WORD (text end character)
```

INITIALIZATION AND CONFIGURATION OF THE SERIAL INTERFACES

SINIT

The function block SINIT is processed *once* with every 0 -> 1 edge at the FREI input. It initializes the serial interface on the 35 DS 91 module specified at the SSK input.



Parameters

FREI	BINARY	A, E, S, M, K	Enable processing of the block, 0 -> 1 edge
SSK	WORD	AW, EW, MW, KW	Interface identifier (1, 2 or 3)
BAUD	WORD	AW, EW, MW, KW	Baud rate; input has no effect
STOP	WORD	AW, EW, MW, KW	Number of stop bits (1 or 2)
ZL	WORD	AW, EW, MW, KW	Character length, 7 or 8 data bits per character
PTY	BINARY	A, E, S, M, K	Parity, enable/disable
E/O	BINARY	A, E, S, M, K	Parity even/odd
ECHO	BINARY	A, E, S, M, K	Echo, on/off
SBRK	BINARY	A, E, S, M, K	Send break character
FEND	BINARY	A, E, S, M, K	Enable end of text character for transmitting direction
ENDS	WORD	AW, EW, MW, KW	End of text character for transmitting direction
ENDE	WORD	AW, EW, MW, KW	End of text character for receiving direction

CE Data

Runtime:	
Basic runtime:	25 µs
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V5 / 935 PC 83 R201 / 35 ZE 93 R101

Description

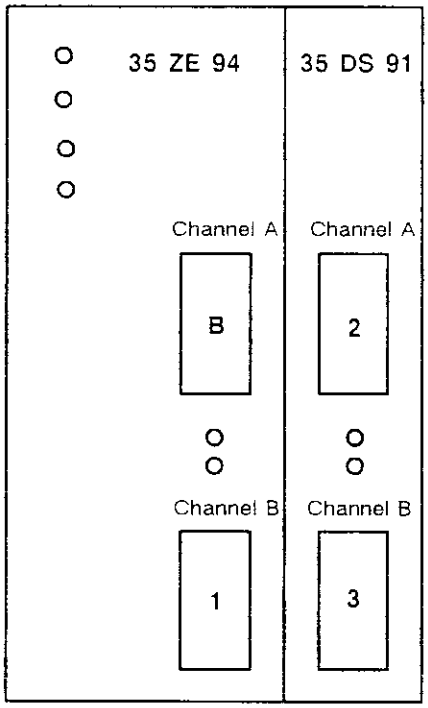
The function block SINIT is processed *once* with every 0 -> 1 edge at the FREI input. It initializes the serial interface on the 35 DS 91 module specified at the SSK input.

The inputs can neither be duplicated nor negated/inverted.

An additional interface module 35 DS 91 can be connected to the PLC. In this case, the serial interfaces 1...3 are freely available to the user. These three interfaces can be operated by the PLC program (e.g. with the DRUCK and EMAS blocks). Please also refer to the chapter entitled "serial interfaces" in the PLC description. *Before* use of one of these three interfaces, it must be initialized. The function block SINIT is available for this purpose.

Interfaces of the PLC

Configuration



The PLC can operate the following interfaces:

- B: Operator interface B
- 1: Interface 1
- 2: Interface 2 (additional 35 DS 91 module)
- 3: Interface 3 (additional 35 DS 91 module)

Use of the interfaces

- User interface B:
Connection of a PC or terminal for programming and testing the PLC.

- Interface 1:
Either connection of a device that can be addressed by an operator control command (e.g. programmer or printer) or suitable for use for the DRUCK and EMAS blocks.

- Interface 2:
Suitable for use for the DRUCK and EMAS blocks.

- Interface 3:
Suitable for use for the DRUCK and EMAS blocks.

Synchronization between the serial interfaces 1...3 and the connected module is always achieved with the signals RTS and CTS.

FREI BINARY
The block is run through *once* when a 0->1 edge is specified at the FREI input. As the result of this, the serial interface whose number is specified at the SSK input is initialized and the interface is then operable.

SSK WORD
The number of the interface to be initialized is specified at the input SSK.
The following applies: 1 ≤ number ≤ 3

BAUD WORD
The value for the baud rate specified at the BAUD input has no significance. In the case of ABB Procontic T300, the baud rate is set on the hardware of the 35 DS 91 module.

STOP WORD
The number of required stop bits is specified at the input STOP. One or two stop bits are possible.

ZL WORD
The required character length is specified at the input ZL. The character length signifies the number of data bits per character.
7 or 8 data bits per character are possible.

PTY BINARY
Whether a character is transferred with or without a parity bit is specified at the input PTY.
PTY = 0 -> Transfer without parity bit
PTY = 1 -> Transfer with parity bit

E/O BINARY
Whether an even or odd parity bit is required is specified at the input E/O.
E/O = 0 -> Odd parity bit
E/O = 1 -> Even parity bit

ECHO BINARY
Whether the characters received through the applicable interface are to be reflected (echoed) by the PLC is specified at the input ECHO. In this way, the sender of a character, for example, can determine whether or not it has arrived correctly in the PLC.
ECHO = 0 -> No echo, character is not reflected
ECHO = 1 -> Echo, character is reflected

SBRK BINARY

The state of the transmit line TxD can be influenced at the input SBRK (send break character).

SBRK = 0 -> Normal state of the transmit line TxD for transfer of characters

SBRK = 1 -> Transmit line TxD set to "0"

FEND BINARY

Whether or not the end of text character planned at input ENDS is output at the same time is specified at the input FEND (enable end character)

FEND = 0 -> End of text character in transmitting direction is not output

FEND = 1 -> End of text character in transmitting direction is output

ENDS WORD

A freely selectable end of text character for the transmitting direction can be specified at the ENDS input. This end character is then appended automatically to every text (telegram) which the DRUCK block sends to the outside world through the serial interface. However, a precondition is that input FEND is enabled.

The end of text character is specified as a numeric value.

Example:

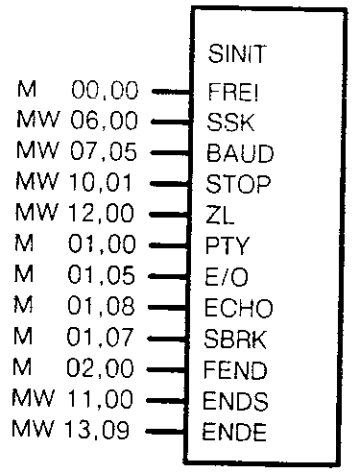
3 or 03H signifies <ETX>
4 or 04H signifies <EOT>
13 or 0DH signifies <CR>
10 or OAH signifies <LF>
32 or 20H signifies <SP>
. . . . .
. . . . .
. . . . .

ENDE WORD

A freely selectable end of text character for the receiving direction can be specified at the ENDE input. When a telegram is received through the serial interface, the PLC recognizes the end of the telegram by virtue of this end character. The end character is specified in the same way as in the case of the ENDS input.

Example

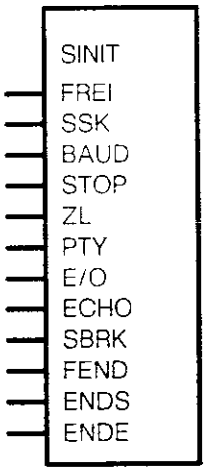
FBD/LD



IL

```
!BA    0
SINIT
       M    00,00
       MW   06,00
       MW   07,05
       MW   10,01
       MW   12,00
       M    01,00
       M    01,05
       M    01,08
       M    01,07
       M    02,00
       MW   11,00
       MW   13,09
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
SSK	E	W	N	P	Y	0	0
BAUD	E	W	N	P	Y	0	0
STOP	E	W	N	P	Y	0	0
ZL	E	W	N	P	Y	0	0
PTY	E	L	N	P	Y	0	0
E/O	E	L	N	P	Y	0	0
ECHO	E	L	N	P	Y	0	0
SBRK	E	L	N	P	Y	0	0
FEND	E	L	N	P	Y	0	0
ENDS	E	W	N	P	Y	0	0
ENDE	E	W	N	P	Y	0	0

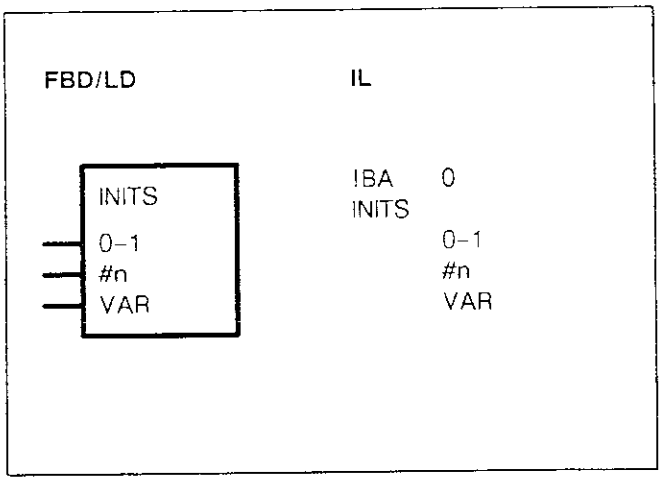
CE IL Definition

00000	!BA	0	Nr	Block No. (vorbelegt 0)
00001	SINIT			
00002		PP 0	FREI	Input BINARY (block enable)
00003		PP 0	SSK	Input WORD (interfaces)
00004		PP 0	BAUD	Input WORD (baud rate)
00005		PP 0	STOP	Input WORD (number of stop bits)
0006		PP 0	ZL	Input WORD (character length)
0007		PP 0	PTY	Input BINARY (parity)
0008		PP 0	E/O	Input BINARY (parity even/odd)
0009		PP 0	ECHO	Input BINARY (echo)
00010		PP 0	SBRK	Input BINARY (send break)
00011		PP 0	FEND	Input BINARY (output ENDS)
00012		PP 0	ENDS	Input WORD (send end of text character)
00013		PP 0	ENDE	Input WORD (receive end of text character)

INITIALIZE MEMORY AREA IN THE OPERAND MEMORY WITH ZERO

INITS

This function block serves the purpose of word-by-word initialization of memory areas in the operand memory with the value 0.



Parameters

0-1	BINARY	E, A, M, S	Enabling for non-recurring processing of the block (0/1 edge)
#n	DIRECT CONSTANT	#, #H	Quantity (n) of memory words to be initialized
VAR	BINARY, WORD, DOUBLE WORD	E, A, M, K, S, EW, AM, MW, KW, MD, KD	The n memory words specified at the input #n are initialized as from this variable.

CE Data

Runtime:	
Basic runtime:	40 µs
Additional runtime:	2.6 µs per memory word to be initialized
Output updating:	not applicable
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block serves the purpose of word-by-word initialization of memory areas in the operand memory with the value 0.

The block is run through precisely once in the event of a 0/1 edge at the input 0-1. In doing so, *n memory words* are initialized with the value 0 as from the variable specified at the input VAR. The variable specified at this input can be a BINARY, WORD or DOUBLE WORD operand.

The following must be observed:

BINARY operand:
Occupies 1 BYTE in the operand memory.
WORD operand:
Occupies 1 WORD in the operand memory.
DOUBLE WORD operand:
Occupies 2 WORDS in the operand memory.

0-1 BINARY

Block enable.

The block is run through precisely *once* when a 0/1 edge appears at this input.

#n DIRECT CONSTANT

Quantity (n) of words to be initialized.

The number of *memory words* to be initialized is specified as a DIRECT CONSTANT at this input.
Value range: $0 \leq n \leq 65535$

No initialization is realized if the value 0 is specified.

VAR BINARY, WORD, DOUBLE WORD

n memory words are initialized with the value 0 as from this variable (inclusive).

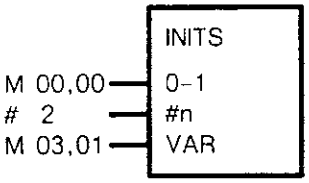
Example

The 4 binary flags

M 03,01
M 03,02
M 03,03
M 03,04

are to be initialized with the value 0. A binary flag in memory takes up 1 byte. Therefore there is a total of 4 bytes = 2 words to be initialized. For this reason, the value 2 must be specified at the input #n as a DIRECT CONSTANT and at input VAR the flag M 03,01.

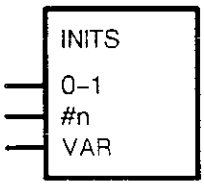
FBD/LD



IL

```
!BA    0
INITS
       M 00,00
       # 2
       M 03,01
```

CE FBD Definition

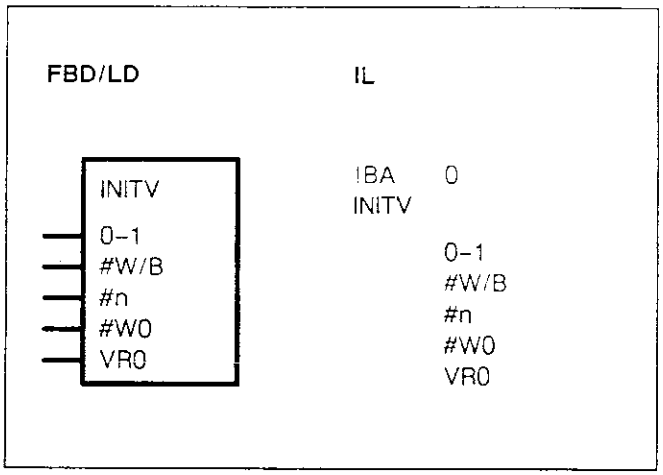


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0-1	E	L	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
VAR	E	X	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	INITS			
00001		PP 0	0-1	Enable edge (BINARY)
00002		PP 0	#n	Number (DIRECT CONSTANT)
00003		PP 0	VAR	Start variable (BINARY, WORD, DOUBLE WORD)

This function block serves to initialize BINARY and WORD variables with numeric values.



Parameters

0-1	BINARY	E, A, M, S	Enabling for non-recurring processing of the block (0/1 edge)
#W/B	DIRECT CONSTANT	#, #H	Format specification #W/B = 1: Initialization of WORD variables #W/B = 0: Initialization of BINARY variables
#n	DIRECT CONSTANT	#, #H	Number of variables to be initialized
#W0	DIRECT CONSTANT	#, #H	Initialization value for the subsequent variable; The inputs #W0 and VR0 can be duplicated in pairs
VR0	BINARY, WORD	E, A, M, EW, AW, MW	Variable to be initialized; The inputs VR0 and #W0 can be duplicated in pairs

CE Data

Runtime:	
Basic runtime:	34 µs
Additional runtime:	11 µs per additional variable VR1 ... VRn-1 to be initialized
Output updating:	not applicable
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block serves to initialize BINARY and WORD variables with numeric values.

The initialization values are specified as direct constants. The block is run through precisely once when a 0/1 edge appears at the input 0-1.

0-1 BINARY

The block is run through precisely *once* when a 0/1 edge appears at the input 0-1.

#W/B DIRECT CONSTANT

The format of the variable to be initialized is specified as a direct constant at the input #W/B. The following applies:

#W/B = 1: Word variable (EW, AW, MW)
#W/B = 0: Binary variable (E, A, M)

#n DIRECT CONSTANT

The quantity n of variables to be initialized is specified as a direct constant at the input #n.

#W0 DIRECT CONSTANT

The initialization value for the subsequent variable is specified as a direct constant at the input #W0.

The inputs #W0 and VR0 can be duplicated in pairs.

The following affiliations apply :

Value 0 -> Variable 0
:
Value n-1 -> Variable n-1

VR0 BINARY/WORD

The first variable to be initialized is specified at the input VR0.

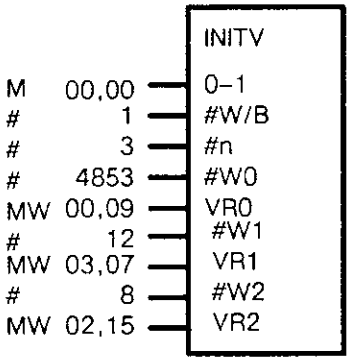
The inputs VR0 and #W0 can be duplicated in pairs.

Example

The following variables are to be initialized:

MW 00,09 = 4853
MW 03,07 = 12
MW 02,15 = 8

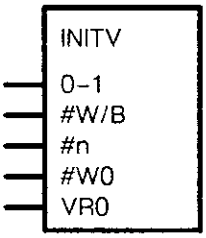
FBD/LD



IL

```
!BA   0
INITV
      M   00,00
      #       1
      #       3
      #    4853
      MW  00,09
      #      12
      MW  03,07
      #       8
      MW  02,15
```

CE FBD Definition

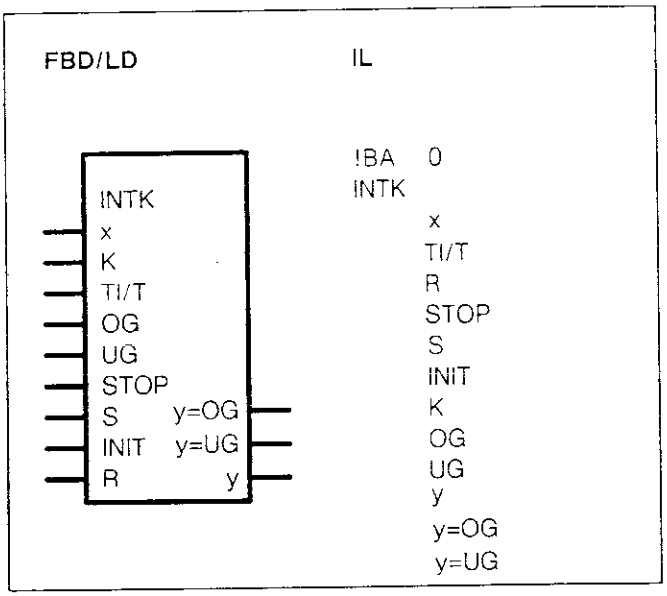


Param.	Param. Goup	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0-1	E	L	N	P	Y	0	0
#W/B	K	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
#W	K	W	N	P	Y	1	1
VR	E	X	N	P	Y	1	1

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	INITV			
00002		PP 0	0-1	Enable edge (BINARY)
00003		PP 0	#W/B	Format für Variable (#)
00004		PP 0	#n	Number (#)
	[	1		
00005		PP 1	#W	Initial value for variable (#)
00006		PP 1	VR	Variable to be initialized (BINARY/WORD)
	]	1		

This block generates the integral of the controlled variable x multiplied by the proportinal coefficient K.



Parameters

x	WORD	EW, AW, MW, KW	Controlled variable
K	WORD	EW, AW, MW, KW	Proportional coefficient, output as a percentage
TI/T	WORD	EW, AW, MW, KW	Integration time scaled to the cycle time
OG	WORD	EW, AW, MW, KW	High limit for the manipulated variable y
UG	WORD	EW, AW, MW, KW	Low limit for the manipulated variable y
STOP	BINARY	E, A, M, S, K	Integrator stop
S	BINARY	E, A, M, S, K	Set output to INIT value
INIT	WORD	EW, AW, MW, KW	Initial value
R	BINARY	E, A, M, S, K	Reset output y to the value 0
y=OG	BINARY	A, M	Output y has reached top limit
y=UG	BINARY	A, M	Output y has reached low limit
y	WORD	AW, MW	Manipulated variable

CE Data

Runtime:	
Basic runtime:	189 - 197 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	3 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701/R801 / 35 ZE 93 R101

Description

This block generates the integral of the controlled variable x multiplied by the proportinal coefficient K.
The integrator's output y can be manipulated as follows:

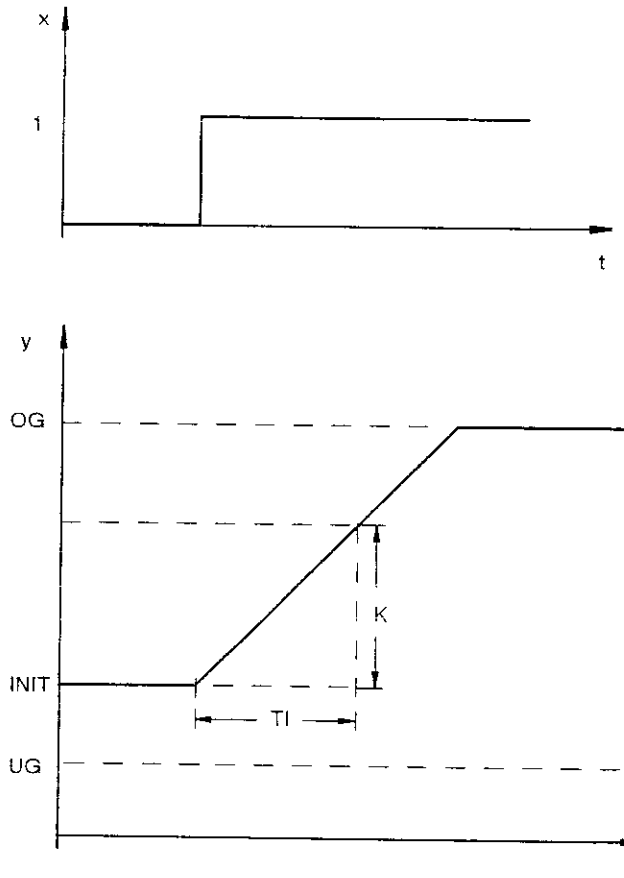
- It can be set to the value 0 by a 1 signal at the input R (reset)
- It can be latched to the current value by a 1 signal at the input STOP
- It can be set to the initial value at the INIT input by a 1 signal at the input S (set)
- It can be limited to a maximum value specified at the input OG (high limit)
- It can be limited to a minimum value specified at the input UG (low limit)

The inputs and outputs can neither be duplicated nor negated/inverted.

Transfer function

$$F(s) = \frac{K}{s * T}$$

Transfer function



x WORD
The operand for the controlled variable is specified at the input x.

K WORD
The proportional coefficient is specified at the input K. It serves to weight the controlled variable at the input x. Weighting is achieved by multiplying the controlled variable by the proportional coefficient.
As from version V6 (GJP 122900 R301), the proportional coefficient is specified as a percentage.

Example:

1	=	1	Percent
55	=	55	Percent
100	=	100	Percent
1000	=	1000	Percent
-100	=	-100	Percent

- 1 percent means that the block multiplies the value at input x by the factor 0.01
- 100 percent means that the block multiplies the value at input x by the factor 1
- 1000 percent means that the block multiplies the value at input x by the factor 10

TI/T WORD
The integration time is specified at the input TI/T. It must be scaled to the cycle time. During the time TI the output y of the integrator changes by the value K * x.

Value range: $0 \leq TI/T \leq 328$

- If values are specified which are beyond the admissible value range the PLC uses the value 328.
- A large integration time (TI) can be achieved by choosing a great cycle time, too. If the block is used within a run number block, the cycle time of the run number block is valid for block INTK and not the cycle time (KD 0,0) of the PLC program.

OG WORD
The manipulated variable y can be limited to a value range. The high limit for the manipulated variable y is specified at the input OG.

UG WORD
The manipulated variable y can be limited to a value range. The low limit for the manipulated variable y is specified at the input UG.

STOP BINARY *)
Integration can be stopped with the STOP input.

STOP = 0 -> integration is not stopped
STOP = 1 -> integration is stopped, i.e. the output y no longer changes.

S BINARY *)
By means of the input S, the manipulated variable y can be set to the initial value specified at the input INIT. Integration then again begins as from the initial value.

S = 0 -> No setting
S = 1 -> Output y is set to the specified initial value

INIT WORD
The initial value to which the output y must be set when required is specified at the input INIT.

R BINARY *)
With the input R, the output y can be reset to the value 0. Integration then again begins as from the value 0.

*) Priority sequence for the inputs STOP, S and R:

R Highest priority
STOP
S Lowest priority

y=OG BINARY
Whether the value at the output y has reached the specified top limit is signalled at the output y=OG. Integration is stopped automatically when the limit is reached.

y=OG = 0 -> y has not reached the limit
y=OG = 1 -> y has reached the limit

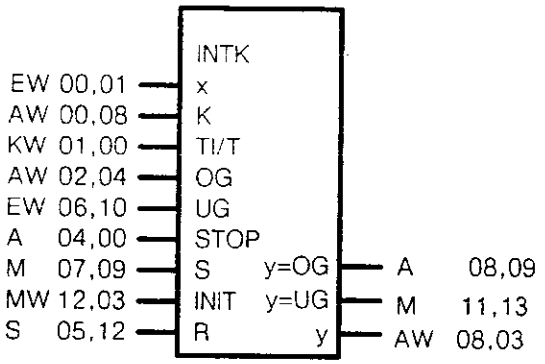
y=UG BINARY
Whether the value at the output y has reached the specified lower limit is signalled at the output y=UG. Integration is stopped automatically when the limit is reached.

y=UG = 0 -> y has not reached the limit
y=UG = 1 -> y has reached the limit

y WORD
The manipulated variable (output value of the integrator) is output through the output y.

Example

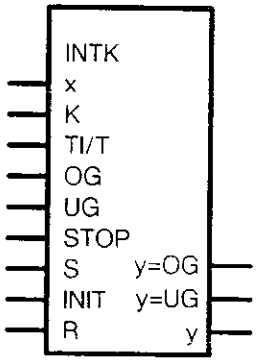
FBD/LD



IL

```
!BA     0
INTK
        EW  00,01
        KW  01,00
        S   05,12
        A   04,00
        M   07,09
        MW  12,03
        AW  00,08
        AW  02,04
        EW  06,10
        AW  08,03
        A   08,09
        M   11,13
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
x	E	W	N	P	Y	0	0
K	E	W	N	P	Y	0	0
TI/T	E	W	N	P	Y	0	0
OG	E	W	N	P	Y	0	0
UG	E	W	N	P	Y	0	0
STOP	E	L	N	P	Y	0	0
S	E	L	N	P	Y	0	0
INIT	E	W	N	P	Y	0	0
R	E	L	N	P	Y	0	0
y=OG	A	L	N	P	Y	0	0
y=UG	A	L	N	P	Y	0	0
y	A	W	N	P	Y	0	0

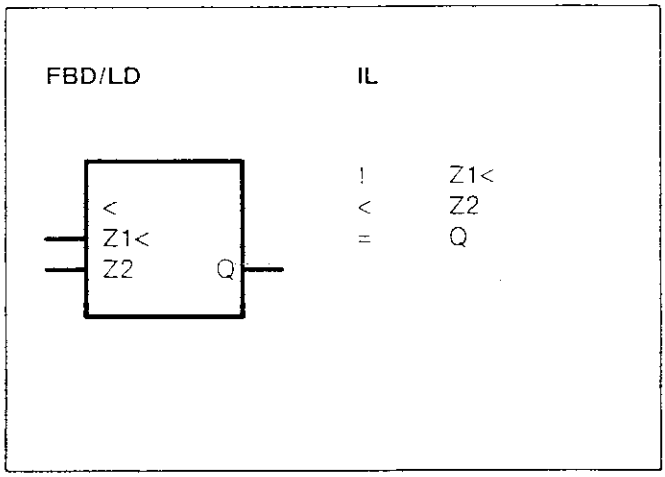
CE IL Definition

```
00000    IBA     0        Nr      Block No. (preset to 0)
00001    INTK
00002            PP  0    x       Input WORD  (controlled variable)
00003            PP  0    TI/T    Input WORD  (integration time)
00004            PP  0    R       Input BINARY  (reset to 0)
00005            PP  0    STOP    Input BINARY  (integrator stop)
00006            PP  0    S       Input BINARY  (set to INIT)
00007            PP  0    INIT    Input WORD  (initial value)
00008            PP  0    K       Input WORD  (proportinal coefficient)
00009            PP  0    OG      Input WORD  (high limit)
00010            PP  0    UG      Input WORD  (iow limit)
00011            PP  0    y       Output WORD (manipulated variable)
00012            PP  0    y=OG    Output BINARY (OG reached)
00013            PP  0    y=UG    Output BINARY (UG reached)
```

LESS THAN <

The value of the operand at the input Z1< is compared to the value of the operand at the input Z2.

If the value at Z1< is less than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1< is equal to or greater than Z2.



Parameters

Z1<	WORD	EW, MW, AW, KW	Value to be compared
Z2	WORD	EW, MW, AW, KW	Comparison value
Q	BINARY	A, M, S	Result of the comparison

CE Data

Runtime:	
Basic runtime:	< 12 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701 R801 / 35 ZE 93 R101

Description

The value of the operand at the input Z1< is compared to the value of the operand at the input Z2.

If the value at Z1< is less than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1< is equal to or greater than Z2.

The inputs can be negated, but not duplicated. The output can be inverted, but not duplicated.

Number range

Integer word (16 Bit)

The following specially applies here to the non-negated inputs:

- Low limit 8000 H -32768
- High limit 7FFF H +32767

The following generally applies:

- Low limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H /////

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

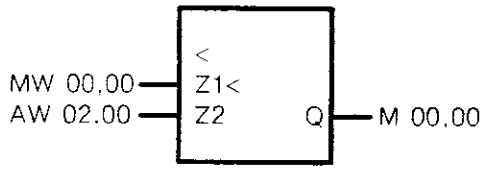
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

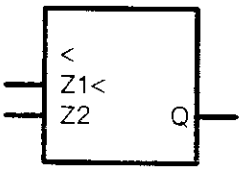
FBD/LD



IL

```
!     MW 00,00
<     AW 02,00
=     M  00,00
```

CE FBD Definition



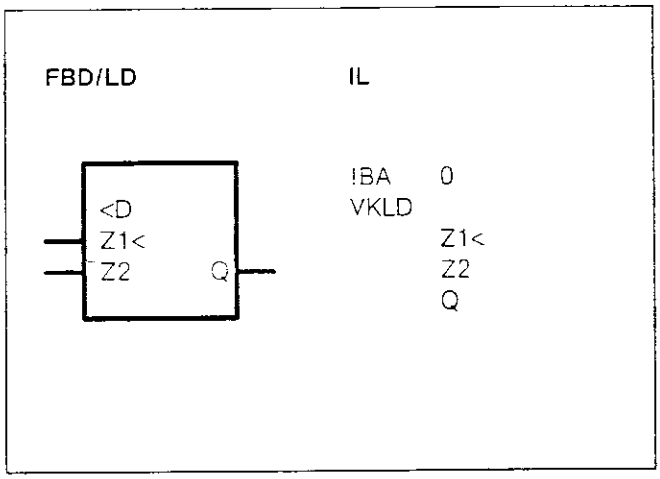
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1<	E	W	Y	P	Y	0	0
Z2	E	W	Y	P	Y	0	0
Q	A	L	Y	P	Y	0	0

CE IL Definition

00000	!	PP	Z1<	Input WORD
00002	<	PP	Z2	Input WORD
00004	=	PP	Q	Output BINARY

LESS THAN, DOUBLE WORD <D

The value of the operand at the input Z1< is compared to the value of the operand at the input Z2. If the value at Z1< is less than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1< is equal to or greater than Z2.



Parameters

Z1<	DOUBLE WORD	MD, KD	Value to be compared
Z2	DOUBLE WORD	MD, KD	Comparison value
Q	BINARY	A, M,	Result of the comparison

CE Data

Runtime:	
Basic runtime:	35 ... 36 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the operand at the input Z1< is compared to the value of the operand at the input Z2. If the value at Z1< is less than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1< is equal to or greater than Z2.

The inputs can be negated, but not duplicated. The output can be inverted, but not duplicated.

Number range

Integer double word (32 Bit)

The following specially applies here to the inputs Z1< and Z2:

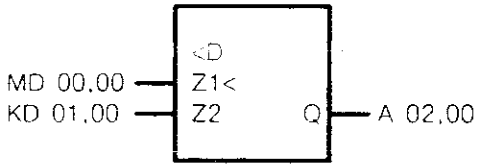
• Low limit:	8000 0000 H	-2 147 483 648
• High limit:	7FFF FFFF H	+2 147 483 647

The following generally applies:

• Low limit:	8000 0001 H	-2 147 483 647
• High limit:	7FFF FFFF H	+2 147 483 647
• Inadmissible value:	8000 0000 H	---

Example

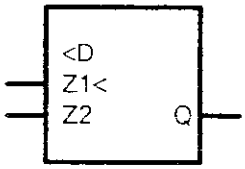
FBD/LD



IL

```
!BA    0
VKLD
       MD 00,00
       KD 01,00
       A  02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1<	E	D	N	P	Y	0	0
Z2	E	D	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

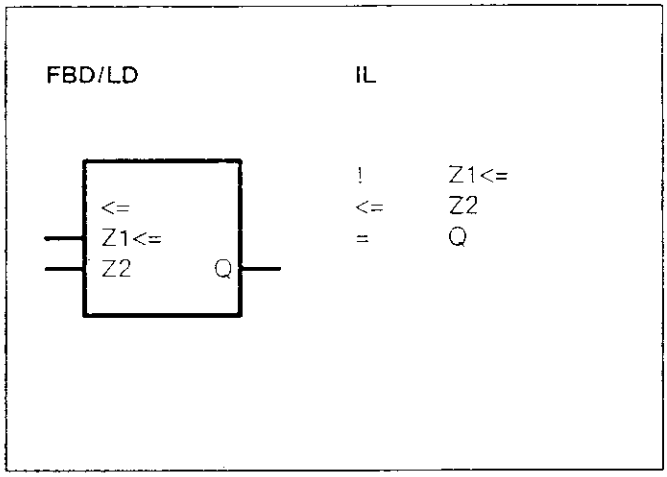
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	VKLD			
00002		PP 0	Z1<	Input DOUBLE WORD
00003		PP 0	Z2	Input DOUBLE WORD
00004		PP 0	Q	Output BINARY

LESS THAN OR EQUAL <=

The value of the operand at the input Z1<= is compared to the value of the operand at the input Z2.

The state 1 is allocated to the operand at the output Q if the value at Z1<= is less than or equal to the value at Z2. The state 0 is allocated to Q if Z1<= is greater than Z2.



Parameters

Z1<=	WORD	EW, MW, AW, KW	Value to be compared
Z2	WORD	EW, MW, AW, KW	Comparison value
Q	BINARY	A, M, S	Result of the comparison

CE Data

Runtime:	
Basic runtime:	< 12 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The value of the operand at the input Z1<= is compared to the value of the operand at the input Z2.

The state 1 is allocated to the operand at the output Q if the value at Z1<= is less than or equal to the value at Z2. The state 0 is allocated to Q if Z1<= is greater than Z2.

The inputs can be negated, but not duplicated. The output can be inverted, but not duplicated.

Number range

Integer word (16 Bit)

The following specially applies here to the non-negated inputs:
- Low limit 8000 H -32768
- High limit 7FFF H +32767

The following generally applies:
- Low limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

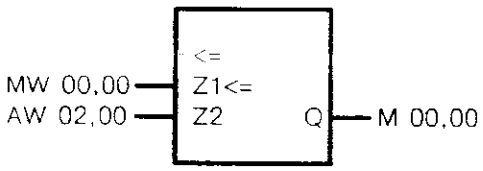
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

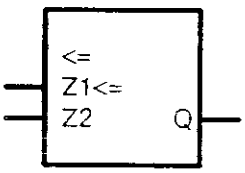
FBD/LD



IL

```
!     MW 00,00
<=    AW 02,00
=     M  00,00
```

CE FBD Definition



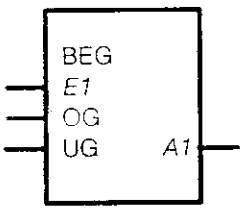
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1<=	E	W	Y	P	Y	0	0
Z2	E	W	Y	P	Y	0	0
Q	A	L	Y	P	Y	0	0

CE IL Definition

00000	!	PP 0	Z1<=	Input WORD
00002	<	PP 0	Z2	Input WORD
00004	=	PP 0	Q	Output BINARY

The value of the operand at the input E1 is limited to the range between the high and low limits.

FBD/LD



IL

```
!BA   0
BEG
      E1
      OG
      UG
      A1
```

Parameters

E1	WORD	EW, MW, AW, KW	Input value
OG	WORD	EW, MW, AW, KW	High limit
UG	WORD	EW, MW, AW, KW	Low limit
A1	WORD	AW, MW	Limited value

CE Data

Runtime:	
Basic runtime:	32 ... 35 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

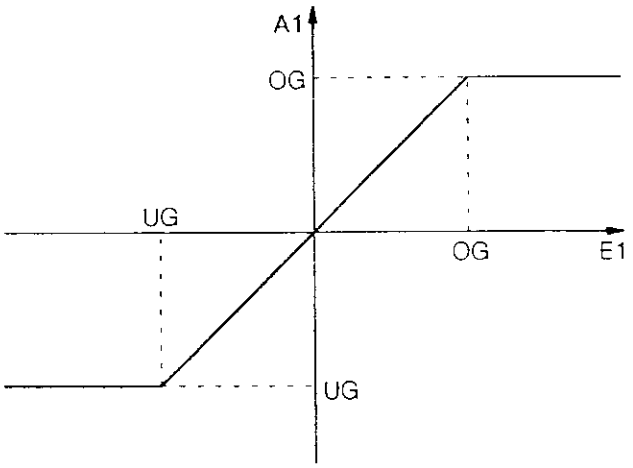
Description

The value of the operand at the input E1 is limited to the range between the high and low limits.

The high limit is specified by the operand at the OG input and the low limit is specified by the one at the UG input.

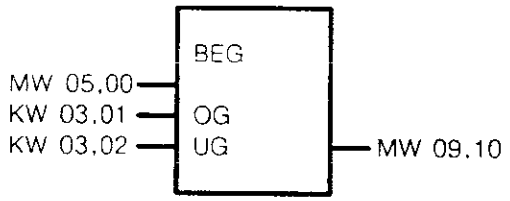
The following applies: A1 = UG for E1 ≤ UG
A1 = E1 for UG ≤ E1 ≤ OG
A1 = OG for E1 ≥ OG

The inputs and the output can neither be duplicated nor negated.



Example

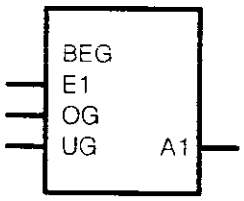
FBD/LD



IL

```
IBA    0
BEG
       MW 05.00
       KW 03,01
       KW 03,02
       MW 09,10
```

CE FBD Definition



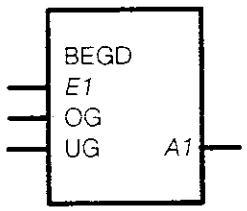
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
OG	E	W	N	P	Y	0	0
UG	E	W	N	P	Y	0	0
A1	A	W	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	BEG			
00002		PP 0	E1	Input WORD (input value)
00003		PP 0	OG	Input WORD (high limit)
00004		PP 0	UG	Input WORD (low limit)
00005		PP 0	A1	Output WORD (limited value)

The value of the operand at the input E1 is limited to the range between the high and low limits.

FBD/LD



IL

```
!BA   0
BEGD
      E1
      OG
      UG
      A1
```

Parameters

E1	DOUBLE WORD	MD,	Input value
OG	DOUBLE WORD	MD, KD	High limit
UG	DOUBLE WORD	MD, KD	Low limit
A1	DOUBLE WORD	MD	Limited value

CE Data

Runtime:	
Basic runtime:	30 ... 38 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

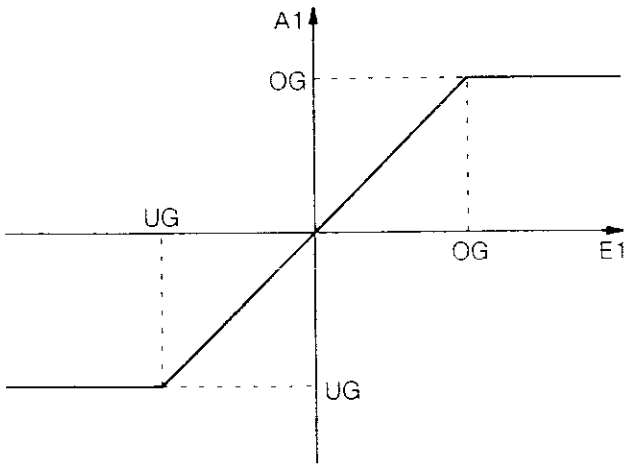
Description

The value of the operand at the input E1 is limited to the range between the high and low limits.

The high limit is specified by the operand at the OG input and the low limit is specified by the one at the UG input.

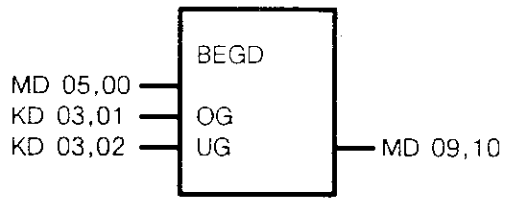
The following applies: A1 = UG for E1 ≤ UG
A1 = E1 for UG ≤ E1 ≤ OG
A1 = OG for E1 ≥ OG

The inputs and the output can neither be duplicated nor negated.



Example

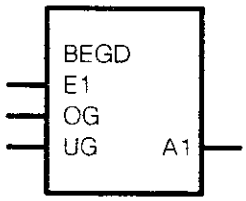
FBD/LD



IL

```
!BA    0
BEGD
       MD 05,00
       KD  03,01
       KD  03,02
       MD 09,10
```

CE FBD Definition



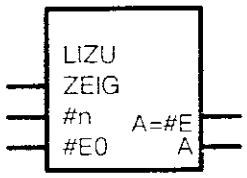
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
OG	E	D	N	P	Y	0	0
UG	E	D	N	P	Y	0	0
A1	A	D	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	BEGD			
00002		PP 0	E1	Input DOUBLE WORD (input value)
00003		PP 0	OG	Input DOUBLE WORD (high limit)
00004		PP 0	UG	Input DOUBLE WORD (low limit)
00005		PP 0	A1	Output DOUBLE WORD (limited value)

This function block has a list of direct constants at its inputs. With a list pointer, it selects a constant out of this list and outputs it through its output.

FBD/LD



IL

```
!BA   0
LIZU
      ZEIG
      #n
      #E0
      A=#E
      A
```

Parameters

ZEIG	WORD	AW, EW, MW, KW	Pointer to the list of direct constants
#n	DIRECT CONSTANT	#, #H	Number of direct constants in the list
#E0	DIRECT CONSTANT	#, #H	List of direct constants; the input is capable of duplication
A	WORD	AW, MW	Selected direct constant
A=#E	BINARY	A, M	0 ≤ ZEIG < #n, i. e. pointer in the valid range

CE Data

Runtime:	
Basic runtime:	19 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block has a list of direct constants at its inputs. With a list pointer, it selects a constant out of this list and outputs it through its output.

The inputs and outputs cannot be negated/inverted. The input #E0 can be duplicated.

ZEIG WORD

The pointer to the direct constant to be selected from the list is specified at the input ZEIG.

The following affiliations apply:
ZEIG = 0 -> Direct constant at E0
ZEIG = 1 -> Direct constant at E1

. . .

ZEIG = n-1 -> Direct constant at En-1

The value at the input ZEIG is subjected to a validity check. The result of this range check is signalled at the output A=#E.

Allowed range:
0 ≤ ZEIG ≤ n-1
Where n: Number of the inputs #E0...#En-1.
No allocation to the output A takes place if the value at the input ZEIG is outside the allowed range.

#n DIRECT CONSTANT

The number n of the direct constants planned at the inputs #E0...#En-1 is specified at the input #n. This is specified as a direct constant.

#E0...#En-1 DIRECT CONSTANT
The input #E0 is capable of duplication (#E0...#En-1). The direct constants out of which one is selected with the value at the input ZEIG and allocated to the output A are specified at the inputs #E0...#En-1.

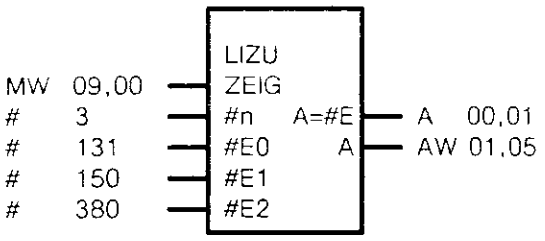
A=#E BINARY
Whether the value of the list pointer (input ZEIG) is within the allowed range is specified at the output A=#E.
Allowed range:
$0 \leq ZEIG \leq n-1$
where n: Number of the direct constants at #E0...#En-1.
The following applies:
ZEIG in the allowed range -> A=#E = 1
ZEIG in the forbidden range -> A=#E = 0

If the list pointer has a forbidden value, no allocation of a constant to the output A takes place because no constant can be selected. In this case, the output A is *not* updated.

A WORD
The value of the selected direct constant is allocated to the operand at the output A.

Example

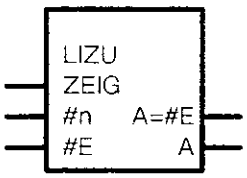
FBD/LD



IL

```
!BA   0
LIZU
      MW  09.00
      #   3
      #   131
      #   150
      #   380
      A    00,01
      AW   01,05
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
ZEIG	E	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
#E	K	W	N	P	Y	1	0
A=#E	A	L	N	P	Y	0	0
A	A	W	N	P	Y	0	0

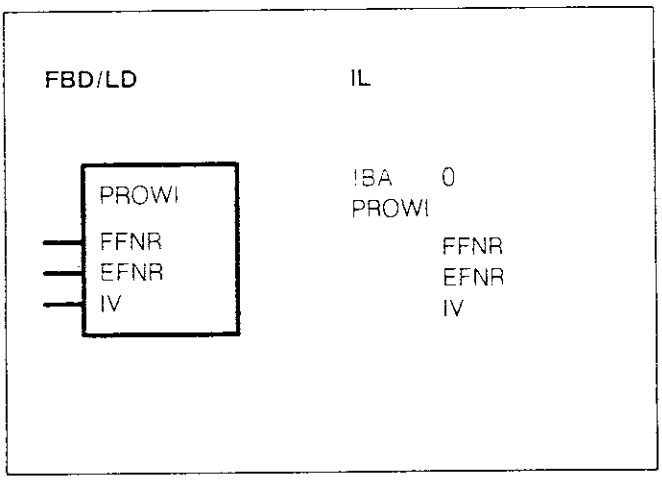
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	LIZU			
00002		PP 0	ZEIG	Input WORD (pointer)
00003		PP 0	#n	# CONSTANT (number of #Es)
	[	1		
00004		PP 1	#E	# CONSTANT (input values)
	]	1		
00005		PP 0	A=#E0	Output BINARY (0 ≤ ZEIG < #n)
00006		PP 0	A	Output WORD (output value)

LOADING MEASUREMENT WINDOW FRAMES (OMS-F) PROWI

On an iconic processor 35 IV 90, this function block calls a single window or a whole window sequence that has been defined with a GENWI block in "teach-in mode".

This block is needed when using the video sensor OMS-F.



Parameters

FFNR	WORD	EW, AW, MW, KW	Window sequence No.
EFNR	WORD	EW, AW, MW, KW	Single window No.
IV	WORD	EW, AW, MW, KW	Number of the 35 IV 90 module

CE Data

Runtime:	
Basic runtime:	790 µs for *one* single window 890 µs for *all* single windows
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

On an iconic processor 35 IV 90, this function block calls a single window or a whole window sequence that has been defined with a GENWI block in "teach-in mode".

The coordinates of this window sequence or the single window have been determined beforehand in "teach-in mode" by means of the PLC command X-WINDOW (XW) and stored in the PLC.

At the program run time, measurement then takes place on the iconic processor to which the window coordinates are loaded. The number of this iconic processor does not need to be the same as the one specified during "teach-in". On the iconic processor, a pixel counter is assigned to each measurement window which is activated during measurement. The pixel counter has the same designation as the affiliated window.

FFNR WORD

Window sequence number

The number of the required window sequence is specified with the operand at this input.
Value range: 1... 512.
A window sequence consists of a maximum of 8 windows.

EFNR WORD

Single window number.

The number of the required single window is specified with the operand at this input. However, all windows of the specified window sequence can also be selected.

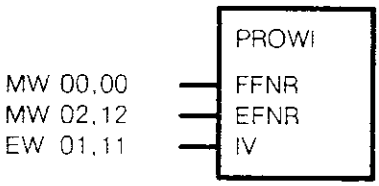
EFNR = 0:	All windows of the window sequence are selected.
EFNR = 1...8:	Number of the window selected.
Value range:	0-8

IV WORD
Number of the 35 IV 90 module

The number of the required 35 IV 90 module with which measurement is to take place is specified with the operand at this input.
Value range: 0...5

Example

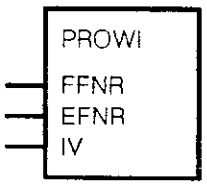
FBD/LD



IL

```
!BA    0
PROWI
       MW 00,00
       MW 02,12
       EW 01,11
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FFNR	E	W	N	P	Y	0	0
EFNR	E	W	N	P	Y	0	0
IV	E	W	N	P	Y	0	0

CE IL Definition

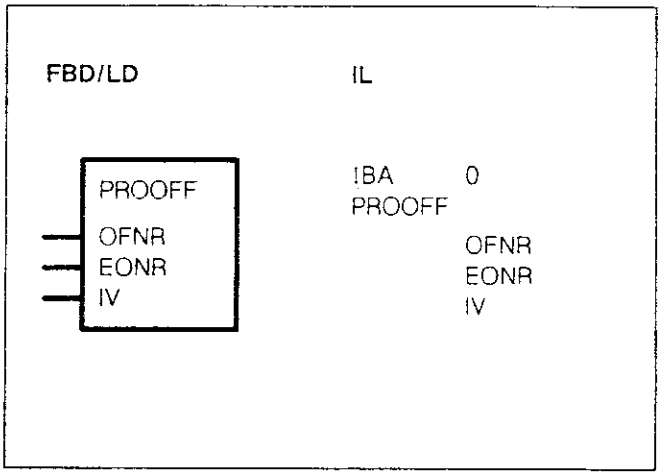
00000	!BA	0	Nr	Block No. (preset to 0)
00001	PROWI			
00002		PP 0	FFNR	Input WORD (window sequence number)
00003		PP 0	EFNR	Input WORD (single window number)
00004		PP 0	IV	Input WORD (IV No.)

LOADING THE RESULT COUNTERS WITH OFFSET VALUES (OMS-F)

PROOFF

On a 35 IV 90 module, this function block loads a single counter with an offset value or a complete string of counters with offset values. The offset values have been entered in the PLC beforehand in teach-in mode by means of the PLC command X-offset (XO) and stored there.

This block is needed when using the video sensor OMS-F.



Parameters

OFNR	WORD	EW, AW, MW, KW	Number of the offset sequence
EONR	WORD	EW, AW, MW, KW	Number of the single offset
IV	WORD	EW, AW, MW, KW	Number of the 35 IV 90 module

CE Data

Runtime:	
Basic runtime:	106 µs
Additional runtime:	no
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

On a 35 IV 90 module, this function block loads a single counter with an offset value or a complete string of counters with offset values. The offset values have been entered in the PLC beforehand in teach-in mode by means of the PLC command X-offset (XO) and stored there. Each single offset and each offset sequence has the same designation as the single counter and counter sequence to be loaded with these offsets. Presetting a result value counter with an offset serves to trigger an OVERFLOW in this counter once it has counted a quite specific number of pixels.

Example: Offset = 5 -> once the counter has counted 5 pixels, it has an OVERFLOW.

By means of the READC block, the status word containing the overflow flags of all result value counters of a 35 IV 90 module can be read. By evaluation of the status word alone, it is now possible to determine which counter of a 35 IV 90 module has counted the number of pixels required by the user after a measurement. In this way it is possible to determine, for instance, whether a specific number of black points occurs in a white area, i.e. the white paint application is poor, and the measured object is a reject.

OFNR WORD

Number of the offset sequence

The number of the offset sequence from which an offset value or all offset values are to be loaded from the PLC to a 35 IV 90 module is specified with the operand at this input.
Value range: 1 ... 10
An offset sequence consists of a maximum of 8 single offsets.

EONR WORD
Number of the single offset

The number of the offset value to be loaded from the PLC into the affiliated counter on a 35 IV 90 module is specified with the operand at this input. When a number equal to 0 is specified, all offset values in the sequence are loaded from the PLC to the affiliated counters of a 35 IV 90 module.

EONR = 0: All offset values of the sequence are loaded.
EONR = 1 ... 8: Number of the offset value to be loaded.
Value range: 0 ... 8.

IV WORD
Number of the 35 IV 90 module.

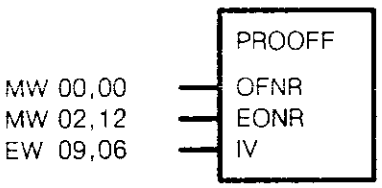
The number of the required 35 IV 90 module with which measurement is to take place is specified with the operand at this input.
Value range: 0...5

LOADING THE RESULT COUNTERS WITH OFFSET VALUES (OMS-F)

PROOFF

Example

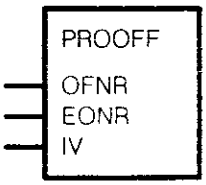
FBD/LD



IL

```
!BA      0
PROOFF
         MW 00,00
         MW 02,12
         EW 09,06
```

CE FBD Definition



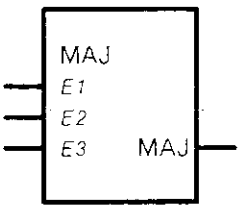
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
OFNR	E	W	N	P	Y	0	0
EOFR	E	W	N	P	Y	0	0
IV	E	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	PROOFF			
00002		PP 0	OFNR	Input WORD (offset sequence No.)
00003		PP 0	EOFNR	input WORD (single offset No.)
00004		PP 0	IV	Input WORD (IV No.)

This connection element realizes a MAJORITY element.

FBD/LD



IL

```
! BA   0
  MAJ
       E1
       E2
       E3
       MAJ
```

Parameters

E1	BINARY	E, M, A, S, K	Operand 1
E2	BINARY	E, M, A, S, K	Operand 2
E3	BINARY	E, M, A, S, K	Operand 3
MAJ	BINARY	M, A	Result

CE Data

Runtime:	
Basic runtime:	36 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

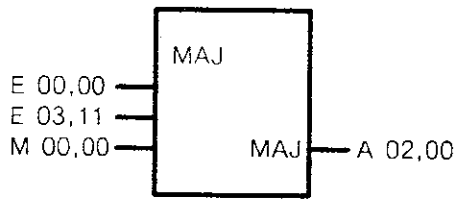
This connection element realizes a MAJORITY element.

If at least 2 of the 3 binary operands at the inputs E1, E2 and E3 have the state 1, then the state 1 is allocated to the binary operand at the output MAJ. The state 0 is allocated if this is not the case.

The inputs and the output can neither be duplicated nor negated.

EXAMPLE

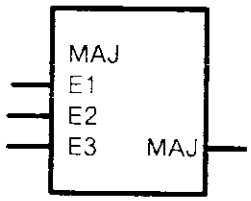
FBD/LD



IL

```
! BA   0
  MAJ
       E 00,00
       E 03,11
       M 00,00
       A 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	L	N	P	N	0	0
E2	E	L	N	P	N	0	0
E3	E	L	N	P	N	0	0
MAJ	A	L	N	P	Y	0	0

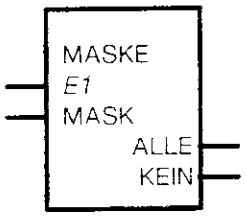
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MAJ			
00002		PP 0	E1	Input BINARY
00003		PP 0	E2	Input BINARY
00004		PP 0	E3	Input BINARY
00005		PP 0	MAJ	Output BINARY (majority)

The individual bits of the operand at the input E1 are compared to the bits of the operand at the input MASK.

The result of the comparison is signalled at the outputs ALLE and KEIN.

FBD/LD



IL

```
!BA     0
MASKE
        E1
        MASK
        ALLE
        KEIN
```

Parameters

E1	WORD	EW, AW, MW, KW	Input value
MASK	WORD	EW, AW, MW, KW	Mask
ALLE	BINARY	A, M	All bits agree
KEIN	BINARY	A, M	No bit agrees

CE Data

Runtime:	
Basic runtime:	35 - 38 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The individual bits of the operand at the input E1 are compared to the bits of the operand at the input MASK.

The result of the comparison is signalled at the outputs ALLE and KEIN.

If at least *all* bits are set on the operand at the input E1 which are set on the operand at the input MASK, the following applies to the outputs:

ALLE = 1
KEIN = 0

If *none* of the bits are set on the operand at the input E1 which are set on the operand at the input MASK, the following applies to the outputs:

ALLE = 0
KEIN = 1

If only *some* of the bits are set on the operand at the input E1 which are set on the operand at the input MASK, the following applies to the outputs:

ALLE = 0
KEIN = 0

The inputs and outputs can neither be duplicated nor negated nor inverted.

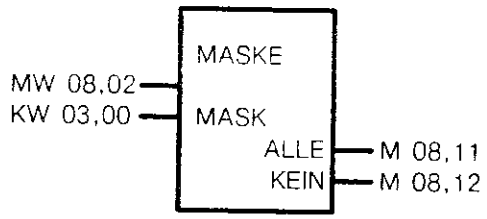
Example

```
E1   : X1111XX11XXXX11X   ALLE = 1
MASK : 0111100110000110   KEIN = 0

E1   : 00XXXXX0XXX0XXXX   ALLE = 0
MASK : 1100000100010000   KEIN = 1

E1   : 1X1X1XXX10XXXX10   ALLE = 0
MASK : 0010100011000011   KEIN = 0
```

X: These bits may have any value (don't care)

Example

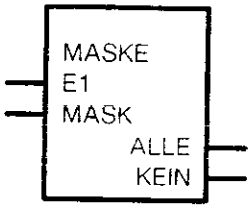
FBD/LD



IL

```
!BA     0
MASKE
        MW 08,02
        KW 03,00
        M  08,11
        M  08,12
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
MASK	E	W	N	P	Y	0	0
ALLE	A	L	N	P	Y	0	0
KEIN	A	L	N	P	Y	0	0

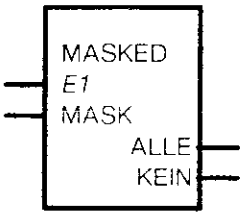
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MASKE			
00002		PP 0	E1	Input WORD (input value)
00003		PP 0	MASK	Input WORD (mask)
00004		PP 0	ALLE	Output BINARY (all bits agree)
00005		PP 0	KEIN	Output BINARY (no bit agrees)

The individual bits of the operand at the input E1 are compared to the bits of the operand at the input MASK.

The result of the comparison is signalled at the outputs ALLE and KEIN.

FBD/LD



IL

```
!BA     0
MASKED
        E1
        MASK
        ALLE
        KEIN
```

Parameters

E1	DOUBLE WORD	MD, KD	Input value
MASK	DOUBLE WORD	MD, KD	Mask
ALLE	BINARY	A, M	All bits agree
KEIN	BINARY	A, M	No bit agrees

CE Data

Runtime:	
Basic runtime:	42 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 81 R301 / 35 ZE 93 R201

Description

The individual bits of the operand at the input E1 are compared to the bits of the operand at the input MASK.

The result of the comparison is signalled at the outputs ALLE and KEIN.

If at least *all* bits are set on the operand at the input E1 which are set on the operand at the input MASK, the following applies to the outputs:
ALLE = 1
KEIN = 0

If *none* of the bits are set on the operand at the input E1 which are set on the operand at the input MASK, the following applies to the outputs:
ALLE = 0
KEIN = 1

If only *some* of the bits are set on the operand at the input E1 which are set on the operand at the input MASK, the following applies to the outputs:
ALLE = 0
KEIN = 0

The inputs and outputs can neither be duplicated nor negated nor inverted.

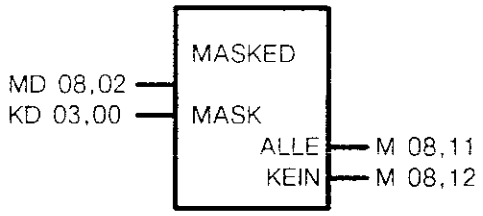
Example

```
E1     :  X1111XX11XXXX11X   X1XXXXX1XXX1XXXX   ALLE = 1
MASK   :  0111100110000110   0100000100010000   KEIN = 0


E1     :  X0000XX00XXXX00X   X0XXXXX0XXX0XXXX   ALLE = 0
MASK   :  0111100110000110   0100000100010000   KEIN = 1


E1     :  X1011XX10XXXX11X   X1XXXXX0XXX1XXXX   ALLE = 0
MASK   :  0111100110000110   0100000100010000   KEIN = 0
```

X: These bits may have any values (don't care).

Example

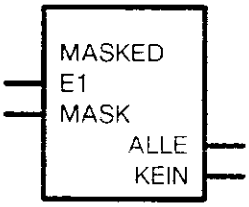
FBD/LD



IL

```
!BA     0
MASKED
        MD 08,02
        KD 03,00
        M  08,11
        M  08,12
```

CE FBD Definition



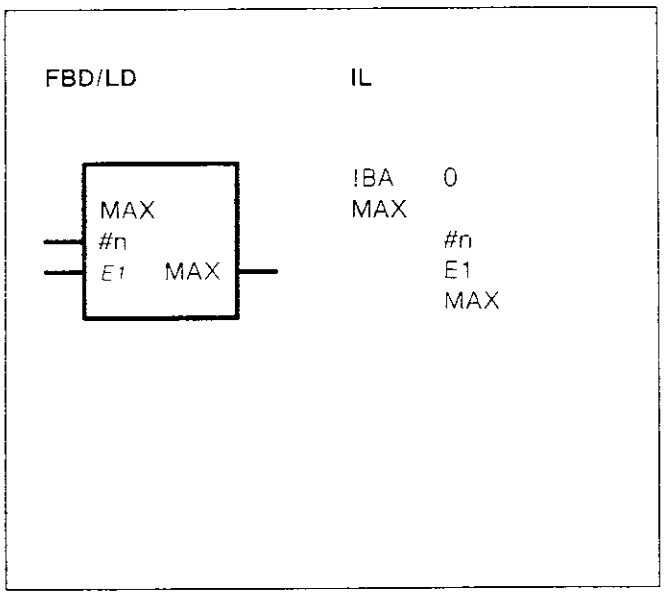
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
MASK	E	D	N	P	Y	0	0
ALLE	A	L	N	P	Y	0	0
KEIN	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MASKED			
00002		PP 0	E1	Input DOUBLE WORD (input value)
00003		PP 0	MASK	Input DOUBLE WORD (mask)
00004		PP 0	ALLE	Output BINARY (all bits agree)
00005		PP 0	KEIN	Output BINARY (no bit agrees)

MAXIMUM VALUE GENERATOR — MAX

From n operands, this function block generates the maximum value and allocates it to the output.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of inputs
E1	WORD	EW, MW, AW, KW	1st input value; the input is capable of duplication
MAX	WORD	AW, MW	Output (maximum value)

CE Data

Runtime:	
Basic runtime:	22 µs
Additional runtime:	9 µs per planned input E1
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

From n operands, this function block generates the maximum value and allocates it to the output.

#n DIRECT CONSTANT (#, #H)
The number of operands from which the maximum value is to be determined is specified at the input #n. This is specified as a direct constant.
n > 0 applies.

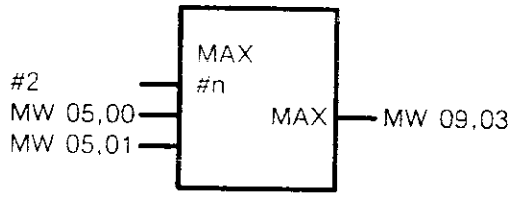
E1 WORD
The input must be duplicated as many times as necessary until the number of inputs specified at the input #n exists. On the basis of the values of the operands at these inputs, the maximum value is determined and is allocated to the output MAX.

MAX WORD
The maximum value from the n input operands is available at the output MAX.

The input E1 can be duplicated.
The output can neither be duplicated nor negated.

Example

FBD/LD



IL

```
!BA   0
MAX
      #2
      MW 05,00
      MW 05,01
      MW 09,03
```

CE FBD Definition

Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
E1	E	W	N	P	N	1	0
MAX	A	W	N	P	Y	0	0

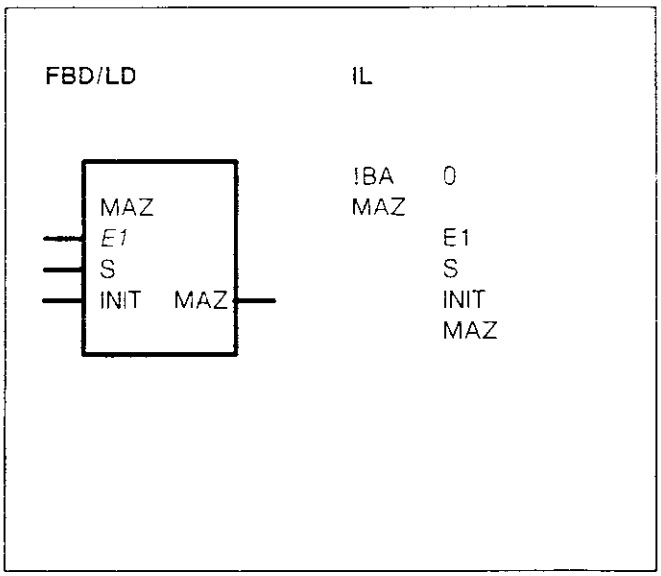
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MAX			
00002		PP 0	#n	# CONSTANT (number of inputs)
	[	1		
00003		PP 1	E1	Input WORD
	]	1		
00004		PP 0	MAX	Output WORD (max. value)

MAXIMUM VALUE GENERATOR AS A FUNCTION OF TIME

MAZ

This function block determines, on the basis of the time progression of a signal, its maximum value occurring up to the current point in time.



Parameters

E1	WORD	EW, MW, AW, KW	Input value whose time maximum is to be determined
S	BINARY	E, M, A, S, K	Set input
INIT	WORD	EW, MW, AW, KW	Initial value
MAZ	WORD	AW, MW	Maximum value

CE Data

Runtime:	
Basic runtime:	36 ... 40 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	2 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block determines, on the basis of the time progression of a signal, its maximum value occurring up to the current point in time.

The value of the operand at the input E1 is compared to the previously occurring maximum value. If the input value E1 is higher than the previously occurring maximum, the input value is the new maximum value and is allocated to the operand at the output MAZ.

If the input value E1 is less than the previously occurring maximum value, the previous maximum value is allocated to the output.

The output MAZ is set to the value of the operand at the input INIT (initial value) with the 0–>1 edge at the binary input S.

The following applies:

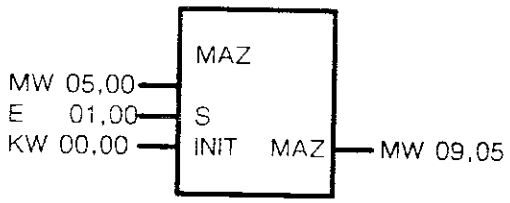
E1 < MAZ -> MAZ = MAZ
E1 ≥ MAZ -> MAZ = E1
S = 0–>1 edge -> MAZ = INIT

The inputs and the output can neither be duplicated nor negated/inverted.

MAXIMUM VALUE GENERATOR AS A FUNCTION OF TIME

Example

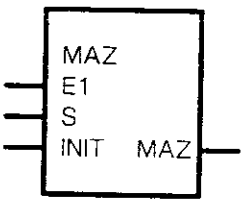
FBD/LD



IL

```
!BA    0
MAZ
       MW 05,00
       E  01,00
       KW 00,00
       MW 09,05
```

CE FBD Definition



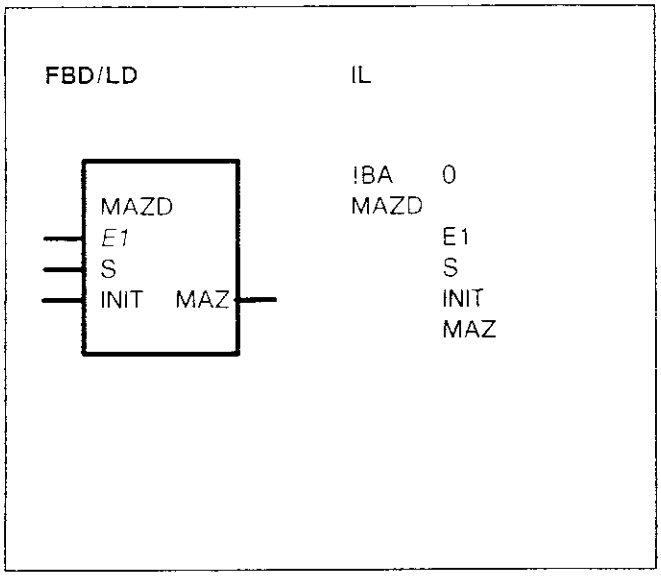
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
S	E	L	N	P	Y	0	0
INIT	E	W	N	P	Y	0	0
MAZ	A	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MAZ			
00002		PP 0	E1	Input WORD (input value)
00003		PP 0	S	Input BINARY (set input)
00004		PP 0	INIT	Input WORD (initial value)
00005		PP 0	MAZ	Output WORD (max. value)

MAXIMUM VALUE GENERATOR AS A FUNCTION OF TIME, DOUBLE WORD — MAZD

This function block determines, on the basis of the time progression of a signal, its maximum value occurring up to the current point in time.



Parameters

E1	DOUBLE WORD	MD, KD	Input value whose time maximum is to be determined
S	BINARY	E, M, A, S, K	Set input
INIT	DOUBLE WORD	MD	Initial value
MAZ	DOUBLE WORD	MD	Maximum value

CE Data

Runtime:	
Basic runtime:	49 ... 58 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	3 words
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block determines, on the basis of the time progression of a signal, its maximum value occurring up to the current point in time.

The value of the operand at the input E1 is compared to the previously occurring maximum value. If the input value E1 is higher than the previously occurring maximum, the input value is the new maximum value and is allocated to the operand at the output MAZ.

If the input value E1 is less than the previously occurring maximum value, the previous maximum value is allocated to the output.

The output MAZ is set to the value of the operand at the input INIT (initial value) with the 0->1 edge at the binary input S.

The following applies:

E1 < MAZ	-> MAZ = MAZ
E1 ≥ MAZ	-> MAZ = E1
S = 0->1 edge	-> MAZ = INIT

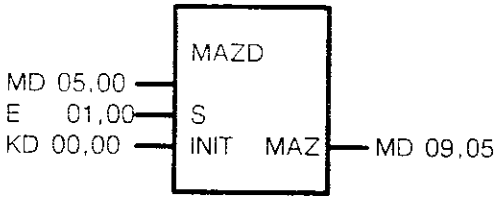
The inputs and the output can neither be duplicated nor negated/inverted.

MAXIMUM VALUE GENERATOR AS A FUNCTION OF TIME, DOUBLE WORD

MAZD

Example

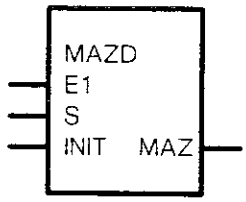
FBD/LD



IL

```
!BA    0
MAZD
       MD 05,00
       E  01,00
       KD 00,00
       MD 09,05
```

CE FBD Definition



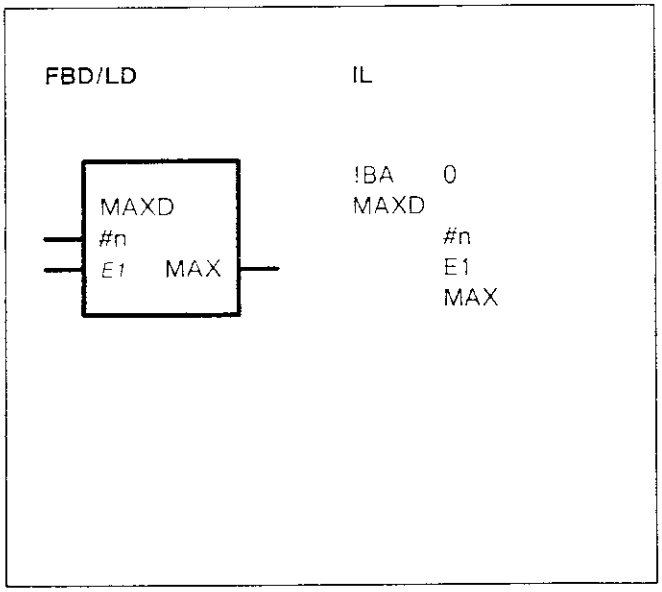
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
S	E	L	N	P	Y	0	0
INIT	E	D	N	P	Y	0	0
MAZ	A	D	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MAZD			
00002		PP 0	E1	input DOUBLE WORD (input value)
00003		PP 0	S	Input BINARY (set input)
00004		PP 0	INIT	Input DOUBLE WORD (initial value)
00005		PP 0	MAZ	Output DOUBLE WORD (max. value)

MAXIMUM VALUE GENERATOR, DOUBLE WORD — MAXD

From n operands, this function block generates the maximum value and allocates it to the output.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of inputs
E1	DOUBLE WORD	MD, KD	1st input value; the input is capable of duplication
MAX	DOUBLE WORD	MD	Output (maximum value)

CE Data

Runtime:	
Basic runtime:	29 µs
Additional runtime:	13 µs per planned input E1
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

From n operands, this function block generates the maximum value and allocates it to the output.

#n DIRECT CONSTANT (#, #H)
The number of operands from which the maximum value is to be determined is specified at the input #n. This is specified as a direct constant.
n > 0 applies.

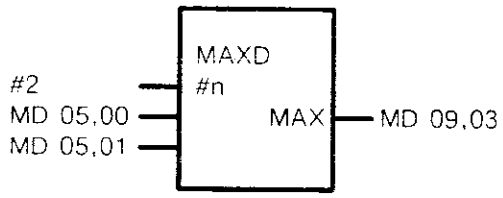
E1 DOUBLE WORD
The input must be duplicated as many times as necessary until the number of inputs specified at the input #n exists. On the basis of the values of the operands at these inputs, the maximum value is determined and is allocated to the output MAX.

MAX DOUBLE WORD
The maximum value from the n input operands is available at the output MAX.

The input E1 can be duplicated.
The output can neither be duplicated nor negated.

Example

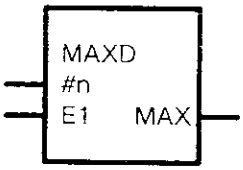
FBD/LD



IL

```
!BA    0
MAXD
       #2
       MD 05,00
       MD 05,01
       MD 09,03
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
E1	E	D	N	P	N	1	0
MAX	A	D	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MAXD			
00002		PP 0	#n	# CONSTANT (number of inputs)
	[	1		
00003		PP 1	E1	Input DOUBLE WORD
	]	1		
00004		PP 0	MAX	Output DOUBLE WORD (max. value)

MINIMUM VALUE GENERATOR — MIN

This function block determines the minimum value from n operands and allocates it to the output.

FBD/LD

IL

```
IBA   0
MIN
      #n
      E1
      MIN
```

Parameter

#n	DIRECT CONSTANT	#, #H	Number of inputs
E1	WORD	EW, MW, AW, KW	1st input value; the input is capable of duplication
MIN	WORD	AW, MW	Output (minimum value)

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	7 µs per planned input E1
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block determines the minimum value from n operands and allocates it to the output.

#n DIRECT CONSTANT (#, #H)
The number of operands from which the minimum value is to be determined is specified at the input #n. This is specified as a direct constant.
n > 0 applies.

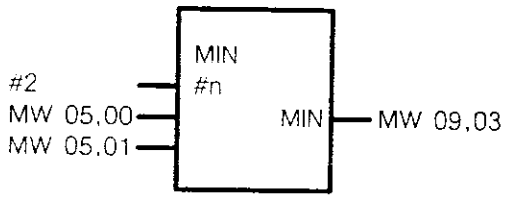
E1 WORD
The input E1 must be duplicated as often as the number of inputs specified at the input #n exists. On the basis of the values of the operands at these inputs, the minimum value is determined and is allocated to the output MIN.

MIN WORD
The minimum value of the n input operands is available at the output MIN.

The input E1 can be duplicated.
The output can neither be duplicated nor negated.

Example

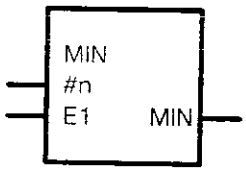
FBD/LD



IL

```
!BA   0
MIN
      #2
      MW 05,00
      MW 05,01
      MW 09,03
```

CE FBD Definition



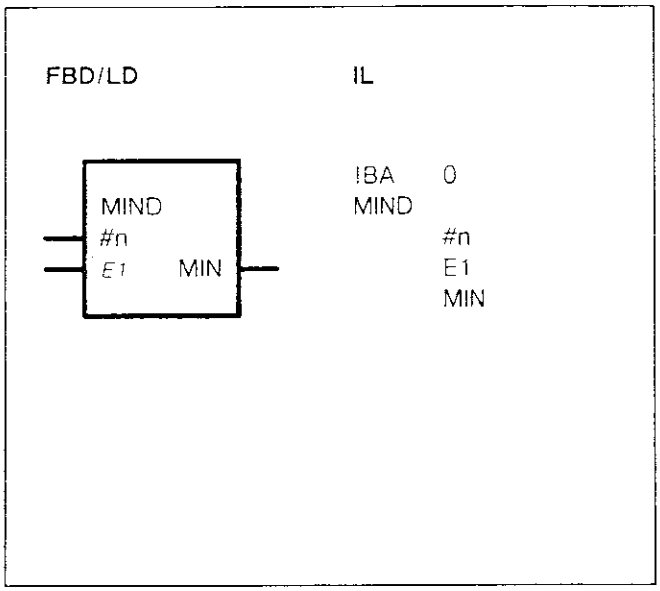
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
E1	E	W	N	P	N	1	0
MIN	A	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MIN			
00002		PP 0	#n	# CONSTANT (number of inputs)
	[	1		
00003		PP 1	E1	Input WORD
	]	1		
00004		PP 0	MIN	Output WORD (min. value)

MINIMUM VALUE GENERATOR, DOUBLE WORD — MIND

This function block determines the minimum value from n operands and allocates this value to the output.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of inputs
E1	DOUBLE WORD	MD, KD	1st input value; the input is capable of duplication
MIN	DOUBLE WORD	MD	Output (minimum value)

CE Data

Runtime:	
Basic runtime:	24 µs
Additional runtime:	14 µs per planned input E1
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block determines the minimum value from n operands and allocates this value to the output.

#n DIRECT CONSTANT (#, #H)
The number of operands from which the minimum value is to be determined is specified at the input #n. This is specified as a direct constant.
n > 0 applies.

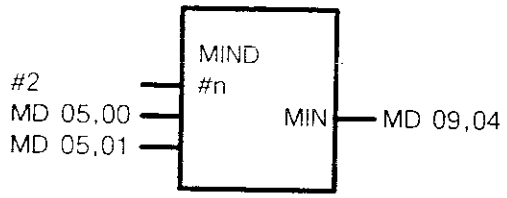
E1 DOUBLE WORD
The input E1 must be duplicated until the number of inputs specified at the input #n exists. On the basis of the values of the operands at these inputs, the minimum value is determined and is allocated to the output MIN.

MIN DOUBLE WORD
The minimum value of the n input operands is available at the output MIN.

The input E1 can be duplicated.
The output can neither be duplicated nor negated.

Example

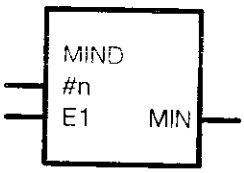
FBD/LD



IL

```
!BA     0
MIND
        #2
        MD 05,00
        MD 05,01
        MD 09,04
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
E1	E	D	N	P	N	1	0
MIN	A	D	N	P	Y	0	0

CE IL Definition

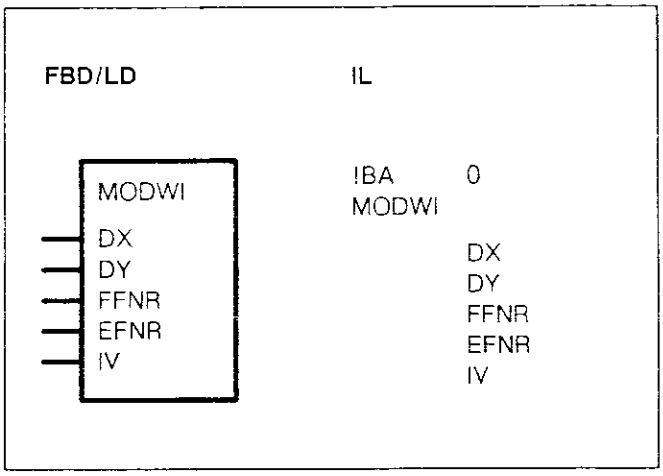
00000	!BA	0	Nr	Block No. (preset to 0)
00001	MIND			
00002		PP 0	#n	# CONSTANT (number of inputs)
	[	1		
00003		PP 1	E1	Input DOUBLE WORD
	]	1		
00004		PP 0	MIN	Output DOUBLE WORD (min. value)

MODIFY AND LOAD THE MEASURMENT WINDOW FRAMES (OMS-F)

MODWI

This function block modifies a single window or an entire window sequence which has been defined beforehand in "teach-in mode" or with the GENWI block.

This block is needed when using the video sensor OMS-F.



Parameters

DX	WORD	EW, AW, MW, KW	Number of pixels by which the window is moved in the x direction.
DY	WORD	EW, AW, MW, KW	Number of pixels by which the window is moved in the y direction.
FFNR	WORD	EW, AW, MW, KW	Window sequence number
EFNR	WORD	EW, AW, MW, KW	Single window number
IV	WORD	EW, AW, MW, KW	Number of the 35 IV 90 module

CE Data

Runtime:	
Basic runtime:	914 µs for *one* single window 1.36 ms for *all* single windows
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block modifies a single window or an entire window sequence which has been defined beforehand in "teach-in mode" or with the GENWI block. After modification, this single window or window sequence is called on the iconic processor 35 IV 90.

Therefore, in addition to the functional scope of the PROWI block, this block also has the capability of moving the windows defined in "teach-in mode" before loading to the 35 IV 90 module. That is to say, original storage of the windows in the PLC is not affected by the move.

Using this block, it is very easy to adapt the position of measurement windows to the real position of the measured object. Measured objects transported on a conveyor belt, for example, generally do not have the same position. The deviation from the nominal position is defined by a premeasurement and the result is forwarded to the block's inputs DX and DY.

The original coordinates of the window sequence or the single window have been determined beforehand by means of the PLC command X-WINDOW (XW) in "teach-in mode" or with the GENWI block and stored in the PLC.

Measurement then takes place on the iconic processor to which the modified window coordinates are loaded. The number of this 35 IV 90 module need not be the same one as the one specified in "teach-in mode".

On the 35 IV 90 module, each measurement window is assigned a pixel counter which is activated during measurement. The pixel counter has the same designation as the affiliated window.

DX WORD
Number of pixels (columns) for moving the window frame in the X direction.

The number of pixels (columns) by which the measurement window is to be moved in the X direction is specified with the operand at this input.

Move:
To the RIGHT -> specify positive value
To the LEFT -> specify negative value

DY WORD
Number of pixels (lines) for moving the window frame in the Y direction.

The number of pixels (lines) by which the measurement window is to be moved in the Y direction is specified with the operand at this input.

Move:
DOWN -> specify a positive value
UP -> specify a negative value

FFNR WORD
Window sequence number.

The number of the required window sequence is specified with the operand at this input.
Value range: 1 ... 512.
A window sequence consists of a maximum of 8 single windows.

EFNR WORD
Single window number.

The number of the required single window is specified with the operand at this input. However, all windows of the specified window sequence can also be selected.

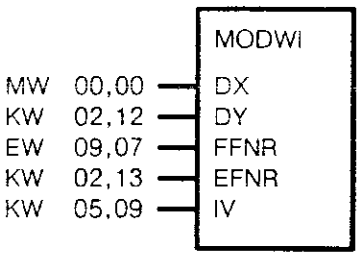
Value range: 0 ... 8.
0: All windows in the window sequence are selected.
1...8: Number of the window that is selected.

IV WORD
Number of the 35 IV 90 module.

The number of the required 35 IV 90 module with which measurement is to take place is specified with the operand at this input.
Value range: 0...5.

Example

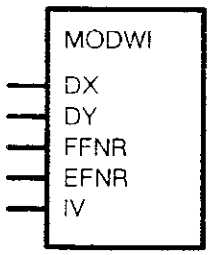
FBD/LD



IL

```
!BA      0
MODWI
         MW   00,00
         KW   02,12
         EW   09,07
         KW   02,13
         KW   05,09
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
DX	E	W	N	P	Y	0	0
DY	E	W	N	P	Y	0	0
FFNR	E	W	N	P	Y	0	0
EFNR	E	W	N	P	Y	0	0
IV	E	W	N	P	Y	0	0

CE IL Definition

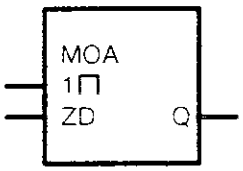
00000	!BA	0	Nr	Block No. (preset to 0)
00001	MODWI			
00002		PP 0	DX	Input WORD (DX shift)
00003		PP 0	DY	Input WORD (DY shift)
00004		PP 0	FFNR	Input WORD (window sequence number)
00005		PP 0	EFNR	Input WORD (single window number)
00006		PP 0	IV	Input WORD (IV No.)

A 0–1 edge at the input 1⊓ produces a 0–1 edge at the output Q. If the input 1⊓ remains at 1 level, a 1–0 edge is output on output Q after duration ZD has elapsed.

The output Q is also set back to 0 level if the input 1⊓ should return to 0 level before expiry of time TD.

Maximum time offset at the output: < 1 cycle time

FBD/LD



IL

```
!BA   0
MOA
      1⊓
      ZD
      Q
```

Parameters

1⊓	BINARY	E, M, A, S, K	Input signal
ZD	DOUBLE WORD	MD, KD	Pulse length
Q	BINARY	M, A	Output signal

CE Data

Runtime:	
Basic runtime:	36 µs; max. runtime in case of edges at input 1⊓: 600 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

A 0–1 edge at the input 1⊓ produces a 0–1 edge at the output Q. If the input 1⊓ remains at 1 level, a 1–0 edge is output on output Q after duration ZD has elapsed.

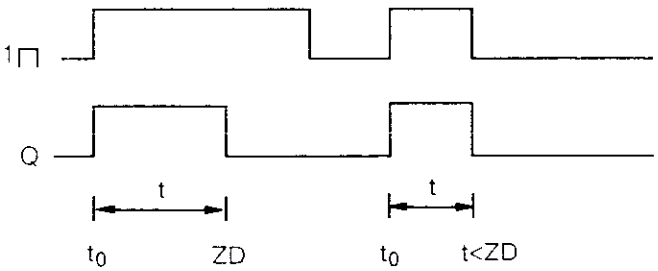
The output Q is also set back to 0 level if the input 1⊓ should return to 0 level before expiry of time TD.

The time is specified in milliseconds. Only integral multiple of 5 ms are admissible (Expl.: 5 ms, 500 ms, 100 000 ms, ...). Time range which can be specified: 5 ms ... 24.8 days.

Maximum time offset at the output: < 1 cycle time

Meaningful range for ZD: > 1 cycle time

The inputs and the output can neither be duplicated nor inverted.



General behavior

- Started timers are processed by the PLC's operating system and are therefore completely independent of processing of the PLC program. A corresponding message is not issued by the operating system to the affiliated timer in the PLC program until the timer has elapsed.
- Processing of a timer in the operating system of the PLC is *not* influenced by the following commands:
  - Abort program
  - Start program
  - Stop program
  - Continue program

  That is to say, if a timer has been started it will continue to be processed in the PLC's operating system even if the affiliated PLC program is aborted, restarted or stopped and continued again.

Initialization

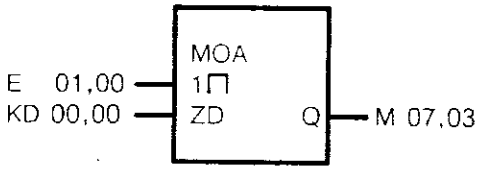
The timers are always initialized in the event of a cold or warm start of the PLC. Therefore, a running timer is always aborted by a cold or warm start of the PLC.

Cold start:
- KALT <CR> command
- Activation of the power for the first time

Warm start:
- WARM <CR> command
- Activation of the power
- RESET switch

Example

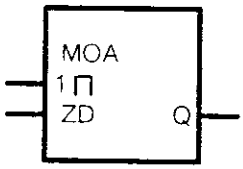
FBD/LD



IL

```
!BA   0
MOA
      E  01,00
      KD 00,00
      M  07,03
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
1⊓	E	L	N	P	Y	0	0
ZD	E	D	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

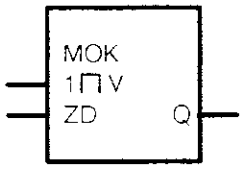
00000	!BA	0	Nr	Block No. (preset to 0)
00001	MOA			
00002		PP 0	1⊓	Input BINARY (start timer)
00003		PP 0	ZD	Input DOUBLE WORD (pulse length)
00004		PP 0	Q	Output BINARY (time)

A 0-1 edge at the input 1⊓ produces a 0-1 edge at the output Q. The output Q is reset to 0 level after expiry of the period ZD.

A second 0-1 edge of the input 1⊓ V before the time ZD has elapsed is ignored.

Maximum time offset at the output: < 1 cycle time

FBD/LD



IL

```
!BA   0
MOK
      1⊓V
      ZD
      Q
```

Parameters

1⊓ V	BINARY	E, M, A, S, K	Input signal
ZD	DOUBLE WORD	MD, KD	Pulse length
Q	BINARY	M, A	Output signal

CE Data

Runtime:	
Basic runtime:	20 µs; max. runtime 380 µs in case of edges at input 1⊓ V
Additional runtime:	---
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

A 0-1 edge at the input 1⊓ produces a 0-1 edge at the output Q. The output Q is reset to 0 level after expiry of the period ZD.

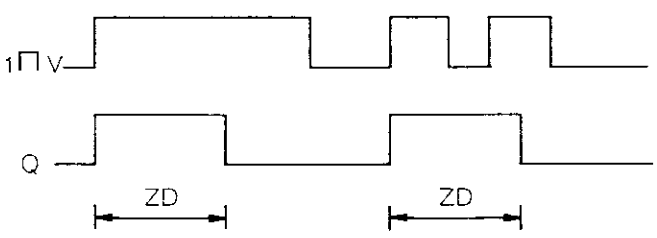
A second 0-1 edge of the input 1⊓ V before the time ZD has elapsed is ignored.

The time is specified in milliseconds. Only integral multiple of 5 ms are admissible (Expl.: 5 ms, 500 ms, 100 000 ms, ...). Time range which can be specified: 5 ms ... 24.8 days.

Maximum time offset at the output: < 1 cycle time

Meaningful range for ZD: > 1 cycle time

The inputs and the output can neither be duplicated nor inverted.



General behavior

- Started timers are processed by the PLC's operating system and are therefore completely independent of processing of the PLC program. A corresponding message is not issued by the operating system to the affiliated timer in the PLC program until the timer has elapsed.
- Processing of a timer in the operating system of the PLC is *not* influenced by the following commands:
  - Abort program
  - Start program
  - Stop program
  - Continue program

  That is to say, if a timer has been started it will continue to be processed in the PLC's operating system even if the affiliated PLC program is aborted, restarted or stopped and continued again.

Initialization

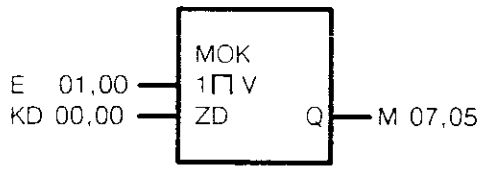
The timers are always initialized in the event of a cold or warm start of the PLC. Therefore, a running timer is always aborted by a cold or warm start of the PLC.

Cold start:
- KALT <CR> command
- Activation of the power for the first time

Warm start:
- WARM <CR> command
- Activation of the power
- RESET switch

Example

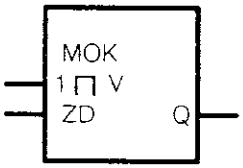
FBD/LD



IL

```
!BA   0
MOK
      E   01,00
      KD  00,00
      M   07,05
```

CE FBD Definition



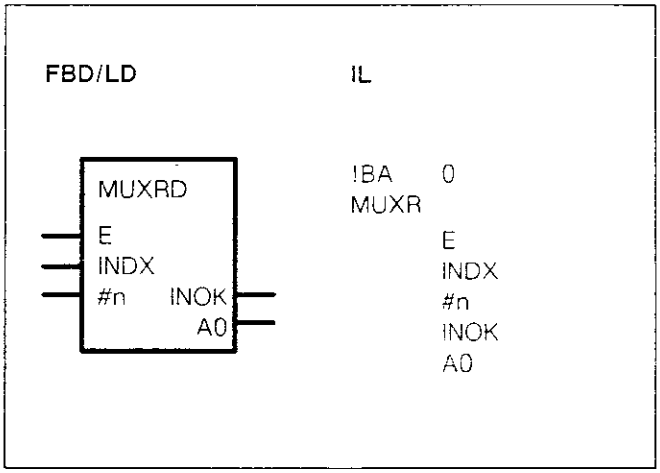
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
1⊓ V	E	L	N	P	Y	0	0
ZD	E	D	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MOK			
00002		PP 0	1⊓ V	Input BINARY (start time)
00003		PP 0	ZD	Input DOUBLE WORD (time value)
00004		PP 0	Q	Output BINARY (time)

MULTIPLEXER, DOUBLE WORD WITH RESET — MUXRD

This function block connects the input E to one of the outputs A0...A_{n-1} depending on the input INDX.



Parameters

E	DOUBLE WORD	MD, KD	Input
INDX	WORD	EW, AW, MW, KW	Index input
#n	DIRECT CONSTANT	#,#H	Quantity n of double word outputs A0 ... An-1
INOK	BINARY	A, M	Range monitoring for input INDX
A0	WORD	MD	Double word outputs A0 ... An-1; the output A0 is capable of duplication

CE Data

Runtime:	
Basic runtime:	43 µs
Additional runtime:	10 µs per additional double word output A1 ... An-1
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block connects the input E to one of the outputs A0...An-1 depending on the input INDX.

The double word outputs that are not connected are set to 0. The validity of the value at the input INDX is checked.

Relationship between E, INDX and A0...An-1:

The input INDX is used to define with which of the outputs A0...A_{n-1} the input E is connected.

The following applies:

INDX = 1 : E -> A0
INDX = 2 : E -> A1
INDX = 3 : E -> A2
: : :
INDX = n : E -> An-1

where: 1 ≤ INDX ≤ n ≤ 32767 (theoretically)

E DOUBLE WORD

Input which is switched through to one of the outputs A0...An-1.

INDX WORD

Index input for selection of one of the outputs A0...An-1.
Value range: 1 ≤ INDX ≤ n

Note:
INDX = 0 can be used to initialize the outputs A0...A_{n-1} (A0...An-1 = 0).

#n DIRECT CONSTANT

Quantity n of double word outputs A0...An-1.

INOK BINARY

Range monitoring of the input INDX

The output indicates whether or not the input INDX is within the valid range.
Valid range: $1 \leq INDX \leq n$

INOK = 1 -> Index input INDX within the valid range
INOK = 0 -> Index input INDX in the invalid range
-> A0...An-1 = 0

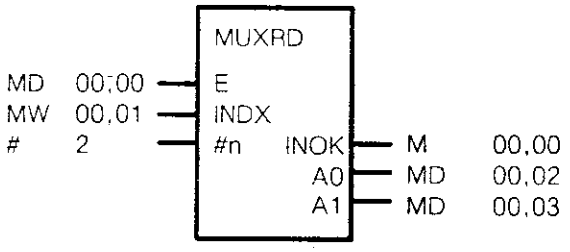
If the word input INDX is not within the valid range, all double word outputs A0 to A_{n-1} are set to 0. Therefore, for example, INDX = 0 can be used to initialize the outputs ($A0...A_{n-1} = 0$).

A0 DOUBLE WORD

The output A0 is capable of duplication ($A0...A_{n-1}$). The input E is allocated to one of the n outputs $A0...A_{n-1}$.

Example

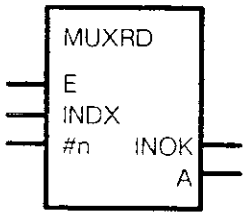
FBD/LD



IL

```
!BA   0
MUXR
      MD   00,00
      MW   00,01
      #    2
      M    00,00
      MD   00,02
      MD   00,03
```

CE FBD Definition

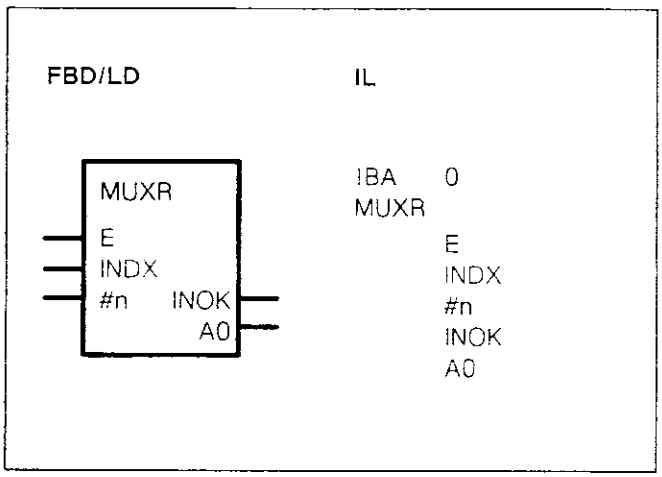


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	D	N	P	Y	0	0
INDX	E	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
INOK	A	L	N	P	Y	0	0
A	A	D	N	P	Y	1	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MUXRD			
00002		PP 0	E	Input (DOUBLE WORD)
00003		PP 0	INDX	Index (WORD)
00004		PP 0	#n	Number of outputs (#)
00005		PP 0	INOK	Output index valid (BINARY)
	[	1		
00006		PP 1	A	Output (DOUBLE WORD)
	]	1		

This function block connects the input E to one of the outputs A0...An-1 depending on the input INDX.



Parameters

E	WORD	EW, AW, MW, KW	Input
INDX	WORD	EW, AW, MW, KW	Index input
#n	DIRECT CONSTANT	#,#H	Quantity n of word outputs A0 ... An-1
INOK	BINARY	A, M	Range monitoring for input INDX
A0	WORD	AW, MW	Word outputs A0 ... An-1; the output A0 is capable of duplication

CE Data

Runtime:	
Basic runtime:	44.5 µs
Additional runtime:	7.5 µs per additional word output A1 ... An-1
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block connects the input E to one of the outputs A0...An-1 depending on the input INDX.

The word outputs that are not connected are set to 0.
The validity of the value at the input INDX is checked.

Relationship between E, INDX and A0...An-1:

The input INDX is used to define with which of the outputs A0...A_{n-1} the input E is connected.
The following apply:

INDX = 1 : E -> A0
INDX = 2 : E -> A1
INDX = 3 : E -> A2
: : :
INDX = n : E -> An-1

where 1 ≤ INDX ≤ n ≤ 32767 (theoretically)

E WORD

Input which is switched through to one of the outputs A0...An-1.

INDX WORD

Index input for selection of one of the outputs A0...An-1.
Value range: 1 ≤ INDX ≤ n

Note:
INDX = 0 can be used to initialize the outputs A0 ... A_{n-1} (A0 ... An-1 = 0).

#n DIRECT CONSTANT

Quantity n of word outputs A0 ... A_{n-1}.

INOK BINARY

Range monitoring of the INDX input

The output indicates whether or not the input INDX is within the valid range.
Valid range: $1 \leq INDX \leq n$

INOK = 1 -> Index input INDX within the valid range
INOK = 0 -> Index input INDX in the invalid range
-> $A0...A_{n-1} = 0$

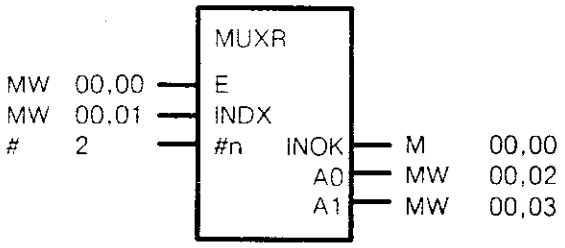
If the word input INDX is not within the valid range, all word outputs A0 to A_{n-1} are set to 0. Thus, for example, INDX = 0 can be used to initialize the outputs ($A0...A_{n-1} = 0$).

A0 WORD

The output A0 is capable of duplication ($A0...A_{n-1}$). The input E is allocated to one of the n outputs $A0...A_{n-1}$.

Example

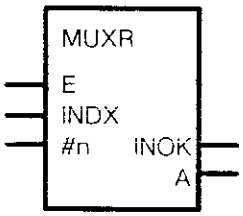
FBD/LD



IL

```
!BA    0
MUXR
       MW   00,00
       MW   00,01
       #    2
       M    00,00
       MW   00,02
       MW   00,03
```

CE FBD Definition



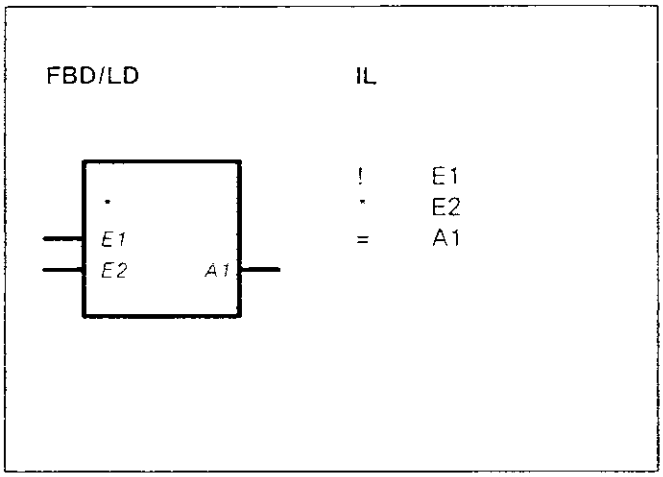
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	W	N	P	Y	0	0
INDX	E	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
INOK	A	L	N	P	Y	0	0
A	A	W	N	P	Y	1	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MUXR			
00002		PP 0	E	Input (WORD)
00003		PP 0	INDX	Index (WORD)
00004		PP 0	#n	Number of outputs (#)
00005		PP 0	INOK	Output index valid (BINARY)
	[	1		
00006		PP 1	A	Output (WORD)
	]	1		

MULTIPLICATION *

The values of the operands at the inputs of this connection element are multiplied by each other and the result is allocated to the operand at the output.



Parameters

E1	WORD	EW, MW, AW, KW	Multiplicand
E2	WORD	EW, MW, AW, KW	Multiplier; The input is capable of duplication
A1	WORD	MW, AW	Result (Product)

CE Data

Runtime:	
Basic runtime:	< 30 µs
Additional runtime:	23 µs per additional input (E3 ... En)
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The values of the operands at the inputs of this connection element are multiplied by each other and the result is allocated to the operand at the output.

Input E2 is capable of duplication (E2...En). All inputs and the output are capable of negation.

Number range

Integer word (16 Bit)

The following specially applies here to the *non-negated* inputs E1 and E2:
Low limit: 8000 H (–32768)

The following generally applies:

- Low limit: 8001 H –32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (–32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

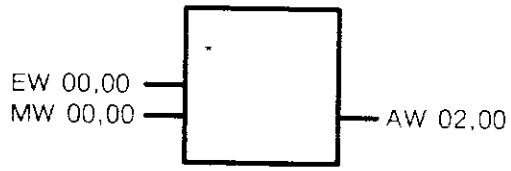
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the forbidden value 8000H (–32768) is corrected to the allowed value 8001H (–32767).

Example

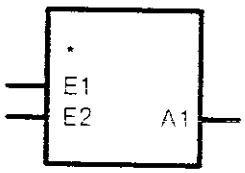
FBD/LD



IL

```
!    EW 00,00
*    MW 00,00
=    AW 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	Y	P	N	0	0
E2	E	W	Y	P	N	1	0
A1	A	W	Y	P	N	0	0

CE IL Definition

00000	!	PP 0	E1	Input WORD
	[	1		
00002	*	PP 1	E2	Input WORD (capable of duplication)
	]	1		
00005	=	PP 0	A1	Output WORD

MULTIPLICATION BY 2 TO THE POWER OF N — MUL2N

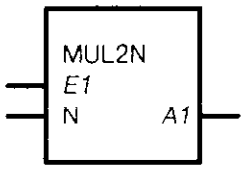
The value of the operand at the input E1 is shifted bit-by-bit N times.

If the value at the input N is positive, the value is shifted to the left. For each shift by 1 bit position, this corresponds to multiplication of the current value by 2.

If the value at the input N is negative, the value is shifted to the right. For each shift by 1 bit position, this corresponds to division of the current value by 2.

The result is allocated to the operand at the output A1.

FBD/LD



IL

```
!BA   0
MUL2N
      E1
      N
      A1
```

Parameters

E1	WORD	EW, MW, AW, KW	Input operand
N	WORD	EW, MW, AW, KW	Quantity
A1	WORD	MW, AW	Result

CE Data

Runtime:	
Basic runtime:	30–35 µs
Additional runtime:	4 µs per shift
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, 801 / 35 ZE 93 R101

Description

The value of the operand at the input E1 is shifted bit-by-bit N times.

If the value at the input N is positive, the value is shifted to the left. For each shift by 1 bit position, this corresponds to multiplication of the current value by 2.

If the value at the input N is negative, it is shifted to the right. For each shift by 1 bit position, this corresponds to division of the current value by 2.

The result is allocated to the operand at the output A1.

The inputs and the output can neither be duplicated nor negated.

Meaningful range for N: $-14 \leq N \leq +14$

If N = 0, the value at the input E1 is passed directly to the output A1.

Sign of the value at the input E1:

The sign of the value E1 is not influenced by the shift operation. That is to say, the sign of the output value is always identical with the input value's sign.

Shift to the left (Multiplication):

When the value at the input is shifted to the left, the released bit 0 is filled with 0. The sign bit (bit 15) is not changed because limiting to the limit of the number range takes place beforehand.

Limiting of the value at the output A1 when shifting to the left:

- The following applies to positive values at the input E1:

If bit 14 has a "1" and if shift operations still have to be carried out on the basis of the value at the input N, these are no longer executed. Instead, the output is set to the limit of the positive number range. That is to say, the limit has been reached in any case at the latest after 14 shifts.

Limit value: Output A1 = +32767 (7FFF H).

- The following applies to negative values at the input E1:

If bit 14 has a "0" and shift operations still have to be carried out because of the value at the input N, these will no longer be executed. Instead, the output is set to the limit of the negative number range. That is to say, the limit has been reached in any case at the latest after 14 shifts.

Limit: Output A1 = -32767 (8001 H)

Shift to the right (Division):

When shifting to the right, every bit moves to the right by one position. At the same time, the sign bit (bit 15) always retains its value. The released bit (bit 14) is filled in each case with the value of the sign bit.

Limiting of the value at the output when shifting to the right:

- The following applies to positive values at the input E1:

If now only bit 0 has a "1" and shift operations still have to be carried out because of the value at the input N, the output will be set to the value 0. That is to say, the value 0 has been reached in any case at the latest after 14 shifts.
Output A1 = 0.

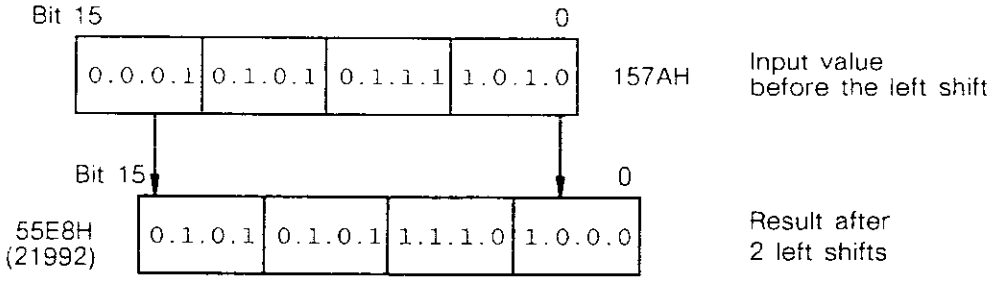
- The following applies to negative values at the input E1:

If bit 0 ... bit 15 has a "1" as the result of the shift, the limit value (-1) has been reached. Further shifts have no effect. That is to say, the value -1 has been reached at the latest after 15 shifts.
Output A1 = -1 (FFFF H).

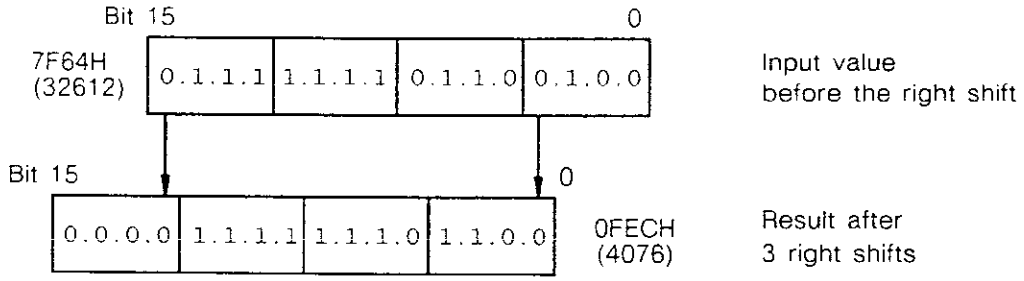
The inputs and the output can neither be duplicated nor negated.

Examples:

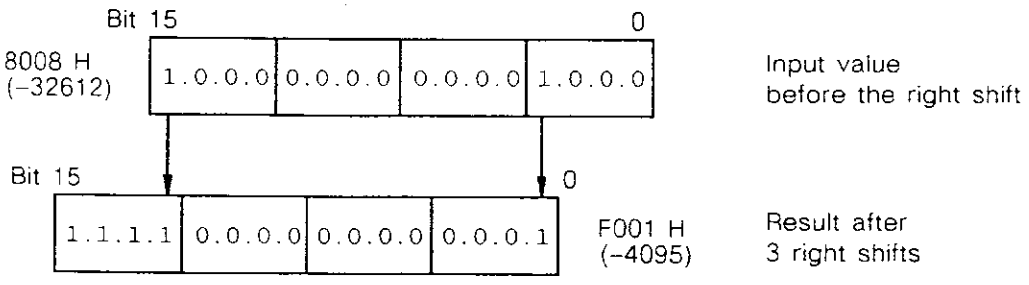
1. Input value E1 = 5498 (157AH)
   Exponent N = 2 -> 2 * Left shift



2. Input value E1 = 32612 (7F64H)
   Exponent N = -3 -> 3 * Right shift

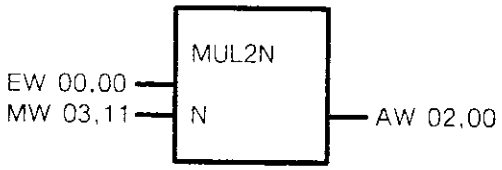


3. Input value E1 = -32612 (8008 H)
   Exponent N = -3 -> 3 * Right shift



Example

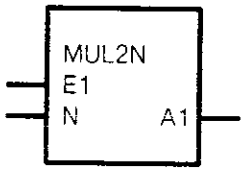
FBD/LD



IL

```
!BA    0
MUL2N
       EW 00,00
       MW 03,11
       AW 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
N	E	W	N	P	Y	0	0
A1	A	W	N	P	N	0	0

CE IL Definition

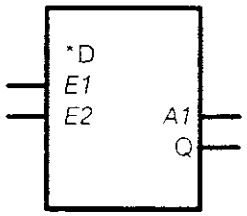
00000	!BA	0	Nr	Block No. (preset to 0)
00001	MUL2N			
00002		PP 0	E1	Input WORD
00003		PP 0	N	Input WORD (number of shifts)
00004		PP 0	A1	Output WORD

MULTIPLICATION, DOUBLE WORD *D

The value of the operand at the input E1 is multiplied by the value of the operand at the input E2 and the result is allocated to the operand at the output A1.

The result is limited to the maximum or minimum value of the number range. If limiting has taken place, a 1 signal is allocated to the binary operand at the output Q. If no limiting has taken place, a 0 signal is allocated to the binary operand at the output Q.

FBD/LD



IL

```
!BA   0
MULD
      E1
      E2
      A1
      Q
```

Parameters

E1	DOUBLE WORD	MD, KD	Multiplicand
E2	DOUBLE WORD	MD, KD	Multiplier
A1	DOUBLE WORD	MD	Result (Product)
Q	BINARY	A,M	Result limited

CE Data

Runtime:	
Basic runtime:	117 ... 120 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the operand at the input E1 is multiplied by the value of the operand at the input E2 and the result is allocated to the operand at the output A1.

The result is limited to the maximum or minimum value of the number range. If limiting has taken place, a 1 signal is allocated to the binary operand at the output Q. If no limiting has taken place, a 0 signal is allocated to the binary operand at the output Q.

The inputs and outputs can neither be duplicated nor negated.

Number range

Integer double word (32 Bits)

The following specially applies here to inputs E1 and E2:

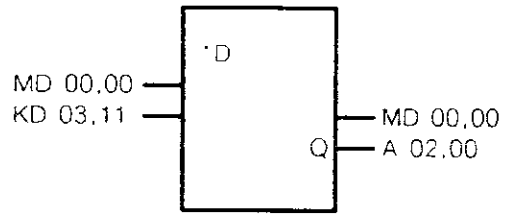
- Low limit: 8000 0000 H -2 147 483 648

The following generally applies:

- Low limit: 8000 0001 H -2 147 483 647
- High limit: 7FFF FFFF H +2 147 483 647
- Inadmissible value: 8000 0000 H ---

Example

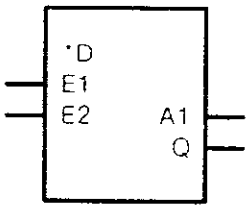
FBD/LD



IL

```
!BA    0
MULD
       MD 00,00
       KD 03,11
       MD 00,00
       A  02,00
```

CE FBD Definition



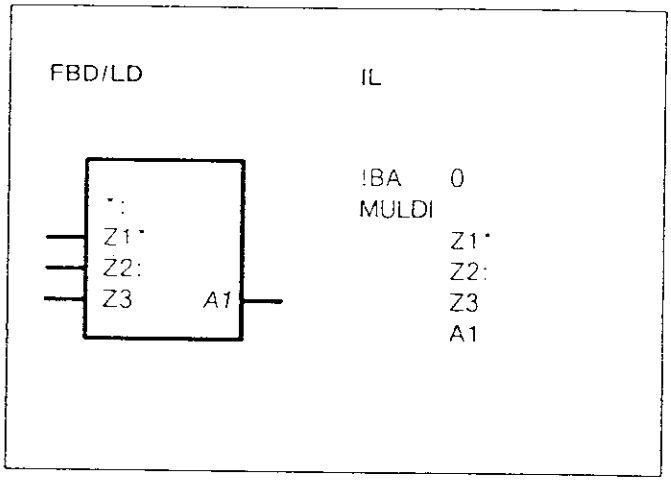
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
E2	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MULD			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	E2	Input DOUBLE WORD
00004		PP 0	A1	Output DOUBLE WORD
00005		PP 0	Q	Output BINARY

MULTIPLICATION WITH DIVISION *:

The value of the operand at the input Z1 is multiplied by the value of the operand at the input Z2, the intermediate result is divided by the value of the operand at the input Z3 and then the result is allocated to the operand at the output A1.



Parameters

Z1*	WORD	EW, MW, AW, KW	Multiplicand
Z2:	WORD	EW, MW, AW, KW	Multiplier
Z3	WORD	EW, MW, AW, KW	Divisor
A1	WORD	MW,AW	Result

CE Data

Runtime:	
Basic runtime:	84 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The value of the operand at the input Z1 is multiplied by the value of the operand at the input Z2, the intermediate result is divided by the value of the operand at the input Z3 and then the result is allocated to the operand at the output A1.

Internally, this function block operates with double word accuracy (32 bits) when multiplying and dividing. Only when allocating the result to the output A1 is the value limited to word accuracy (16 bits). The result is rounded up if the remainder of division is ≥ 0.5. If a number overflow occurs during division (e.g. division by 0), the limit value of the number range is allocated to the output A1 with the correct sign.

The inputs and the output can neither be duplicated nor negated.

Number range

Integer word (16 Bits)

• Low limit:	8001 H	-32767
• High limit:	7FFF H	+32767
• Inadmissible value:	8000 H	---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

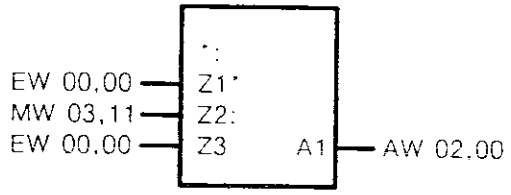
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

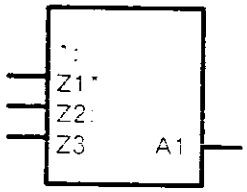
FBD/LD



IL

```
!BA     0
MULDI
        EW 00,00
        MW 03,11
        EW 00,00
        AW 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1*	E	W	N	P	Y	0	0
Z2:	E	W	N	P	Y	0	0
Z3	E	W	N	P	Y	0	0
A1	A	W	N	P	N	0	0

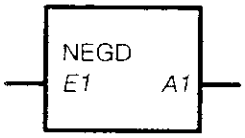
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	MULDI			
00002		PP 0	Z1*	Input WORD
00003		PP 0	Z2:	Input WORD
00004		PP 0	Z3	Input WORD
00005		PP 0	A1	Output WORD

NEGATION, DOUBLE WORD — NEGD

The value of the operand at the input E1 is negated and the result is allocated to the operand at the output A1.

FBD/LD



IL

```
!BA   0
NEGD
      E1
      A1
```

Parameters

E1	DOUBLE WORD	MD, KD	Input value
A1	DOUBLE WORD	MD	Negated value

CE Data

Runtime:	
Basic runtime:	34 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the operand at the input E1 is negated and the result is allocated to the operand at the output A1.

If the inadmissible value 8000 0000 H (-2 147 483 648) is present at the input E1, the value 7FFF FFFF H (+2 147 483 647) is allocated to the output A1. Therefore, before negation the inadmissible value is replaced by the admissible value 8000 0001 H (-2 147 483 647).

The input and the output can neither be duplicated nor negated.

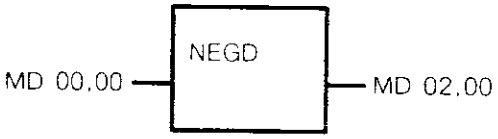
Number range

Integer double word (32 Bit).

- Low limit: 8000 0001 H -2 147 483 647
- High limit: 7FFF FFFF H +2 147 483 647
- Inadmissible value: 8000 0000 H ---

Example

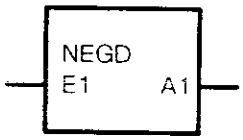
FBD/LD



IL

```
!BA    0
NEGD
       MD 00.00
       MD 02.00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	NEGD			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	A1	Output DOUBLE WORD

NOTBIT is a basic function in the manufacturer library. The name of the element in the library is NOTBIT. This element cannot be called in the CE and CE-IL editors.

FBD/LD	IL
Calll not possible	

Parameters

CE Data

Runtime:	
Basic runtime:	4 µs
Additional runtime:	---
Output updating:	---
Number of historical values:	---
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

NOTBIT is a basic function in the manufacturer library. The name of the element in the library is NOTBIT. This element cannot be called in the CE and CE-IL editors. The programming unit uses this element to realize connecting lines.

Example

FBD/LD	IL
Call not possible	

CE FBD Definition

not defined

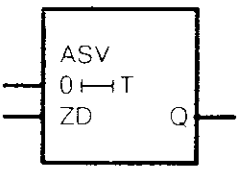
CE IL Definition

```
00000    !     PP 0    inp
00002    =N    PP 0    outp
```

The 1-0 edge of the input 0⊢⊣T is delayed by the time ZD and is output as a 1-0 edge at the output Q.

If the input 0⊢⊣T returns to 1 level before expiry of the time ZD, the output Q retains 1 level.

FBD/LD



IL

```
!BA   0
ASV
      0⊢⊣T
      ZD
      Q
```

Parameters

0⊢⊣T	BINARY	E, M, A, S, K	Input signal
ZD	DOUBLE WORD	MD, KD	Delay time
Q	BINARY	M, A	Delayed signal

CE Data

Runtime:	
Basic runtime:	45 µs; max. runtime in case of edges at input 0⊢⊣T: 541 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The 1-0 edge of the input 0⊢⊣T is delayed by the time ZD and is output as a 1-0 edge at the output Q.

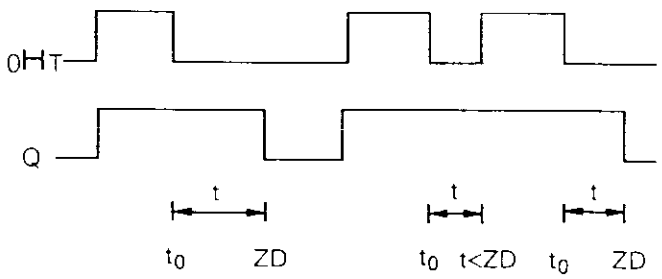
If the input 0⊢⊣T returns to 1 level before expiry of the time ZD, the output Q retains 1 level.

The time is specified in milliseconds. Only integral multiple of 5 ms are admissible (Expl.: 5 ms, 500 ms, 100 000 ms, ...). Time range which can be specified: 5 ms ... 24.8 days.

Maximum time offset at the output: < 1 cycle time

Meaningful range for ZD: > 1 cycle time.

The inputs and the output can neither be duplicated nor inverted.



General behavior

- Started timers are processed by the PLC's operating system and are therefore completely independent of PLC program processing. The operating system does not issue a corresponding message to the affiliated timer block in the PLC program until the timer has elapsed.
- Processing of a timer in the PLC's operating system is *not* influenced by the following commands:
  - Abort program
  - Start program
  - Stop program
  - Continue program

  That is to say, if a timer has been started, it continues to be processed in the PLC's operating system if the affiliated PLC program is aborted, restarted or stopped and then continued again.

Initialization

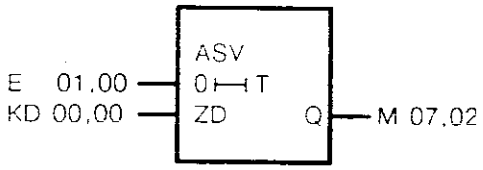
The timers are always initiated each time a PLC cold or warm start is executed. Therefore, any running timer is always aborted by a cold or warm start of the PLC.

Cold start:
- KALT <CR> command
- When activating the voltage for the first time

Warm start:
- WARM <CR> command
- Activating the voltage
- RESET switch

EXAMPLE

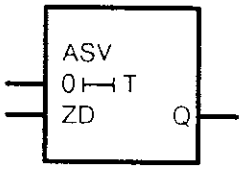
FBD/LD



IL

```
!BA   0
ASV
      E  01.00
      KD 00.00
      M  07.02
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0⊢T	E	L	N	P	Y	0	0
ZD	E	D	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

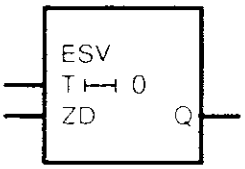
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	ASV			
00002		PP 0	0 ⊢ T	Input BINARY (start time)
00003		PP 0	ZD	Input DOUBLE WORD (time value)
00004		PP 0	Q	Output BINARY (time)

The 0-1 edge of the input T⊢⊣0 is delayed by the time ZD and is output as a 0-1 edge at the output Q.

The output Q retains 0 level if the input T⊢⊣0 returns to 0 level before the time ZD has elapsed.

FBD/LD



IL

```
!BA   0
ESV
      T ⊢⊣ 0
      ZD
      Q
```

Parameters

T ⊢⊣ 0	BINARY	E, M, A, S, K	Input signal
ZD	DOUBLE WORD	MD, KD	Delay time
Q	BINARY	M, A	Delayed signal

CE Data

Runtime:	
Basic runtime:	38 µs; max. runtime 400 µs, in case of edges at input T⊢⊣0
Additional runtime:	---
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The 0-1 edge of the input T⊢⊣0 is delayed by the time ZD and is output as a 0-1 edge at the output Q.

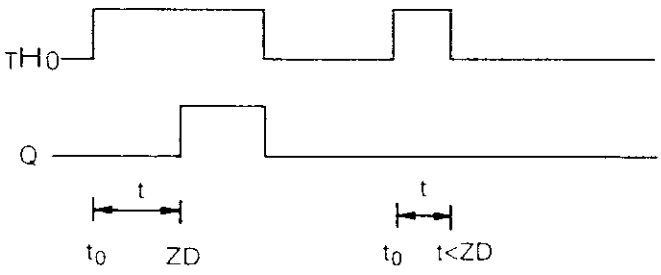
The output Q retains 0 level if the input T⊢⊣0 returns to 0 level before the time ZD has elapsed.

The time is specified in milliseconds. Only integral multiple of 5 ms are admissible (Expl.: 5 ms, 500 ms, 100 000 ms, ...). Time range which can be specified: 5 ms ... 24.8 days.

Maximum time offset at the output: < 1 cycle time

Meaningful range for ZD: > 1 cycle time

The inputs and the output can neither be duplicated nor inverted.



General response

- Started timers are processed by the PLC's operating system and are therefore completely independent of processing of the PLC program. An appropriate message of the operating system is not issued to the affiliated timer block in the PLC program until the timer has elapsed.
- Processing of a timer in the PLC's operating system is *not* influenced by the following commands:
  - Abort program
  - Start program
  - Stop program
  - Continue program

  That is to say, processing of a started timer is continued in the PLC's operating system even if the affiliated PLC program is aborted, restarted or stopped and continued again.

Initialization

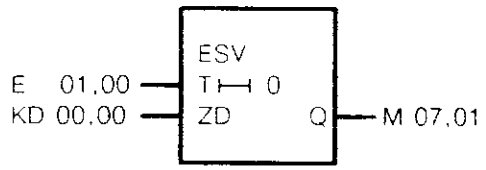
The timers are always initialized each time a cold or warm start of the PLC is executed. A running timer will always be aborted by a cold or warm start of the PLC.

Cold start:
- KALT <CR> command
- Activating the voltage for the first time

Warm start:
- WARM <CR> command
- Activating the voltage
- RESET switch

Example

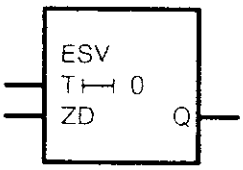
FBD/LD



IL

```
!BA   0
ESV
      E  01.00
      KD 00.00
      M  07.01
```

CE FBD Definition



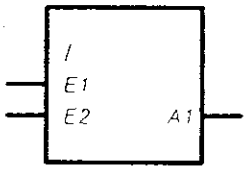
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
T⊢⊣ 0	E	L	N	P	Y	0	0
ZD	E	D	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	ESV			
00002		PP 0	T⊢⊣ 0	Input BINARY (start time)
00003		PP 0	ZD	Input DOUBLE WORD (time value)
00004		PP 0	Q	Output BINARY (time)

This connection element realizes a logical OR combination of the operands at the inputs. The result is allocated to the operand at the output.

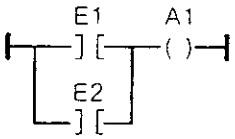
FBD



IL

```
!   E1
/   E2
=   A1
```

KOP



Parameters

E1	BINARY	E, M, A, S, K	Operand 1 of the OR combination
E2	BINARY	E, M, A, S, K	Operand 2 of the OR combination, capable of duplication
A1	BINARY	M, A, S	Result of the OR combination

CE Data

Runtime:	
Basic runtime:	6.6 µs
Additional runtime:	2.3 µs per additional input
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This connection element realizes a logical OR combination of the operands at the inputs. The result is allocated to the operand at the output.

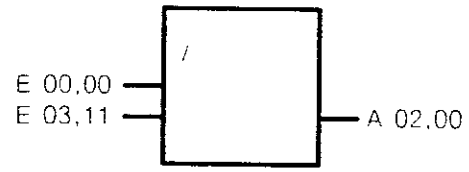
The input E2 is capable of duplication. All inputs and the output can be inverted.

Truth table:

E1	E2	A1
0	0	0
1	0	1
0	1	1
1	1	1

Example

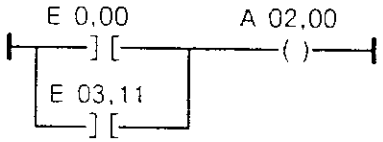
FBD



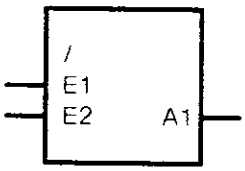
IL

```
!    E 00,00
/    E 03,11
=    A 02,00
```

KOP



CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	L	Y	P	N	0	0
E2	E	L	Y	P	N	1	0
A1	A	L	Y	P	N	0	0

CE IL Definition

00000	!	PP 0	E1	Input BINARY
	[	1		
00002	/	PP 1	E2	Input BINARY (capable of duplication)
	]	1		
00004	=	PP 0	A1	Output BINARY

This function block generates the bit-by-bit OR combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

FBD/LD

IL

```
!BA   0
DWOR
      E1
      E2
      A1
```

Parameters

E1	DOUBLE WORD	MD, KD	Operand 1
E2	DOUBLE WORD	MD, KD	Operand 2
A1	DOUBLE WORD	MD	Result of the OR combination

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block generates a bit-by-bit OR combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

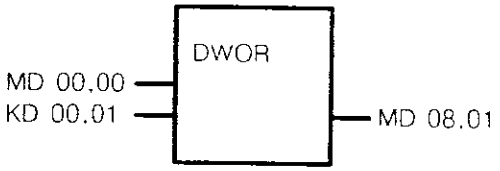
The inputs and the output can neither be duplicated nor negated nor inverted.

Example

E1	1.0.0.0	0.0.1.1	0.0.1.0	0.1.1.0	1.0.1.0	1.1.0.0	0.0.1.1	0.1.0.1
E2	1.0.0.1	0.1.1.0	0.0.1.0	1.1.1.1	1.1.1.1	0.0.0.0	0.1.1.0	1.1.0.0
A1	1.0.0.1	0.1.1.1	0.0.1.0	1.1.1.1	1.1.1.1	1.1.0.0	0.1.1.1	1.1.0.1

Example

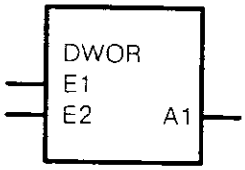
FBD/LD



IL

```
!BA    0
DWOR
       MD 00.00
       KD 00.01
       MD 08.01
```

CE FBD Definition



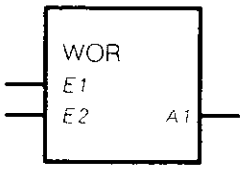
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
E2	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	DWOR			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	E2	Input DOUBLE WORD
00004		PP 0	A1	Output DOUBLE WORD

This function block generates the bit-by-bit OR combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

FBD/LD



IL

```
!BA   0
WOR
      E1
      E2
      A1
```

Parameters

E1	WORD	EW, MW, AW, KW	Operand 1
E2	WORD	EW, MW, AW, KW	Operand 2
A1	WORD	MW, AW	Result of the OR combination

CE Data

Runtime:

Basic runtime:	29 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block generates the bit-by-bit OR combination of the operands present at the inputs E1 and E2. The result is allocated to the operand at the output A1.

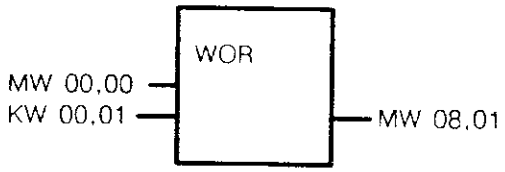
The inputs and the output can neither be duplicated nor negated nor inverted.

Example

E1	0.0.0.0	0.0.1.1	0.0.1.0	0.1.1.0
E2	1.0.0.1	0.0.0.0	0.0.1.0	1.1.1.1
A1	1.0.0.1	0.0.1.1	0.0.1.0	1.1.1.1

Example

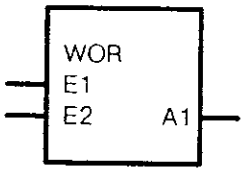
FBD/LD



IL

```
IBA   0
WOR
      MW 00.00
      KW 00.01
      MW 08.01
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
E2	E	W	N	P	N	0	0
A1	A	W	N	P	N	0	0

CE IL Definition

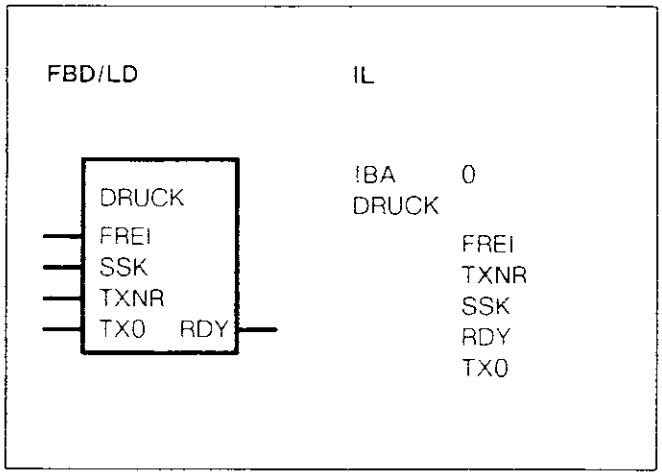
00000	IBA	0	Nr	Block No. (preset to 0)
00001	WOR			
00002		PP 0	E1	Input WORD
00003		PP 0	E2	Input WORD
00004		PP 0	A1	Output WORD

OUPUT OF ASCII CHARACTERS AND HEX VALUES THROUGH A SERIAL INTERFACE

DRUCK

The DRUCK function block outputs texts and values of operands stored in the PLC program through a serial interface.

The numeric values to be output are conditioned for a diversity of representations by specifying a format identifier.



Parameters

FREI	BINARY	A, E, S, M	Enable signal for output of *one* text (0–>1–edge)
SSK	WORD	AW, EW, MW, KW	Interface indentifier; Number of the serial interface
TXNR	WORD	AW, EW, MW, KW	Text number; number of the text to be output
TX0	ALL	Texte	Texts; input for the texts; input can be duplicated
RDY	BINARY	A, E, M	Ready; block is ready for output of a text

CE Data

Runtime:	
Basic runtime:	not available
Additional runtime:	not available
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V5 / 935 PC 83 R201 / 35 ZE 93 R101

Description

The DRUCK function block outputs texts and values of operands stored in the PLC program through a serial interface.

The numeric values to be output are conditioned for a diversity of representations by specifying a format identifier.

The texts and operand identifiers to be output are stored in the user program in the PLC directly after the DRUCK block.

The texts each have a text number for identification.

The DRUCK block writes texts and operand values to be output to a buffer from where they are fetched by an interface driver and output through the serial interface.

In doing so, the interface driver interprets the display formats for the operand values and converts them to the required display format before output.

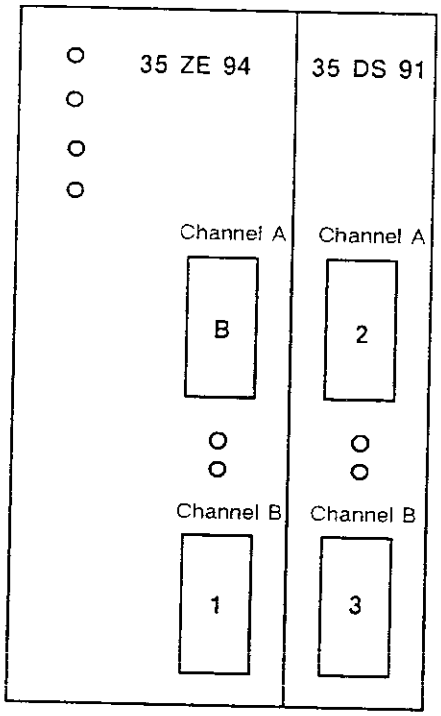
Initialization of the serial interface

Before the DRUCK block can communicate with one of the 3 possible serial interfaces, it must be initialized with the SINIT block.

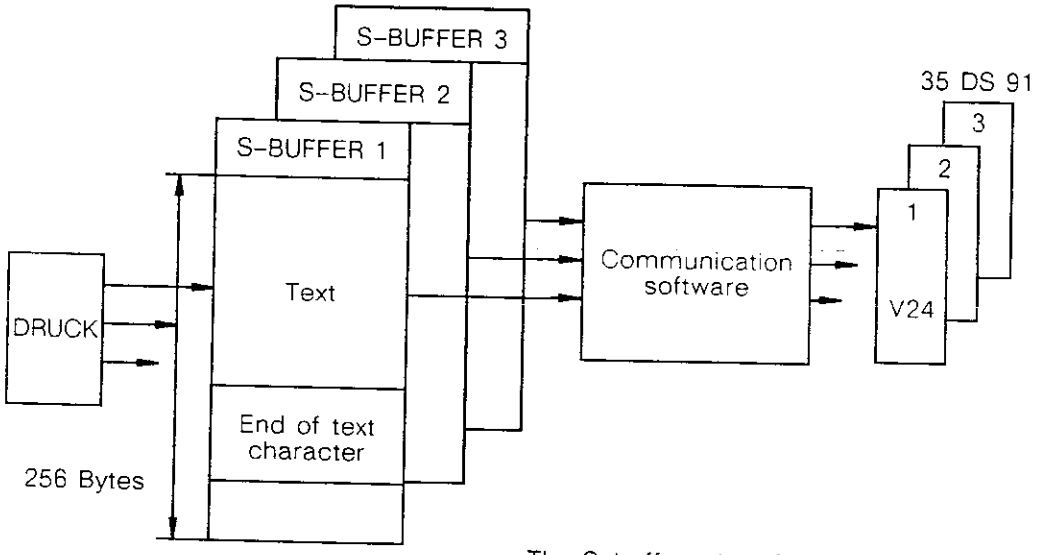
Apart from the required interface parameters, a freely definable end of text character is specified when using the SINIT block in order to mark the end of the overall text/telegram to be sent. Whether or not the end of text character is to be output through the interface is also specified in the SINIT block.

The data flow is synchronized principally by way of the RTS and CTS signals.

Interface of the PLC



PLC with serial interfaces 1 to 3



The S-buffers 1 to 3 are the send buffers which are assigned to the serial interfaces 1 to 3.

Brief overview of the block's parameters

FREI	RDY	TXNR	SSK	MEANING
0/1-EDGE	1	2	1	The text with the number 2 is output through interface 1
0/1-EDGE	0	2	1	0/1-edge is ignored because the block is not ready
no 0/1-EDGE	X	X	X	No output of a new text

FREI BINARY
A 0 -> 1 edge at the FREI input results in output of *one* text provided the block is ready to do this (RDY = 1).

If a 0 -> 1 edge appears at the FREI input although the RDY output is equal to 0, i.e. the block is not yet ready for a new transfer, the 0 -> 1 edge will be *ignored*. Therefore, no new text transfer can be started as long as the RDY signal is 0.

SSK WORD
The number of the interface through which the text is to be output is specified at the SSK input.
The following applies: 1 ≤ number ≤ 3.

TXNR WORD
The number of the text to be output is specified at the TXNR input.

1...99 texts are possible in a DRUCK block. The number of the text to be output must be present at the TXNR input until the block indicates the end of text transfer with a 1 signal at its RDY output.

RDY BINARY
In the program cycle in which the block is called for the first time, and during the time when a text is output, the following applies:
RDY = 0.

As long as RDY is equal to 0, no new text output can be activated and any 0 -> 1 edge present at the FREI input will be ignored and will be lost.

After the first call of the block or after termination of a text output, the following applies:

RDY = 1

The block is now ready for output of a new text.

The RDY signal can be used to activate a new text transfer with the FREI input, for example.

Storage of texts in the PLC

Storage: Directly after the DRUCK block

Quantity: 1...99

Length: Up to 256 characters (owing to the send buffer length = 256)

In this case, the following applies to the length of the texts:

- Each text character counts as 1 character
- Each format identifier counts as 3 characters
- Each bit operand counts as 1 character
- Each word operand counts as 2 characters
- Each double word operand counts as 4 characters

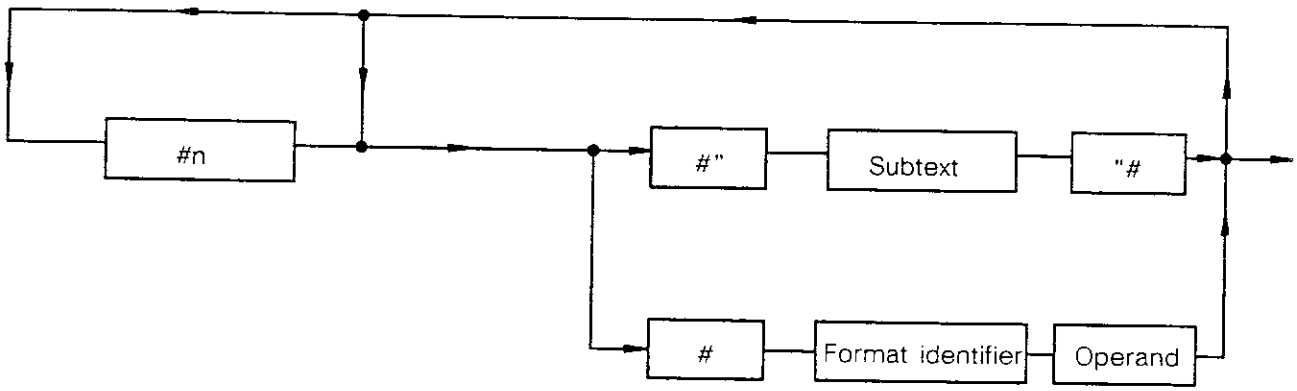
When the program is started, the PLC checks the texts to determine whether or not the maximum length is exceeded.

Syntax (structure) of texts

A text for the DRUCK block consists of:

- The text number
- One or several subtexts (optional)
- Operands with format identifier (optional)

Syntax diagram of texts



#n : Number of the text to be entered as a direct constant (1...99).

#" : Start identifier for text input (see also PLC description, text processing chapter).

"# : End identifier for text input (only in terminal mode; see also PLC description, text processing chapter).

Caution: This end identifier is *not* specified when using the ABB Procontic programming system. Only when texts are entered directly in the PLC in terminal mode do these have to be terminated with this identifier.

Subtext: ASCII characters 1...FF. The ASCII character <NUL> is used as a prefix for the format identifier in the send buffer.

Format identifier: The format identifier specifies the type of the operand and the display format of its value. The format identifier is a 3-digit number. The 1st digit from the left specifies the operand type. There are 3 operand types:

Binary: 1
Word : 2
Double word: 3

Numbers 2 and 3 define the display format.

Operand: Binary, word or double word operands whose values are output depending on the display format.

Input of texts

Two text input cases can be distinguished:

- Input of texts in the ABB Procontic programming system
- Input of texts directly in the PLC using a terminal

Input of texts in the ABB Procontic programming system

On input into the programming system, the following parts of the overall text are treated as independent operands:

- the text number	e.g. # 1
- a subtext	e.g. #" Text1
- a format identifier	e.g. # 203
- an operand	e.g. MW 2,3
- a further subtext	e.g. #" Text2

In the function block diagram, each of these parts occupies one of the inputs TX0...TXn at the symbol of the DRUCK block.

Input of special characters for screen or printer control:

Control characters such as "line feed" <LF> or "carriage return" <CR> are needed to arrange the text when output to a screen or printer. These special characters can be placed anywhere within a subtext. In the programming system, these special characters are entered by means of:

\Numerical value of the character

The numerical value of the character is specified as a three-digit decimal number.

Example:

The following output is to be made on a printer:

First line
Blank line
Second line

To do this, the following text must be planned:

First line <CR> <LF> <LF> second line

The following applies:

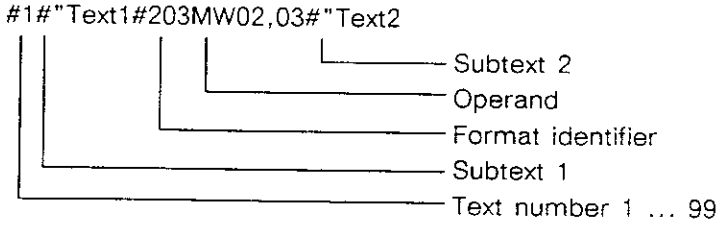
<CR> = 013
<LF> = 010

The text input in the programming system is as follows:

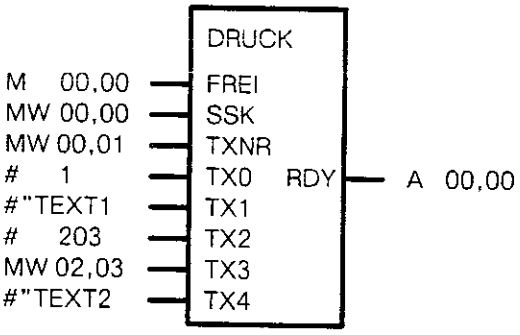
#"First line\013\010\010second line

Example

The text with the number 1 is to be planned in the DRUCK block. The text consists of two subtexts (text 1 and text 2). The value of the word flag MW 2,3 is to be output as a three-digit number (format 03) between these subtexts.



FBD/LD



IL

```
!BA    0
DRUCK
       M    00,00
       MW   00,01
       MW   00,00
       A    00,00
       #    1
       #"TEXT1
       #    203
       MW   02,03
       #"TEXT2
```

Further texts can be planned with numbers 2...99. The text with the serial number 2 is planned in the same way as from the input TX5. In this case, the number of the next text (#2) is specified at the input TX5.

OUPUT OF ASCII CHARACTERS AND HEX VALUES THROUGH A SERIAL INTERFACE

DRUCK

Input of texts with a terminal directly into the PLC

If no ABB Procontic programming system is available to the user, he can enter his PLC program directly into the PLC as an instruction list (IL) using a terminal, or can read it out.

In doing so, the texts are entered directly after the DRUCK block, i.e. the text is entered as from the program memory address following the operand at the RDY output.

Input of texts directly in the PLC

When entering texts directly in the PLC, the user must pay attention to the syntax rules for text formatting.

A text for the DRUCK block consists of:

- the text number
- one or several subtexts (optional)
- operand with format identifier (optional)

Texts are entered in the PLC within the "enter/edit user program" mode (see also the chapter "Operating and test functions" of the PLC description).

Entered texts or subtexts (special characters are also possible) are embedded in the special characters #" (start of text input) and "# (end of text input). Refer also to the chapter entitled "text processing" in the PLC description.

Characteristics: Texts for the DRUCK block

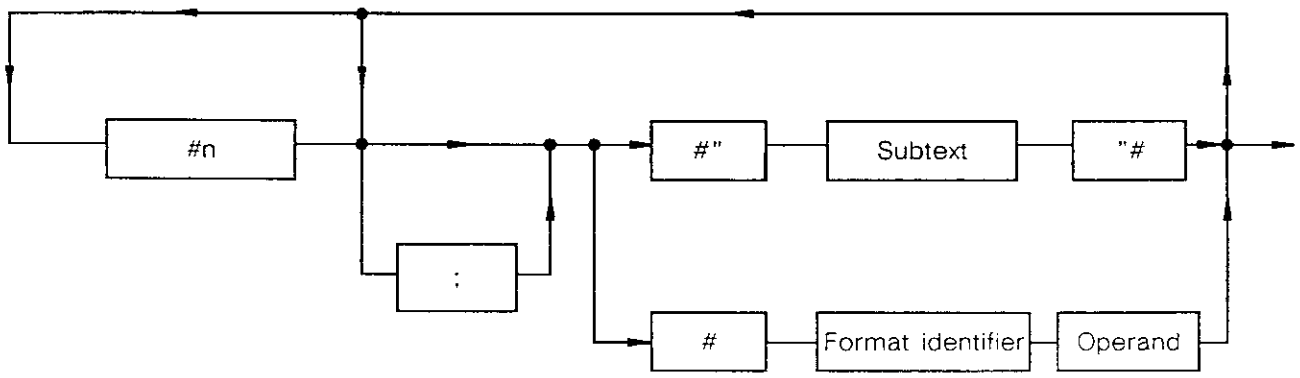
Storage: Directly after the DRUCK block
Quantity: 1...99

Length: Up to 256 characters (owing to their send buffer length = 256)

In this case, the following applies to the length of stored texts:
- Each text character counts as 1 character
- Each format identifier counts as 3 charaters
- Each bit operand counts as 1 character
- Each word operand counts as 2 characters
- Each double word operand counts as 4 characters

The program editing (PA) function checks texts to ensure that they will not exceed the maximum length.

Syntax diagramm of texts



#n : Number of the text to be entered as a direct constant (1...99).

#" : Start identifier for text input (see also PLC description, text processing chapter).

"# : End identifier for text input in terminal mode (see also PLC description, text processing chapter).

Caution: This end identifier is *not* specified when using the ABB Procontic programming system. Only when texts are entered directly in the PLC in terminal mode do these have to be terminated with this identifier.

Subtext: ASCII characters 1...FF. The ASCII character <NUL> is used as a prefix for the format identifier in the send buffer.

Format identifier: The format identifier specifies the type of the operand and the display format of its value. The format identifier is a 3-digit number. The 1st digit from the left specifies the operand type. There are 3 operand types:

Binary:	1
Word:	2
Double word:	3

Numbers 2 and 3 define the display format.

Operand: Binary, word or double word operands whose values are output depending on the display format.

Correcting a character during text input with a terminal

An incorrect character can *not* be corrected when entering a text on terminal. To correct the text input it must be terminated with the end of text input identifier "# and the program must be terminated by pressing <CR>. By program input, you must restart at the address of the character you wish to correct. Text input then begins with the start of text input identifier #".

Input of texts containing more than 80 characters on a terminal

When making a text input, a maximum of 80 characters can be displayed in each line of the screen. If the text is more than 80 characters long, text input must be concluded at the end of the screen line by means of the end of text input identifier "# and you must move to the next line in the screen by entering the semicolon (;). Now continue text input with the start of text input identifier #".

Communication between several DRUCK blocks and the same serial interface

Several DRUCK blocks may use *the same* serial interface without the need for taking special precautions. If the serial interface is occupied by a DRUCK block, the other DRUCK blocks automatically wait until it is free again.

- If several DRUCK blocks wish to access the same interface simultaneously, the block located nearest to the front in the user program is given access. Therefore, the order of priority corresponds to the sequence in which the DRUCK blocks are called in the user program.
- If the user wishes a different processing sequence, this must be planned by appropriate mutual interlokking of the DRUCK blocks.

Communication by a DRUCK block and an EMAS block with the same serial interface

A DRUCK block and an EMAS block may use the *same* serial interface without the need for special precautions.

Display formats for numerical values to be output

The PLC is capable of presenting the numerical values to be output on a screen or a printer in diverse ways.

The display format for a numerical value is defined by the format identifier. This is planned directly *before* the flag to be output and consists of three digits. It is defined as follows:

Format identifier: The format identifier specifies the type of the operand and the display format of its value. The format identifier is a three-digit number. The first digit from the left specifies the operand type. There are three operand types:

Binary:	1
Word:	2
Double word:	3

Digits 2 and 3 define the display format.

Examples:
- Format identifier 103
  - Digit 1: 1: Binary operand
  - Digits 2 and 3: 03: Display format 03 (see table)
- Format identifier 204
  - Digit 1: 2: Word operand
  - Digits 2 and 3: 04: Display format 04 (see table)
- Format identifier 341
  - Digit 1: 3: Double word operand
  - Digits 2 and 3: 41: Display format 41 (see table)

OUPUT OF ASCII CHARACTERS AND HEX VALUES THROUGH A SERIAL INTERFACE

Formats

All possible formats are listed in the following table. With the exception of the special indentifiers 98 and 99, all identifiers are applicable to the binary, word and double word data types.

Possible display formats

Displ. form. identifier	Display format	Numerical example	ASCII output
Format with leading zeros			
01	x	0012345678	8
02	xx		78
03	xxx		678
04	xxxx		5678
05	xxxxx		45678
37	xxxxxx		345678
38	xxxxxxx		2345678
39	xxxxxxxx		12345678
40	xxxxxxxxx		012345678
41	xxxxxxxxxx		0012345678
06	xxxx,x	02215	0221,5
07	xxx,xx		022,15
08	xx,xxx		02,215
09	x,xxxx		0,2215
10	,xxxxx		,02215
42	xxxxxxxxx,x	0011223344	001122334,4
43	xxxxxxxx,xx		00112233,44
44	xxxxxxx,xxx		0011223,344
45	xxxxxx,xxxx		001122,3344
46	xxxxx,xxxxx		00112,23344
47	xxxx,xxxxxx		0011,223344
48	xxx,xxxxxxx		001,1223344
49	xx,xxxxxxxx		00,11223344
50	x,xxxxxxxxx		0,011223344
51	,xxxxxxxxxx		,0011223344
11	+/- xxxxx	00331	+00331
12	+/- xxxx,x		+0033,1
13	+/- xxx,xx		+003,31
14	+/- xx,xxx		+00,331
15	+/- x,xxxx		+0,0331
16	+/- ,xxxxx		+,00331
52	+/- xxxxxxxxx	0055667788	+0055667788
53	+/- xxxxxxxx,x		+005566778,8
54	+/- xxxxxxx,xx		+00556677,88
55	+/- xxxxxx,xxx		+0055667,788
56	+/- xxxxx,xxxx		+005566,7788
57	+/- xxxx,xxxxx		+00556,67788
58	+/- xxx,xxxxxx		+0055,667788
59	+/- xx,xxxxxxx		+005,5667788
60	+/- x,xxxxxxxx		+00,55667788
61	+/- ,xxxxxxxxx		+0,055667788
62	+/- ,xxxxxxxxx		+,0055667788

Displ. form. identifier	Display format	Numerical example	ASCII output
Leading zeros substituted by blanks (in this example, blanks are indicated by _).			
17	x	00234	____4
18	xx		___34
19	xxx		__234
20	xxxx		__234
21	xxxxx		__234
63	x	0087654321	_________1
64	xx		________21
65	xxx		_______321
66	xxxx		______4321
67	xxxxx		_____54321
68	xxxxxx		____654321
69	xxxxxxx		___7654321
70	xxxxxxxx		__87654321
71	xxxxxxxxx		__87654321
72	xxxxxxxxxx		__87654321
22	xxxx,x	00347	__34,7
23	xxx,xx		__3,47
24	xx,xxx		__,347
25	x,xxxx		_,0347
26	,xxxxx		,00347
27	+/- xxxxx		+__347
28	+/- xxxx,x		+__34,7
73	xxxxxxxxx,x	0012345678	__1234567,8
74	xxxxxxxx,xx		__123456,78
75	xxxxxxx,xxx		__12345,678
76	xxxxxx,xxxx		__1234,5678
77	xxxxx,xxxxx		__123,45678
78	xxxx,xxxxxx		__12,345678
79	xxx,xxxxxxx		__1,2345678
80	xx,xxxxxxxx		__,12345678
81	x,xxxxxxxxx		_,012345678
82	,xxxxxxxxxx		,0012345678
83	+/- xxxxxxxxx		+ __12345678
84	+/- xxxxxxxxx,x		+ __1234567,8

Format identifier	Format	Numerical example	ASCII output
Addtional formats with suppression of leading zeros			
29	xxxxx	00839	839
30	xxxx,x		83,9
31	xxx,xx		8,39
32	xx,xxx		,839
33	+/- xxxxx		+839
34	+/- xxxx,x		+83,9
35	+/- xxx,xx		+8,39
36	+/- xx,xxx		+,839
85	xxxxxxxxxx	0000000321	321
86	xxxxxxxxx,x		32,1
87	xxxxxxxx,xx		3,21
88	xxxxxxx,xxx		,321
89	+/- xxxxxxxxx		+321
90	+/- xxxxxxxx,x		+32,1
91	+/- xxxxxxx,xx		+3,21
92	+/- xxxxxx,xxx		+,321

Special format: Output of a word operand's HEX value

The value of a word operand is output directly as a hexadecimal value. Therefore, the value is not converted to ASCII before output.

98 Only the LOW BYTE (8 bits) of the word operand is output

99 The LOW BYTE of the word operand is output first, followed by its HIGH BYTE

Important: This special format is only permissible for the "WORD" data type.
Permissible: 298 and 299
Inadmissible: 198, 199, 398 and 399

Example: Input of a text in the user program and output through the interface

The folowing text is to be output:

Text-No.: 3
Subtext 1: DISTANCE
Value of the flag: MW0,0
Output format: 3-digit with leading zeros
Subtext 2: KM

User Input	Meaning	Output through the serial interface
#3	Text-No. 3, direct constant	---
#"	Identifier for the start of text input	---
D	Letter in ASCII	44H
I	Letter in ASCII	49H
.	. . .	.
.	. . .	.
.	. . .	.
C	. . .	43H
E	. . .	45H
"# *)	Identifier for the end of text input	---
#203	Type, format identifier as a direct constant Type: Word, Format: 03	---
MW0,0	Contents of MW0,0	Formatted decimal value of the flag MW0,0 in ASCII
#"	Identifier for the start of text input	---
K	Letter in ASCII	4BH
M	" "	4DH
"# *)	Identifier for the end of text input	---
	End of text character (e.g. <CR>) if this is planned in the SINIT block and is enabled for output	0DH

*) The end identifier "# is dropped if input is made into the ABB Procontic programming system. The end identifier "# is needed only when making inputs directly into the PLC using a terminal.

Example of end of text characters to be specified in the SINIT block

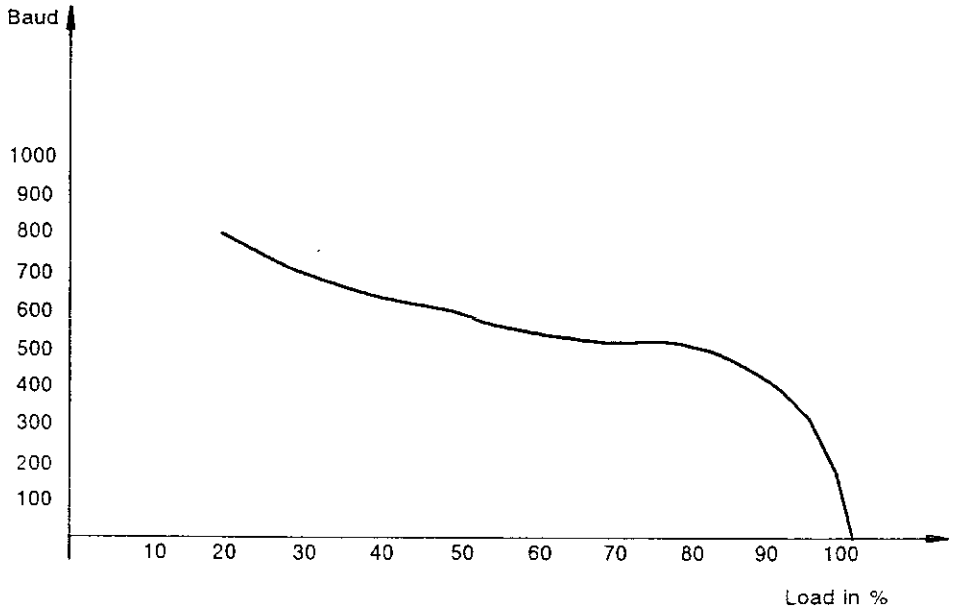
<ETX> = 03H = 3
<EOT> = 04H = 4
<CR> = 0DH = 13
<LF> = 0AH = 10
<SP> = 20H = 32
. . .
. . .
. . .

Effective data transmission rate when using the DRUCK block

The effective data transmission rate (charcters actually output through the serial interface) essentially depends on the load on the PLC and the baud rate set on the jumper panel X8.

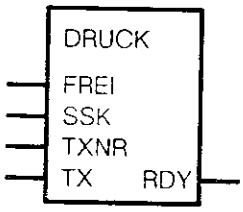
The following relationship roughly applies:

Marginal condition: Baud rate set on X8: 9600 baud



Important: The load indication (command AL) is correct only if *no* communication is taking place through the serial interface at the time of indication. Communication adulterates the load indication and it is therefore useless. If a user program contains the DRUCK or EMAS block, these must be disabled when determining the load.

CE FBD Definition



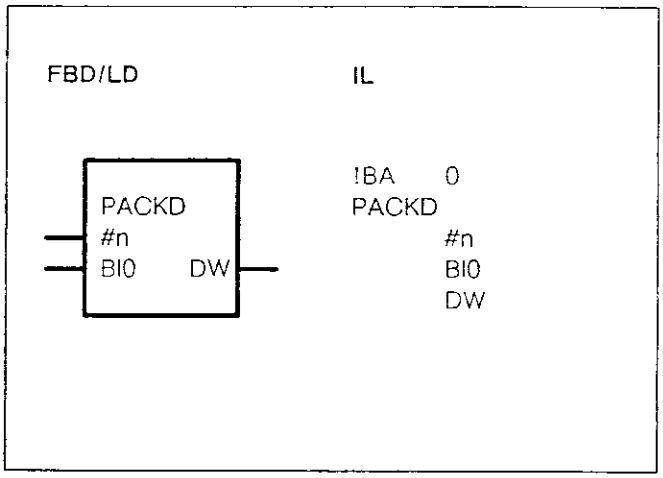
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
SSK	E	W	N	P	Y	0	0
TXNR	E	W	N	P	Y	0	0
TX	E	X	N	P	Y	1	0
RDY	A	L	N	P	Y	0	0

CE IL Definition

```
00000   !BA     0           Nr      Block No. (preset to 0)
00001   DRUCK
00002           PP  0       FREI    Input BINARY (block enable)
00003           PP  0       TXNR    Input WORD (text No.)
00004           PP  0       SSK     Input WORD (interface identifier)
00005           PP  0       RDY     Output BINARY (output text)
        [       1
00006           PP  1       TX      Input ALL  (No., text, format, 0)
        ]       1
```

PACK BINARY VARIABLES IN DOUBLE WORD — PACKD

This function block packs n binary variables in one double word variable.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of binary variables to be packed
BI0	BINARY	E, M, A, K, S	1st binary variable to be packed; the input is capable of duplication
DW	DOUBLE WORD	MD	Double word variable

CE Data

Runtime:	
Basic runtime:	31 µs
Additional runtime:	9 µs, per planned binary variable
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block packs n binary variables in one double word variable.

#n DIRECT CONSTANT (#, #H)
The number of binary variables to be packed is specified at the input #n. This is specified as a direct constant.
The following applies: $1 \leq n \leq 32$
$n = 0$ is forbidden!

BI0 BINARY
The input BI0 is capable of duplication (BI0...BI31). The binary variables to be packed are specified at the inputs BI0...BIn-1.

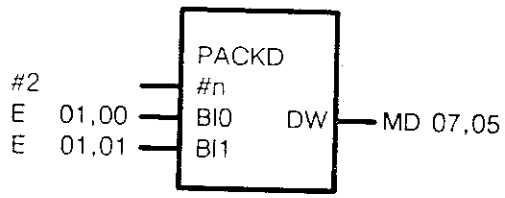
DW DOUBLE WORD
The value of each binary variable at the inputs BI0...BIn is loaded into the corresponding bit (bit 0...bit 31) of the variable at the output DW.

Affiliations
BI0 -> bit 0 of the output variable
BI1 -> bit 1 of the output variable
.
.
.
BI31 -> bit 31 of the output variable

Note: If the user plans less than 32 binary input variables, the bits of the output variables not needed are occupied with the value 0.

Example

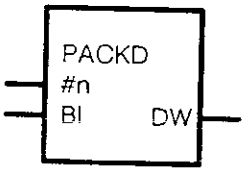
FBD/LD



IL

```
!BA    0
PACKD
       #2
       E   01,00
       E   01,01
       MD 07,05
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
BI	E	L	N	P	Y	1	0
DW	A	D	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	PACKD			
00002	[	PP 0 1	#n	# CONSTANT (number of bits)
00003	]	PP 1 1	BI	Input BINARY
00004		PP 0	DW	Output DOUBLE WORD

PACK BINARY VARIABLES IN WORD

This function block packs n binary variables in one word variable.

FBD/LD

IL

```
!BA   0
PACK
      #n
      BI0
      WORT
```

Parameters

#n	DIRECT CONSTANT	#,#H	Number of binary variables to be packed
BI0	BINARY	E, M, A, K, S	1st binary variable to be packed; the input is capable of duplication
WORT	WORD	AW, MW	Word variable

CE Data

Runtime:	
Basic runtime:	25 µs
Additional runtime:	8 µs per planned binary variable
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block packs n binary variables in one word variable.

#n DIRECT CONSTANT (#, #H)
The number of binary variables to be packed is specified at the input #n. This is specified as a direct constant.
The following applies: $1 \leq n \leq 16$
$n = 0$ is forbidden!

BI0 BINARY
The input BI0 is capable of duplication (BI0...BI15). The binary variables to be packed are specified at the inputs BI0...BIn-1.

WORT WORD
The value of each binary variable at the inputs BI0...BIn-1 is loaded into the corresponding bit (bit 0 ... bit 15) of the variable at the output WORT.

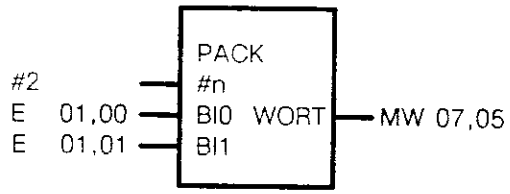
Affiliations
BI0 -> bit 0 of the output variable

BI1 -> bit 1 of the output variable
. .
. .
BI15 -> bit 15 of the output variable

Note: If the user plans less than 16 binary input variables, the bits of the output variable not needed are occupied with the value 0.

Example

FBD/LD



IL

```
!BA    0
PACK
       #2
       E   01,00
       E   01,01
       MW 07,05
```

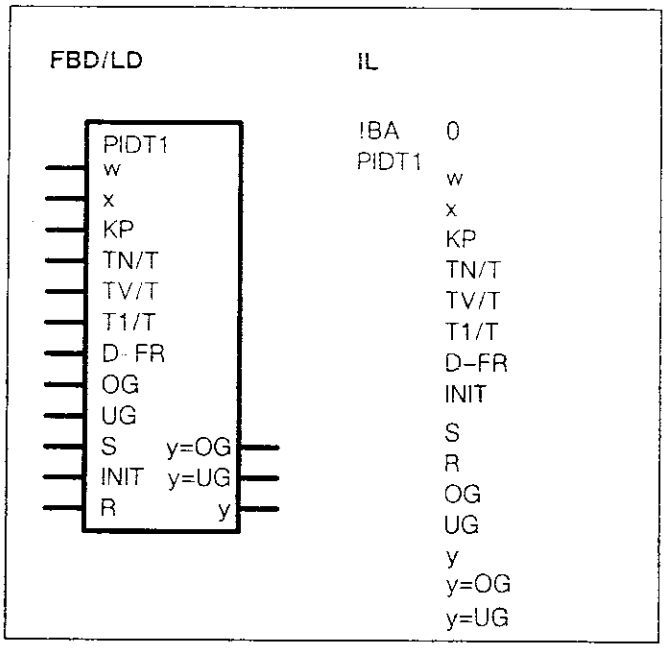
CE FBD Definition

Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
BI	E	L	N	P	Y	1	0
WORT	A	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	PACK			
00002		PP 0	#n	# CONSTANT (number of bits)
	[	1		
00003		PP 1	BI	Input BINARY
	]	1		
00004		PP 0	WORT	Output WORD

The PI controller changes its output y (manipulated variable) until the input x (controlled variable) is equal to the input w (command variable).



Parameters

w	WORD	EW, AW, MW, KW	Command variable (setpoint)
x	WORD	EW, AW, MW, KW	Controlled variable (actual value)
KP	WORD	EW, AW, MW, KW	Proportional coefficient, specified as a percentage
TN/T	WORD	EW, AW, MW, KW	Integral action time scaled to the PLC cycle time
TV/T	WORD	EW, AW, MW, KW	Derivative action time scaled to the PLC cycle time
T1/T	WORD	EW, AW, MW, KW	Returning time scaled to the PLC cycle time
D-FR	BINARY	E, A, M, K	Enable DT1 component
OG	WORD	EW, AW, MW, KW	High limit for the manipulated variable y
UG	WORD	EW, AW, MW, KW	Low limit for the manipulated variable y
S	BINARY	E, A, M, S, K	Enable for setting to initial value INIT
INIT	WORD	EW, AW, MW, KW	Initial value for the manipulated variable y
R	BINARY	E, A, M, S, K	Reset the manipulated variable y to the value 0
y=OG	BINARY	A, M	High value has been reached
y=UG	BINARY	A, M	Low value has been reached
y	WORD	AW, MW	Output for manipulated variable y

CE Data

Runtime:	
Basic runtime:	318 µs without DT1 component, 564 µs with DT1 component
Additional runtime:	no
Output updating:	yes
Number of historical values:	9 words
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The PI controller changes its output y (manipulated variable) until the input x (controlled variable) is equal to the input w (command variable).

Transfer function:

$$F(s) = KP * (1 + \frac{1}{s * TN} + \frac{s * TV}{1 + (s * T1)})$$

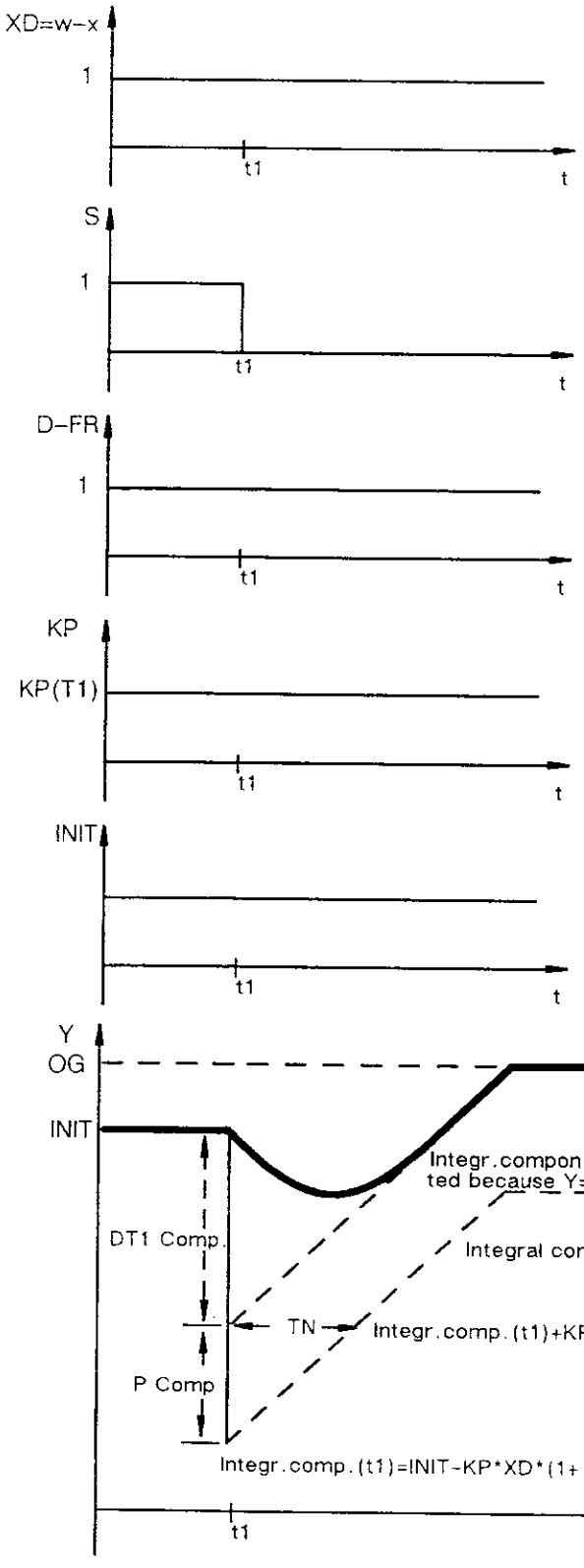
Control algorithm: simple rectangle rule

$$y = \frac{KP * XD}{100} + \frac{KP}{100} * \frac{XD}{TN/TZ} + YI(z-1) + \frac{T1/TZ}{1 + (T1/TZ)} * (YDT1(z-1) + \frac{1}{T1/TZ} * \frac{TV}{TZ} * \frac{KP}{100} * (XD - XD(z-1)))$$

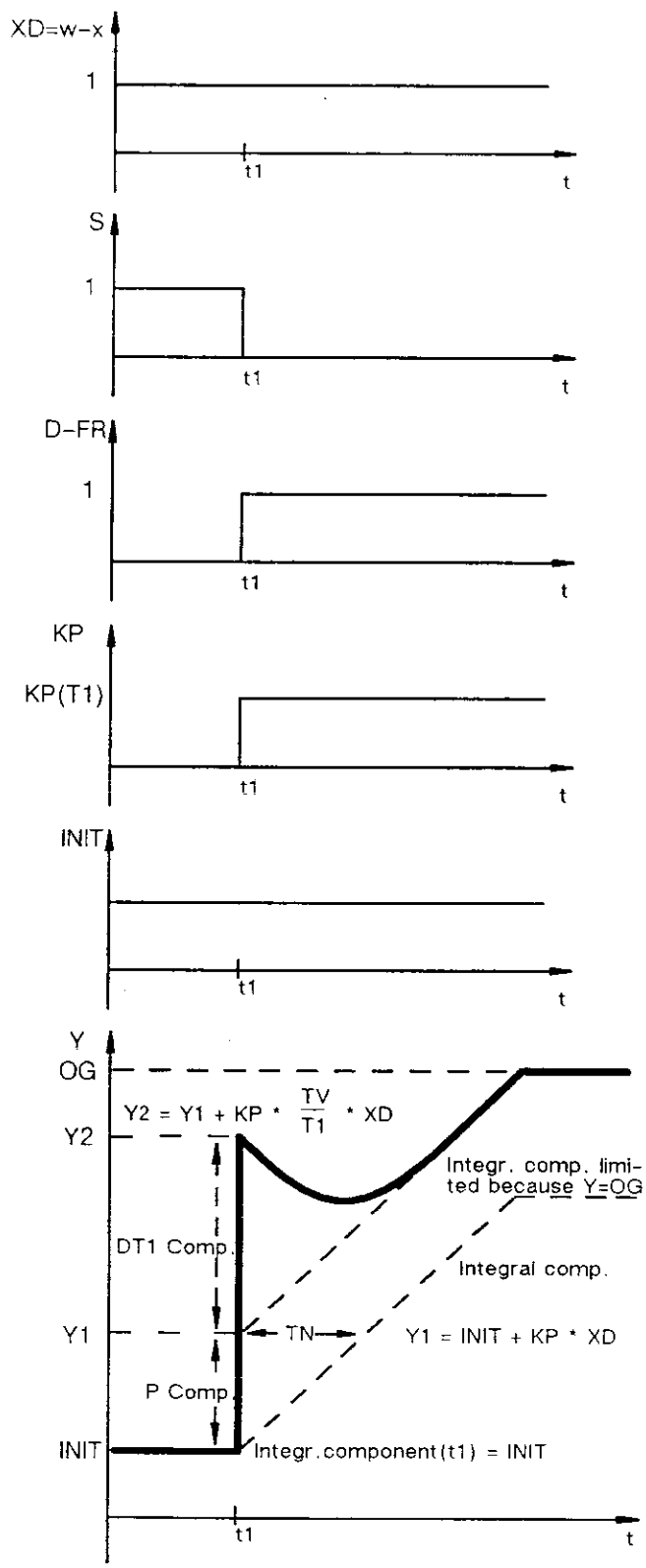
where:

YI(z-1): The integral portion from the previous program cycle
XDT1(z-1): The differential portion from the previous program cycle
XD(z-1): Control system difference from the previous program cycle

The inputs and outputs can neither be duplicated nor negated/inverted.



PIDT1 controller: Surge-free transition from the specified intitial value to control mode



PIDT1 controller: Surging transition from the specified initial value to control mode

w WORD

The command variable (setpoint) is specified at the input w.

x WORD

The controlled variable (actual value) is specified at the input x.

KP WORD

The proportional coefficient is specified at the input KP. This value is specified as a percentage and may be positive or negative.

Example:

1	=	1 Percent
55	=	55 Percent
100	=	100 Percent
1000	=	1000 Percent
-500	=	-500 Percent

- 1 percent means that the block multiplies the system deviation by the factor 0.01 (see also control algoritm)
- 100 percent means that the block multiplies the system deviation by the factor 1 (see also control algoritm)
- 1000 percent means that the block multiplies the system deviation by the factor 10 (see also control algoritm)

Generally, proportional coefficients of more than 1000% are not meaningful in control systems.

TN/T WORD

The integral action time TN is scaled to the PLC cycle time T and is specified at the input TN/T.

Value range: $0 \leq TN/T \leq 328$

- If values are specified which are beyond the admissible value range the PLC uses the value 328.
- A large integral action time TN can be achieved by choosing a great cycle time T, too. If the block is used within a run number block, the cycle time of the run number block is valid for block PIDT1 and not the cycle time (KD 0,0) of the PLC program.

TV/T WORD

The derivative action time TV is scaled to the PLC cycle time T and is specified at the input TV/T ($0 < TV/T \leq 32767$).

T1/T WORD

The returning time T1 is scaled to the PLC cycle time T and is specified at the input T1/T ($0 < T1/T \leq 32767$).

The returning time is the time in which the DT1 component has decreased to approximately 37% of its initial value.

Inadmissible time parameters

Every time value is set to the maximum positive value 32767 if the time value at the input is erroneously specified as less than or equal to "0".

D-FR WORD

The DT1 component of the controller can be connected or deactivated by means of the D-FR input.

D-FR	Meaning
0	DT1-Comp. is deactivated -> pure PI controller
1	DT1-Comp. is connected -> PIDT1 controller

In the following cases, from the control point of view it is often disturbing and not meaningful for the DT1 component to be active:

- During activations
- In the event of large system deviations
- When setting the controller to a specified initial value
- When resetting the controller to the value 0

The command and controlled variables can be compared outside of the controller. Depending on this comparison, the DT1 component can be activated or deactivated specifically by way of the D-FR input.

For example, activation can be restricted to ensuring that the system deviation is within a required bandwidth. That is to say, the DT1 component is only active if the controlled variable fluctuates around the setpoint within a specific bandwidth. The DT1 component is deactivated if the controlled variable leaves this tolerance band.

Limiting the maniulated variable y

OG WORD

UG WORD

The output y (manipulated variable) of the controller can be limited

- To a maximum value by specifying a limit at the input OG (high limit);
- To a minimum value by specifying a limit at the input UG (low limit)

The high and low limits *also* apply to the controller's internal I component. That is to say, the I component can only assume values between the high and low limits. If the manipulated variable y reaches one of the two limits, the controller's I component is *no longer* altered.

This prevents the I component from running amock in the event of limiting of the controller output y, assuming meaningless values from the point of view of control and, in certain circumstances, not returning to the operating range until after a very long time. This response of a controller is also referred to as a "special anti-reset windup measure (ARW)".

Setting and resetting the controller

S BINARY
INIT WORD
R BINARY

Setting the controller to an initial value

- The output y of the controller is set to the initial value specified at the INIT input by means of a 1 signal at the input S (set).

- A 1 signal at the input R (reset) is equivalent to specifying the initial value 0 (see above).

Surge-free setting/resetting

- The output y of the controller is set to the initial value specified at the input INIT by means of a 1 signal at the binary input S (set).

- A 1 signal at the input R (reset) is equivalent to specifying the initial value 0.
  In doing so, adjustment to the initial value takes place internally in the controller. This adjustment is a shift of the controller output from the momentary value to the required initial value. Now, the controller continues operating from this initial value precisely as it would have done at the old operating point before the shift, i.e. without surges. The controller's I component is defined so that the sum of the P component, I component and DT1 component just results in the initial value.

Advantage of surge-free operation:

- Control as from the new initial value is devoid of surges.

Disadvantage of surge-free operation:

- The following equation applies:

  I component = INIT - P component - DT1 component
  In certain circumstances, the I component is set to high values and may take very long before this "wrong" I component from the point of view of control is dissipated again.

Surging setting/resetting

- The output y of the controller is set to the initial value specified at the INIT input by means of a 1 signal at the input S (set).

- A 1 signal at the input R (reset) is equivalent to specifying the initial value 0.

- In the event of surging setting or resetting of the controller, the I component is set equal to the initial value. To do this, the P and DT1 components must be suppressed during setting.

Where: I component = INIT

Surging setting to an initial value is achieved by the following measures during setting:

- Deactivation of the DT1 component via the D-FR control input and

- Specifying the value 0 at the input KP.

These measures render the P component and the DT1 component inactive during setting of the controller.

The controller output assumes the initial value during the set cycle.

The P and DT1 components are enabled again after the set cycle. From the initial value, the controller output y jumps according to the P and DT1 components of the controller.

Advantage of surging setting:

- The I component is not set to "wrong" values from the point of view of control.

Disadvantage of surging setting:

- No freedom from surging

y WORD
The controller's manipulated variable y is output through the output y.

y=OG BINARY
The output y=OG signals whether or not the value at the output y has reached the specified high limit.

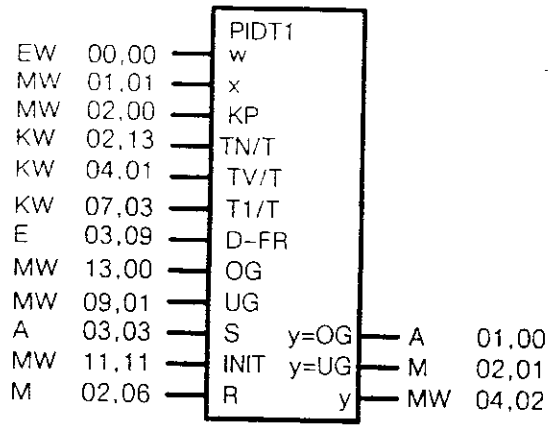
y=OG = 0 limit has *not* been reached.
y=OG = 1 limit has been reached.

y=UG BINARY
The output y=UG signals whether or not the value at the output y has reached the specified low limit.

y=UG = 0 limit has *not* been reached.
y=UG = 1 limit has been reached.

Example

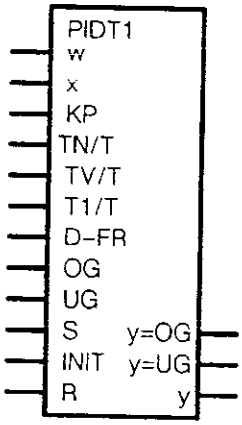
FBD/LD



IL

```
!BA     0
PIDT1
        EW   00,00
        MW   01,01
        MW   02,00
        KW   02,13
        KW   04,01
        KW   07,03
        E    03,09
        MW   11,11
        A    03,03
        M    02,06
        MW   13,00
        MW   09,01
        MW   04,02
        A    01,00
        M    02,01
```

CE FBD Definition



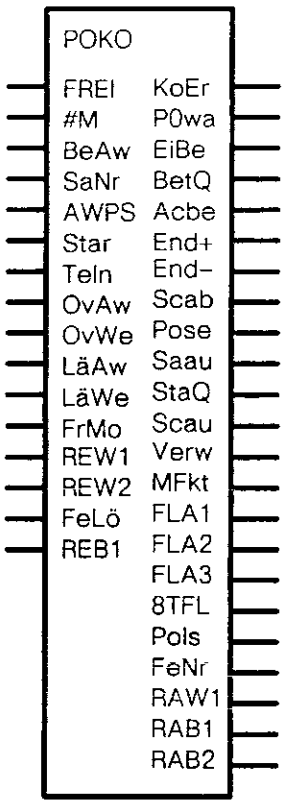
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
w	E	W	N	P	Y	0	0
x	E	W	N	P	Y	0	0
KP	E	W	N	P	Y	0	0
TN/T	E	W	N	P	Y	0	0
TV/T	E	W	N	P	Y	0	0
T1/T	E	W	N	P	Y	0	0
D-FR	E	L	N	P	Y	0	0
OG	E	W	N	P	Y	0	0
UG	E	W	N	P	Y	0	0
S	E	L	N	P	Y	0	0
INIT	E	W	N	P	Y	0	0
R	E	L	N	P	Y	0	0
y=OG	A	L	N	P	Y	0	0
y=UG	A	L	N	P	Y	0	0
y	A	W	N	P	Y	0	0

CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)	
00001	PIDT1				
00002		PP 0	w	Input	WORD (command variable)
00003		PP 0	x	Input	WORD (controlled variable)
00004		PP 0	KP	Input	WORD (proportional coefficient)
00005		PP 0	TN/T	Input	WORD (integral action time)
00006		PP 0	TV/T	Input	WORD (derivative time)
00007		PP 0	T1/T	Input	WORD (returning time)
00008		PP 0	D-FR	Input	BINARY (enable DT1 component)
00009		PP 0	INIT	Input	WORD (initial value)
00010		PP 0	S	Input	BINARY (set to INIT)
00011		PP 0	R	Input	BINARY (reset to 0)
00012		PP 0	OG	Input	WORD (high limit)
00013		PP 0	UG	Input	WORD (low limit)
00014		PP 0	y	Output	BINARY (manipulated variable)
00015		PP 0	y=OG	Output	BINARY (OG reached)
00016		PP 0	y=UG	Output	BINARY (UG reached)

Together with the auxiliary blocks POKANF and POKEND, the function block POKO handles communication between the PLC and **one** single axis positioner 35 PO 90.

FBD/LD



IL

```
!BA   0
POKO
      #0
      #0
      FREI
      #M
      BeAw
      SaNr
      AWPS
      Star
      Teln
      OvAw
      OvWe
      LäAw
      LäWe
      FrMo
      REW1
      REW2
      FeLö
      REB1
      KoEr
      P0wa
      EiBe
      BetQ
      Acbe
      End+
      End-
      Scab
      Pose
      Saau
      StaQ
      Scau
      Verw
      MFkt
      FLA1
      FLA2
      FLA3
      8TFL
      Pols
      FeNr
      RAW1
      RAB1
      RAB2
```

Parameters

#0	DIRECT CONSTANT	#0	Dummy parameter in the IL; here the editing program PA of the PLC enters the address of the subsequent POKO block. This parameter is *not* specified in the FBD
#0	DIRECT CONSTANT	#0	Dummy parameter in the IL; here, the editing program PA of the PLC enters the pointer to the historical value of the next POKO. This parameter is *not* specified in the FBD.
FREI	BINARY	E, A, M, K, S	Block enable
#M	DIRECT CONSTANT	#, #H	Module number (device number) of the affiliated 35 PO 90 module
BeAw	WORD	EW, AW, MW, KW	Selection of the 35 PO 90's mode
SaNr	WORD	EW, AW, MW, KW	Selection of the 35 PO 90's NC set
AWPS	WORD	EW, AW, MW, KW	Selection of the user program segment of the 35 PO 90
Star	BINARY	E, A, M, K, S	Start of an NC set
Teln	BINARY	E, A, M, K, S	Adoption of the momentary actual position value as the position setpoint for the selected NC set.
OvAw	WORD	EW, AW, MW, KW	Speed override selection
OvWe	WORD	EW, AW, MW, KW	External override value
LäAw	WORD	EW, AW, MW, KW	Length compensation selection
LäWe	DOUBLE WORD	MD,KD	Value of the external length compensation
FrMo	WORD	EW, AW, MW, KW	Enable mode
REW1	WORD	EW, AW, MW, KW	Reserved; enter the value 0
REW2	WORD	EW, AW, MW, KW	Reserved; enter the value 0
FeLö	BINARY	E, A, M, K, S	Clear error number
REB1	BINARY	E, A, M, K, S	Reserved; enter the value 0
KoEr	BINARY	A, M	Communication disturbed
POwa	BINARY	A, M	POKO waiting, latching output
EiBe	BINARY	A, M	Input limiting, latching output
BetQ	WORD	AW, MW	Quit modes
Acbe	BINARY	A, M	Axis ready
End+	BINARY	A, M	End position for positive direction
End–	BINARY	A, M	End position for negative direction
Scab	BINARY	A, M	Max. trailing distance exceeded
Pose	BINARY	A, M	Position reached
Saau	BINARY	A, M	Set executed
StaQ	BINARY	A, M	Start acknowledgement for NC set
Scau	BINARY	A, M	Loop executed
Verw	BINARY	A, M	Dwell time
MFkt	WORD	AW, MW	Machine function
FLA1	BINARY	A, M	FLAG1
FLA2	BINARY	A, M	FLAG2
FLA3	BINARY	A, M	FLAG3
8TFL	WORD	AW, MW	8 section flags
Pols	DOUBLE WORD	MD	Actual position value of the axis in absolute dimensions
FeNr	WORD	AW, MW	Error number
RAW1	WORD	AW, MW	Reserved
RAB1	BINARY	A, M	Reserved
RAB2	BINARY	A, M	Reserved

CE Data

Runtime:	
Basic runtime:	350 µs, regardless of the waiting time for the communication with 35 PO 90
Additional runtime:	none
Output updating:	yes
Number of historical values:	2 words
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

Together with the auxiliary blocks POKANF and POKEND, the POKO block handles communication between the PLC and *one* single axis positioning module 35 PO 90. The 35 PO 90 positioning module is the AXUMERIK-m 35 AM 50 adapted to the ABB Procontic T300 system. It serves to position *one* drive with the ABB Procontic T300 system.

The 35 PO 90 module is a passive station in the ABB Procontic T300 system.

Data exchange between the PLC (active station) and the 35 PO 90 takes place by way of special data structures. In the case of the 35 PO 90 module, these data structures are called MPST bus inputs/outputs. On the MPST bus, the 35 PO 90 module occupies an address space of 256 bytes.

An application note is available. Title:

"Application examples for ABB Procontic T300 with the single axis positioning module 35 PO 90 and operating station 35 BS 93" (type designation: 907 PB 36; only in German).

The 35 PO 90 module description contains further information. This also provides further information about the block parameters of the POKO block, with the following exceptions:

- FREI input, enable block
- KoEr output, communication error
- POwa output, POKO waiting
- EiBe output, input limiting

These signals are used/generated only by the POKO block and are not available on the 35 PO 90 module.

The POKO block serves to control one 35 PO 90 module. To do this, it requires the auxiliary communication blocks POKANF and POKEND. All block parameters for communication of a PLC program with a 35 PO 90 module are planned in a POKO block. These block parameters are also used by the POKANF and POKEND blocks.

The POKANF, POKO and POKEND blocks communicate with each other by way of an internal memory area of the PLC (historical values memory). Communication takes place automatically and without the user having to do anything. The block outputs KoEr, POwa and EiBe are set or reset exclusively by the POKO block. The POKANF and POKEND blocks communicate these outputs to the POKO block by way of the internal memory area.

Initialization of the 35 PO 90

After a cold start of the 35 PO 90 or a single board reset of the 35 PO 90, the POKO block sets the output KoEr = 1 and carries out initial communication with the affiliated 35 PO 90 module. If the 35 PO 90 module can be addressed correctly, the POKO block internally acknowledges this for the POKANF and POKEND blocks and sets the block output KoEr = 0.

POKANF

After initialization, the POKANF block requests communication with *all* planned 35 PO 90 modules for the POKO and POKEND blocks. The period of time between requesting communication by the PLC and permission to communicate by the 35 PO 90 module amounts to a maximum of 10 ms.

POKO

In each PLC cycle, the POKO block reads the 35 PO 90 MPST bus outputs and provides their values at its own outputs for evaluation by the PLC program.

Exception: KoEr = 1 after calling the POKO block. In this case, communication between the PLC and the 35 PO 90 module is disturbed and the values at the outputs of the POKO block are therefore invalid.

POKEND

In each PLC cycle the POKEND block transfers the values of the inputs planned on the POKO block to the MPST bus inputs of the 35 PO 90 module. This ensures that all signal changes of these POKO inputs are acquired up to the point when they are transferred by the POKEND block. That is to say, the results of the evaluation program, which is located after the POKO block, are still communicated to the 35 PO 90 module in the same program cycle.

Exception: Transfer does not take place if communication is disturbed. This is recognizable by virtue of the fact that after processing of the POKO block in the next program cycle its output KoEr is set to 1.

The input values of the POKO block are checked for admissiblity with their permissible value range. This is done before transfer by the POKEND block. If the values lie outside the permissible range, the POKEND block sets them to specified limits. In this case, in the next program cycle, after processing of the POKO block, its input limiting output EiBe is set in latching form to 1. The EiBe output can only be reset as follows:

- By aborting and subsequently starting the PLC program in INI OPS mode;
- By occurrence of a communication error, i.e. KoEr = 1; for example, this communication error can also be provoked by a single board reset.

The POKEND block also terminates communication requested by POKANF with all 35 PO 90 modules.

Disturbance of communication

If communication with a 35 PO 90 module is disturbed (35 PO 90 not yet ready, 35 PO 90 single board reset,...), then the output KoEr = 1 after processing of POKO. The values present at the POKO block outputs then do not agree with the 35 PO 90 MPST bus outputs and are thus invalid.

After a disturbance of communication with the 35 PO 90, the POKO block attempts to address it again. The KoEr output remains set until the 35 PO 90 module can again be addressed correctly.

Communication of the POKO block and communication of the POKANF and POKEND blocks with the affiliated 35 PO 90 module is aborted by resetting the block enable (FREI = 0). In the next program cycle, the KoEr output is set (KoEr = 1) after processing of the POKO block.

FREI BINARY

FREI = 0 -> No communication takes place between the POKO, POKANF and POKEND blocks and the affiliated 35 PO 90 module.
-> Block output KoEr = 1 (see application notes).

FREI = 1 -> Communication takes place between the POKO, POKANF and POKEND blocks and the affiliated 35 PO 90 module.

#M DIRECT CONSTANT

35 PO 90 Module number (unit No.)
Value range : 0–255 (8 Bits)

Affiliation: Module number and unit address

Module number	Device address
0	2000:FFXX
1	2000:FEXX
:	:
255	2000:00XX

XX: Address space 0..255 Bytes, for one 35 PO 90 module.

If #M > 255 or #M < 0
-> Block output KoEr = 1
and in the next cycle EiBe is
set to 1 in latching mode by POKO

BeAw WORD

Selection of the mode for the 35 PO 90 module
Value range: 0...3

0 -> No external mode
1 -> External automatic single set
2 -> External automatic follow-up set
3 -> External mode

If BeAw > 3 or BeAw < 0
-> POKEND sets the default BeAw = 0
and in the next cycle EiBe is set to 1
in latching mode by POKO

SaNr WORD

Specification of the NC set No. for 35 PO 90 module
Value range: 0...255

If SaNr > 255 or SaNr < 0
-> POKEND sets the default SaNr = 255
and EiBe is set to 1 in a latching mode
in the next cycle by POKO

AWPS WORD

Specification of the user program segment for 35 PO 90.
Value range: 0 ... 255.

If AWPS > 255 or AWPS < 0
-> POKEND sets the default AWPS = 0 and EiBe is set to 1 in latching mode in the next cycle by POKO.

Star BINARY

Start of an NC set

TeIn BINARY

Adoption of the momentary actual position value as the position setpoint for the selected NC SET.

OvAw WORD

Speed override selection
Value range: 0...3

If OvAw > 3 oder OvAw < 0
-> POKEND sets the default OvAw = 0 and in the next cycle EiBe is set to 1 in latching mode by POKO.

OvWe WORD

Value of the external override
Value range: 0...120

If OvWe > 120
-> POKEND sets the default OvWe = 120 and in the next cycle EiBe is set to 1 in latching mode by POKO;

if OvWe < 0
-> POKEND sets the default OvWe = 0 and in the next cycle EiBe is set to 1 in latching mode by POKO.

LäAw WORD

Length compensation selection
Value range: 0...4

If LäAw > 4 oder LÄAw < 0
-> POKEND sets the default LäAw = 0 and in the next cycle EiBe is set to 1 in latching mode by POKO.

LäWe DOUBLE WORD

External length compensation value
Value range: -9999999D ... +9999999D

If LäWe > +9999999D
-> POKEND sets the default LäWe = +9999999D and in the next cycle EiBe is set to 1 in latching mode by POKO;

If LäWe < -9999999D
-> POKEND sets the default LäWe = -9999999D and in the next cycle EiBe is set to 1 in latching mode by POKO.

FrMo WORD

Enable mode selection
FrMo = 0 -> Enable mode 1
FrMo = 1 -> Enable mode 2
FrMo = 2 -> Enable mode 3

REW1 WORD

Reserved; enter the value 0 (e.g. via indirect constant).

REW2 WORT

Reserved; enter the value 0 (e.g. via indirect constant).

FeLö BINARY

Deletion of an error number
An error number at the FeNr output can be deleted with this input.
FeLö =1 -> Deletion (Setting to 0) of the error number at the FeNr output.
Important: Continuous 1 signal -> No more error messages are possible.

FeLö = 0 ->No deletion of the error number.

REB1 BINARY

Reserved; Enter the value 0 (e.g. via indirect constant).

KoEr BINARY

Communication with the affiliated 35 PO 90 module is disturbed.

KoEr = 1 -> Communication with the affiliated 35 PO 90 module has not been established or is disturbed. It can also be interrupted by FREI = 0 (see also application notes).
KoEr = 0 -> The affiliated 35 PO 90 module can be addressed correctly.

POwa BINARY

POKO block is waiting

POwa = 1 -> In the POKO block computing time is used up by waiting for communication with the 35 PO 90 module. This output is set in latching mode (see also application notes).
The POwa output can only be reset as follows:
- By aborting and then starting the PLC program in OPS INI mode (initialization of the operand memory)
- By occurrence of a communication error, i.e. output KoEr = 1. This case can also be provoked by a single board reset.

POwa = 0 -> In the POKO block no computing time is and has been used up by waiting.

EiBe BINARY

Diagnosis to the effect that a value concerning this axis lies or lay outside the value range.

EiBe = 1 -> Diagnosis to the effect that a value concerning this axis lies or lay outside of the value range (see also application notes).
The EiBe output can only be reset as follows:
- By aborting and then starting the PLC program in OPS INI mode (initialization of the operand memory)
- By occurrence of a communication error, i.e. output KoEr = 1. This case can also be provoked by a single board reset.

EiBe = 0 -> The input values at the POKO block lie and, also up to now, lay within the permissible value range

BetQ WORD

Indication of the current mode of the 35 PO 90
Value range 0...3

0 -> No external mode
1 -> External automatic single set
2 -> External automatic follow-up set
3 -> External mode

Acbe BINARY

Axis ready

Acbe = 0 -> Positioning of the 35 PO 90 not possible in external mode or interrupted.

Acbe = 1 -> Positioning of the 35 PO 90 allowed or not interrupted in external mode.

End+ BINARY

End position for positive direction

End+ = 1 -> End position for positve direction moved to.
End+ = 0 -> End position for positive direction not moved to.

End- BINARY

End position for negative direction

End- = 1 -> End position for negative direction moved to.
End- = 0 -> End position for negative direction not moved to.

Scab BINARY

Maximum following error exceeded.

Scab = 1 -> Maximum following error has been exceeded.
Scab = 0 -> Maximum following error has not been exceeded.

Pose BINARY

Position reached

Pose = 1 -> Axis in the target window
Pose = 0 -> Axis not in the target window

Saau BINARY

Set executed

Saau = 1
- After connection of the supply voltage
- After execution of an NC set
- After aborting of a positioning set with 35 PO 90 enable initiator = 0, in enable mode 2 or 3

Saau = 0 - After starting of an NC set

StaQ BINARY

Start acknowledgement

StaQ is set to 1 for one PLC cycle after starting of an NC set. In the case of an NC set with an execution time less than the cycle time, with this signal and the Saau signal it is possible to recognize that the NC set has been executed.

Scau BINARY

Loop executed

In a loop set, execution of a loop of several NC sets is defined.

Scau = 0 -> Loop set has been called and the loop is still being executed.
Scau = 1 -> No loop set is called or the loop has been executed.

Verw BINARY

Dwell time elapsed

Verw = 0 -> Indicates that NC set has been started.
Verw = 1 -> Indicates that NC set has been executed and dwell time has elapsed.

MFkt WORD

Machine function
Value range: 0–999

FLA1 BINARY

Programmed flag position 1 has been exceeded.

FLA1 = 1 -> Programmed flag position 1 has been exceeded.
FLA1 = 0 -> Programmed flag position 1 has not been exceeded.

FLA2 BINARY

Programmed flag position 2 has been exceeded.

FLA2 = 1 -> Programmed flag position 2 has been exceeded.
FLA2 = 0 -> Programmed flag position 2 has not been exceeded.

FLA3 BINARY

Programmed flag position 3 has been exceeded.

FLA3 = 1 -> Programmed flag position 3 has been exceeded.
FLA3 = 0 -> Programmed flag position 3 has not been exceeded.

8TFL WORD

8 section flags

A section flag is set to 1 after 1/8 of the set's travel distance in each case.

Bit 0 -> Section flag 1 has been exceeded
Bit 1 -> Section flag 2 has been exceeded
Bit 2 -> Section flag 3 has been exceeded
Bit 3 -> Section flag 4 has been exceeded
Bit 4 -> Section flag 5 has been exceeded
Bit 5 -> Section flag 6 has been exceeded
Bit 6 -> Section flag 7 has been exceeded
Bit 7 -> Section flag 8 has been exceeded

Pols DOUBLE WORD

Actual position value of the axis in absolute dimensions

FeNr WORD

Error number

Error numbers of the 35 PO 90 module (e.g. 31 -> software limit switch moved to) are indicated at this output. New error numbers overwrite the previous ones. The error number 0FFFFH signifies an unidentifiable error.

After activation, the error number is set to 0.

Note:
Clear the error number by means of input FeLö = 1.

RAW1 WORD

Reserved

RAB1 BINARY

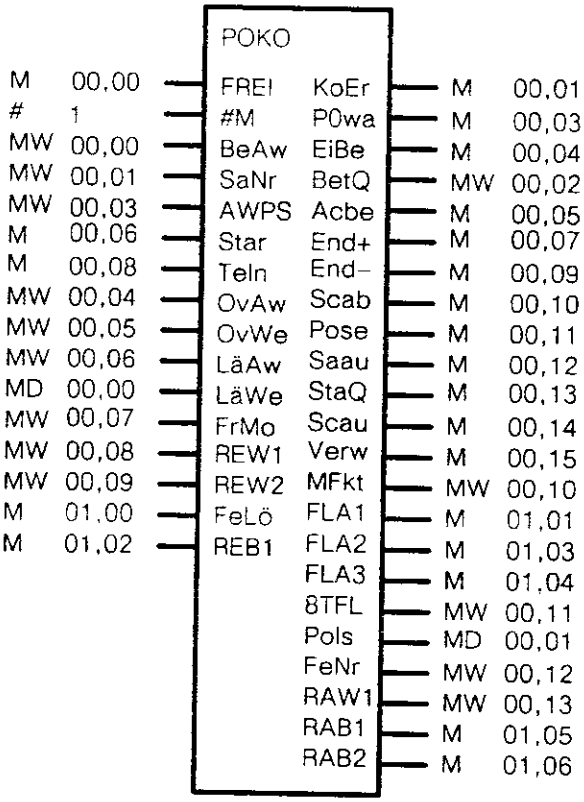
Reserved

RAB2 BINARY

Reserved

Example

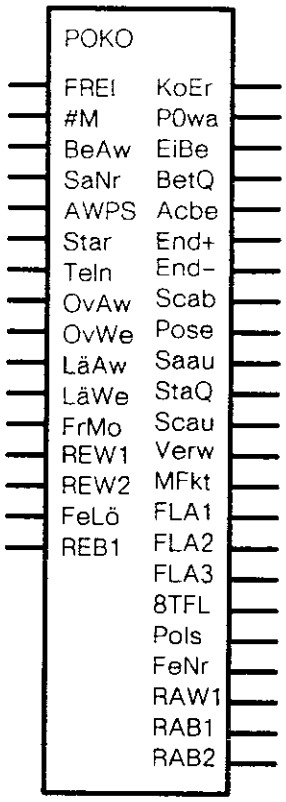
FBD/LD



IL

```
!BA    0
POKO
       #    0
       #    0
       M    00,00
       #    1
       MW   00,00
       MW   00,01
       MW   00,03
       M    00,06
       M    00,08
       MW   00,04
       MW   00,05
       MW   00,06
       MD   00,00
       MW   00,07
       MW   00,08
       MW   00,09
       M    01,00
       M    01,02
       M    00,01
       M    00,03
       M    00,04
       MW   00,02
       M    00,05
       M    00,07
       M    00,09
       M    00,10
       M    00,11
       M    00,12
       M    00,13
       M    00,14
       M    00,15
       MW   00,10
       M    01,01
       M    01,03
       M    01,04
       MW   00,11
       MD   00,01
       MW   00,12
       MW   00,13
       M    01,05
       M    01,06
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
KoEr	A	L	N	P	Y	0	0
#M	K	W	N	P	Y	0	0
POwa	A	L	N	P	Y	0	0
BeAw	E	W	N	P	Y	0	0
EiBe	A	L	N	P	Y	0	0
SaNr	E	W	N	P	Y	0	0
BetQ	A	W	N	P	Y	0	0
AWPS	E	W	N	P	Y	0	0
Acbe	A	L	N	P	Y	0	0
Star	E	L	N	P	Y	0	0
End+	A	L	N	P	Y	0	0
Teln	E	L	N	P	Y	0	0
End-	A	L	N	P	Y	0	0
OvAw	E	W	N	P	Y	0	0
Scab	A	L	N	P	Y	0	0
OvWe	E	W	N	P	Y	0	0
Pose	A	L	N	P	Y	0	0
LäAw	E	W	N	P	Y	0	0
Saau	A	L	N	P	Y	0	0
LäWe	E	D	N	P	Y	0	0
STaQ	A	L	N	P	Y	0	0
FrMo	E	W	N	P	Y	0	0
Scau	A	L	N	P	Y	0	0
REW1	E	W	N	P	Y	0	0
Verw	A	L	N	P	Y	0	0
REW2	E	W	N	P	Y	0	0
MFkt	A	W	N	P	Y	0	0
FeLö	E	L	N	P	Y	0	0
FLA1	A	L	N	P	Y	0	0
REB1	E	L	N	P	Y	0	0
FLA2	A	L	N	P	Y	0	0
FLA3	A	L	N	P	Y	0	0
8TFL	A	W	N	P	Y	0	0
Pols	A	D	N	P	Y	0	0
FeNr	A	W	N	P	Y	0	0
RAW1	A	W	N	P	Y	0	0
RAB1	A	L	N	P	Y	0	0
RAB2	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	POKO			
00002		# 0		
00003		# 0		
00004		PP 0	FREI	Block enable
00005		PP 0	#M	Module No.
00006		PP 0	BeAw	Mode selection
00007		PP 0	SaNr	Set No.
00008		PP 0	AWPS	User program segment
00009		PP 0	Star	Start
00010		PP 0	Teln	Teach In
00011		PP 0	OvAw	Override selection
00012		PP 0	OvWe	Override value
00013		PP 0	LäAw	Length compensation selection
00014		PP 0	LäWe	Value of the length compensation
00015		PP 0	FrMo	Enable mode
00016		PP 0	REW1	reserved
00017		PP 0	REW2	reserved
00018		PP 0	FeLö	Delete error number
00019		PP 0	REB1	reserved
00020		PP 0	KoEr	Kom Error
00021		PP 0	POwa	POKO waiting
00022		PP 0	EiBe	Input limiting
00023		PP 0	BetQ	Quit modes
00024		PP 0	Acbe	Axis ready
00025		PP 0	End+	End position +
00026		PP 0	End-	End position -
00027		PP 0	Scab	Trailing distance
00028		PP 0	Pose	Position reached
00029		PP 0	Saau	Set executed
00030		PP 0	StaQ	Start acknowledgement
00031		PP 0	Scau	Loop executed
00032		PP 0	Verw	Dwell time
00033		PP 0	MFkt	Maschine function
00034		PP 0	FLA1	FLAG 1
00035		PP 0	FLA2	FLAG 2
00036		PP 0	FLA3	FLAG 3
00037		PP 0	8TFL	8 section flags
00038		PP 0	Pols	Actual position value
00039		PP 0	FeNr	Error number
00040		PP 0	RAW1	reserved
00041		PP 0	RAB1	reserved
00042		PP 0	RAB2	reserved

APPLICATION NOTES

Arrangement of POKO, POKANF and POKEND in the PLC program

-No more than 1 POKANF block call and 1 POKEND block call may be planned in each PLC program.
-Up to 256 POKO block calls may be planned in 1 PLC program. In doing so, each POKO must be assigned to a 35 PO 90 module with the aid of the module number (unit number).
-POKANF must be placed at the start of the PLC program, but in any case before all POKO blocks.
-If a PLC program part with a run time of more than 10 ms is placed between the call of the POKANF block and the first POKO block call, no PLC computing time is used up in POKO for waiting for the reaction of the 35 PO 90 module. If this waiting time occurs, it is added to the block 's normal run time. Refer also to the application note entitled "planning the PLC program cycle time".
-All POKO calls must be placed after the POKANF call and before the POKEND call.
-The PLC evaluation program for a specific 35 PO 90 module should be placed after the affiliated POKO block call and before the POKEND block call.
-POKEND must be placed at the end of the PLC program, but in any case after all POKO block calls.

PLC evaluation program for a 35 PO 90 module

-The PLC evaluation program for a 35 PO 90 module evaluates the 35 PO 90 MPST bus outputs which are available as POKO block outputs. After the POKO block has been called, the POKO block outputs are the current image of the 35 PO 90 MPST outputs. This does not apply in the event that communication between the PLC and 35 PO 90 is disturbed (KoEr = 1).
-Depending on the state of these block outputs (BetQ, Saau,...), by way of the POKO block outputs, the user can commission the 35 PO 90 module (e.g. select mode, start NC set ...).

Block enabling FREI

-Normally, (no commissioning, 35 PO 90 is controlled by only 1 active MPST station (e.g. PLC)), enabling of all POKO blocks should be set permanently.
-Resetting enabling of a block (FREI = 0) suppresses access by POKO, POKANF and POKEND to the 35 PO 90 module assigned to this POKO block.
-Resetting enabling of a POKO block always results in setting of the blocks communication error output (KoEr).
-Resetting enabling of a POKO block has no effect on communication between other POKO blocks and the 35 PO 90 modules assigned to them.
-Once the KoEr block output has been set and the corresponding block enable input has again assumed the value 1, the POKO block again establishes communication with the 35 PO 90 module and acknowledges establishment of communication with a new 1 -> 0 changeover of the output signal KoEr.

Evaluation of the KoEr block output

-After every PLC program start, the output KoEr = 1 is set (prerequisite: flag area of the PLC is cleared by the PLC when the PLC program is started, i.e. the INI OPS mode is set in the PLC).
-The 1 -> 0 changeover of the KoEr output after the POKO block call acknowledges the PLC <-> 35 PO 90 connection establishment.
-The KoEr output is set or reset exclusively by the corresponding POKO block. Any disturbances in communication by a POKANF or POKEND and a 35 PO 90 are buffered within the PLC until the corresponding POKO block is called and are then indicated by its output KoEr.
-If the KoEr output is again set by POKO after acknowledgement of connection establishment, then communication with the affiliated 35 PO 90 module is disturbed (e.g. 35 PO 90 malfunctioning, 35 PO 90 single board reset, block enable disabled, ...). The POKO block then attempts (prerequisite: block enabled) to again establish communication with the 35 PO 90 module and acknowledges this after establishment of the connection by a new 1 -> 0 changeover of the KoEr output. The reaction to a single board reset of a 35 PO 90 module can be planned in the PLC program. Therefore, it is not necessary to deactive and restart the PLC program.
-Evaluation of the KoEr signal and the required reaction to it can be planned by the user himself in his PLC program. This reaction generally depends on the momentary operating state and the system configuration etc. For example, the reaction may be emergency deactivation of the overall system or only the PLC program will be set to a specific state in which the system can continue to operate.

Planning the PLC program cycle time

-The PLC program cycle time to be planned is minimized by paying attention to the application note entitled "Arrangement of POKO, POKANF and POKEND in the PLC program".

-When set, the *POKO waiting output* (POwa) indicates that computing time is being used in the POKO block by waiting for permission to communicate with the 35 PO 90 module. That is to say, the processing time of the PLC program part between the POKANF and POKO blocks is less than 10 ms. This is the maximum time that may elapse between requesting of communication in a 35 PO 90 module and permission to communicate by the 35 PO 90 module. If this waiting time occurs, it is added to the block's normal run time. The waiting time in the POKO block is avoided by placing a PLC program part with a processing time of more than 10 ms between the POKANF and POKO blocks. This can be seen by virtue of the fact that the "POKO waiting" output (POwa) is not set.

Important: The POwa output of the block is set in latching mode.

Diagnosis outputs

-*POKO waiting* POwa

This output indicates that computing time is being used in the POKO block waiting for the 35 PO 90 to be ready for communication. Normally, this can be prevented by planning (by observing the application note entitled "Arrangement of POKANF, POKO and POKEND in the PLC program").

Problem: In the PLC program the POKANF and POKO blocks follow each other too quickly (interval less than 10 ms).

Remedy: Place further existing PLC program parts between POKANF and POKO until the POwa output is no longer set. The POwa output is set in latching mode and can only be reset as follows:

- By aborting and then starting the PLC program in INI OPS mode
- By occurrence of a communication error, i.e. KoEr = 1; for example, this communication error can also be provoked by means of a single board reset.
- Input limiting EiBe

This output indicates that a value of an input planned in the POKO block lay outside of the permissible value range and was limited by the POKEND block or set to the default before transfer to the 35 PO 90 unit. The EiBe output is set in latching mode and can only be reset as follows:

-By aborting and then starting the PLC program in INI OPS mode

-By occurrence of a communication error, i.e. KoEr = 1; for example, this communication error can also be provoked by means of a single board reset.

Block outputs: Start, Quit, StaQ

-By virtue of the block output StaQ = 1, the user recognizes that a 35 PO 90 set processing cycle was started in the previous PLC cycle. The output assumes the state 1 for one PLC cycle only. This also applies when processing of the set takes longer than one PLC cycle.

-If, after aborting of a positioning operation (e.g. by pressing the stop key on the operator control unit 35 AB 50) positioning is continued, the block's output StaQ is not set equal to 1. A positioning operation aborted by pressing the stop key is continued by pressing the start key and additionally by means of a start signal Star of the POKO block.

-Problem evaluating the 'set executed' Saau output of the block:

Only during processing of the POKEND block are the values of the inputs planned in the POKO block transferred to the 35 PO 90 module and, for example, 35 PO 90 set processing is started. In the next PLC program cycle the user recognizes the start of set processing, e.g. by Saau = 0. The user can recognize the end of set processing by virtue of the fact that Saau = 1. Now, if the set processing time is very short, (less than the PLC cycle time), the output Saau remains at 1. By virtue of the fact that the signal StaQ = 1, the user can recognize in this PLC cycle that processing of the set has been started and already concluded.

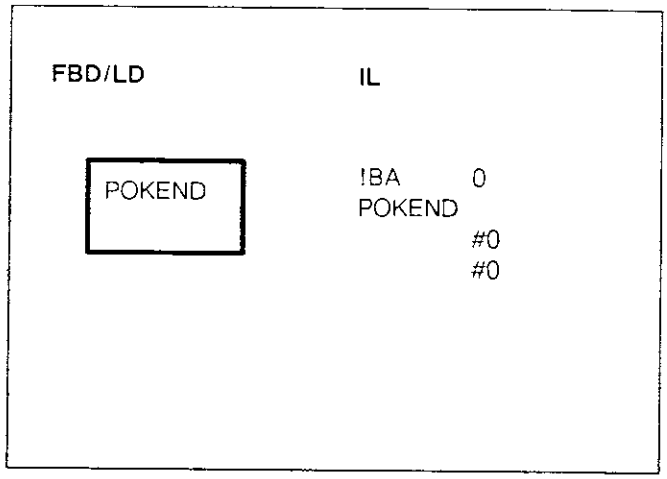
-It is therefore advisable to always evaluate both signals Saau and StaQ in the PLC evaluation program (see also application note, 907 PB 36; only in German).

Direct access to the 35 PO 90 by several active T300 stations

-Before a further active station (in addition to the PLC) may access a 35 PO 90 module, the FREI input of the POKO block must be set to 0 (see application note "block enable").

-The FREI inputs of the POKO blocks can be controlled directly by another active station by planning as "super global values".

The POKEND function block is an auxiliary communication block for the POKO block.



Parameters

#0	DIRECT CONSTANT	#0	Dummy parameter in the IL; here, the PLC's editing program PA enters the address of the first POKO block.
#0	DIRECT CONSTANT	#0	Dummy parameter in the IL; here, the PLC's editing program PA enters the pointer to the historical value of the first POKO block.

CE Data

Runtime:	
Basic runtime:	280 µs per unit 35 PO 90 used
Additional runtime:	none
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

The POKEND function block is an auxiliary communication block for the POKO block.

In every PLC cycle, the POKEND function block transfers the block inputs planned in the POKO to the MPST bus inputs of the 35 PO 90 module.

In doing so, the POKEND block directly accesses the inputs of the POKO block. This guarantees that the values transferred to the 35 PO 90 module are up to date. At the same time, the results of the PLC evaluation program located after the POKO block are still transferred to the 35 PO 90 module in the same program cycle.

The block ends communication requested by POKANF with all 35 PO 90 modules.

Example

FBD/LD

IL

```
!BA      0
POKEND
         #0
         #0
```

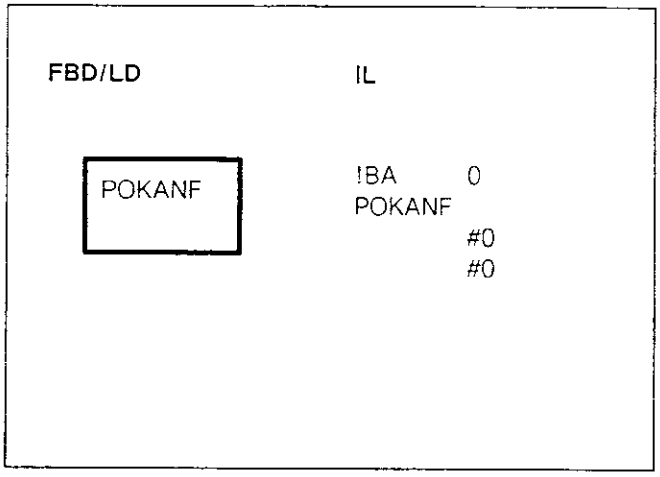
CE FBD Definition

Parameter list: not available

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	POKEND			
00002		# 0		
00003		# 0		

The function block POKANF is an auxiliary communication block for the POKO block (see POKO description).



Parameters

#0	DIRECT CONSTANT	#0	Dummy parameter in the IL; here, the PLC's editing program PA enters the address of the first POKO block.
#0	DIRECT CONSTANT	#0	Dummy parameter in the IL; here, the PLC's editing program PA enters the pointer to the historical value of the first POKO block.

CE Data

Runtime:	
Basic runtime:	80 µs per unit 35 PO 90 used
Additional runtime:	none
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

The function block POKANF is an auxiliary communication block for the POKO block (see POKO description).

After initialization, the POKANF function block requests communication with *all* planned 35 PO 90 modules for the POKO and POKEND blocks.

Example

FBD/LD

IL

```
!BA      0
POKANF
         #0
         #0
```

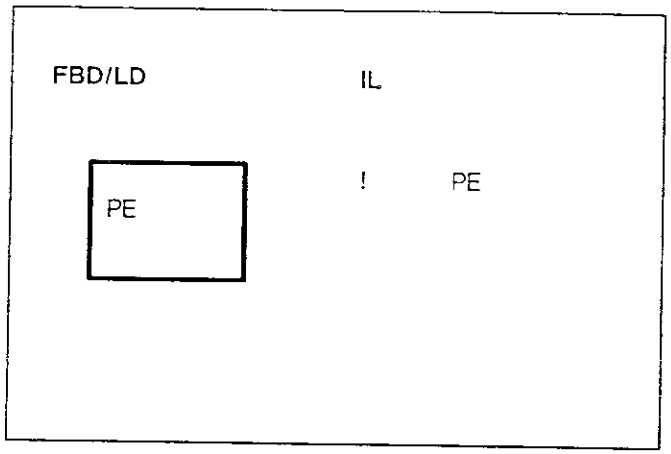
CE FBD Definition

Parameter list: not available

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	POKANF			
00002		# 0		
00003		# 0		

The program end identifies the end of the PLC program. Commands located after this end identifier are not processed by the PLC.



Parameters

CE Data

Runtime:	
Basic runtime:	2 µs
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The program end identifies the end of the PLC program. Commands located after this end identifier are not processed by the PLC.

Example

FBD/LD

IL

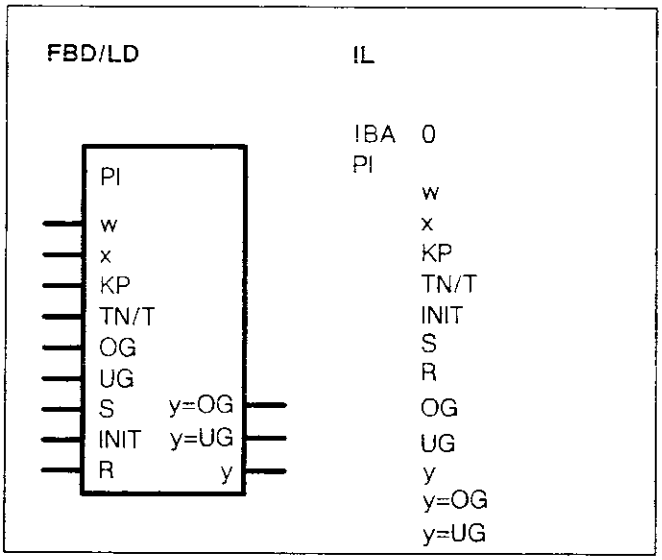
```
!        PE
```

CE FBD Definition

CE IL Definition

```
00000    !PE
```

The PI controller changes the value at its output y (manipulated variable) until the value at the input x (controlled variable) is equal to the value at the input w (command variable).



Parameters

w	WORD	EW, AW, MW, KW	Command variable (setpoint)
x	WORD	EW, AW, MW, KW	Controlled variable (actual value)
KP	WORD	EW, AW, MW, KW	Proportional coefficient; specified as a %
TN/T	WORD	EW, AW, MW, KW	Integral action time scaled to the PLC cycle time
OG	WORD	EW, AW, MW, KW	High limit for the manipulated variable y
UG	WORD	EW, AW, MW, KW	Low limit for the manipulated variable y
S	BINARY	E, A, M, S, K	Enabling for setting the manipulated variable y to the initial value INIT
INIT	WORD	EW, AW, MW, KW	Initial value for the manipulated variable y
R	BINARY	E, A, M, S, K	Reset of the manipulated variable y to the value 0
y=OG	BINARY	A, M	High limit has been reached
y=UG	BINARY	A, M	Low limit has been reached
y	WORD	AW, MW	Output for the manipulated variable y

CE Data

Runtime:

Basic runtime:	280 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	4 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701/R801 / 35 ZE 93 R101

Description

The PI controller changes the value at its output y (manipulated variable) until the value at the input x (controlled variable) is equal to the value at the input w (command variable).

Control algorithm: Simple rectangle rule

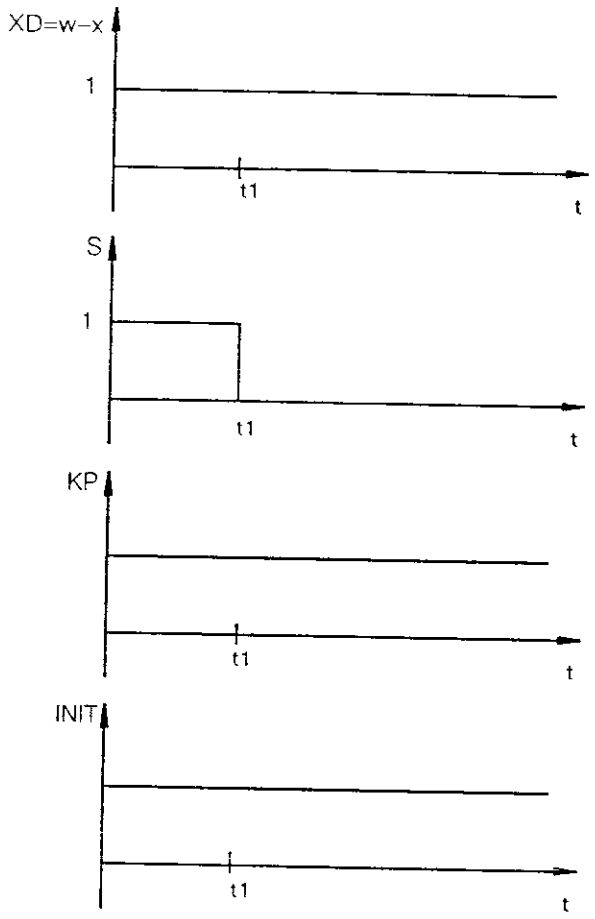
$$Y = \frac{KP}{100} * \frac{w-x}{TN/T} + YI(z-1) + \frac{KP}{100} * (w-x)$$

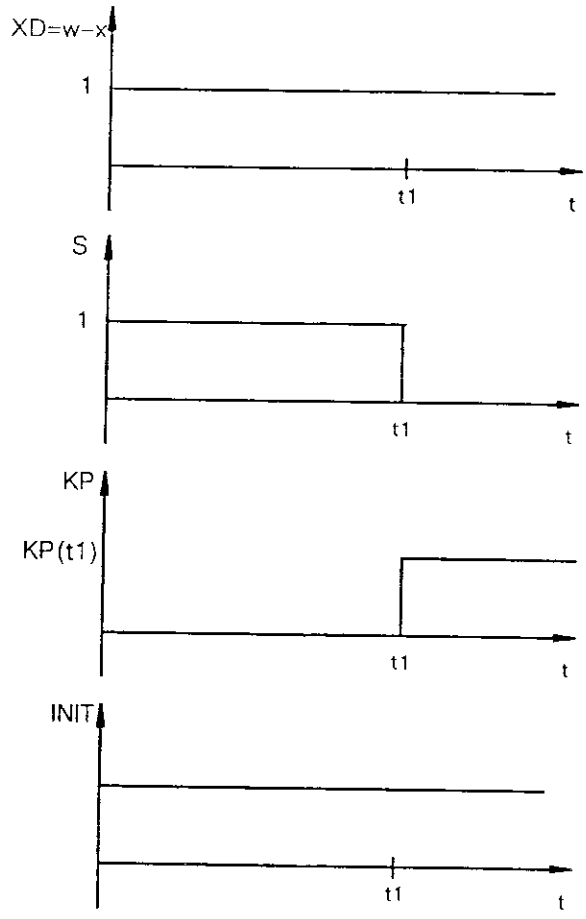
Where: YI(z-1) is the integral component from the previous program cycle

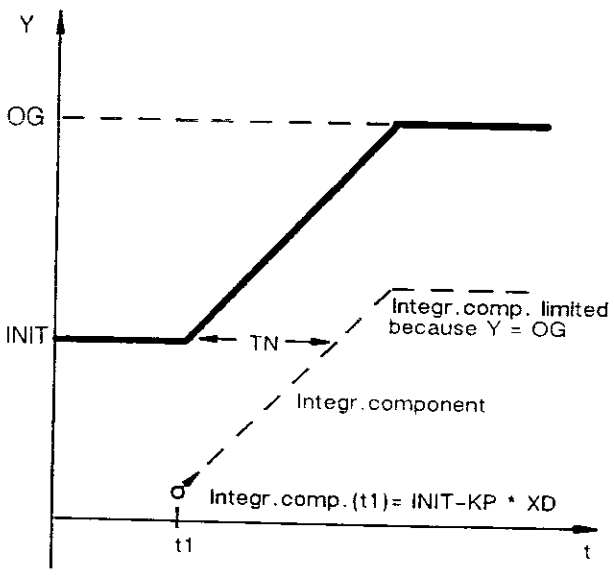
Transfer funktion:

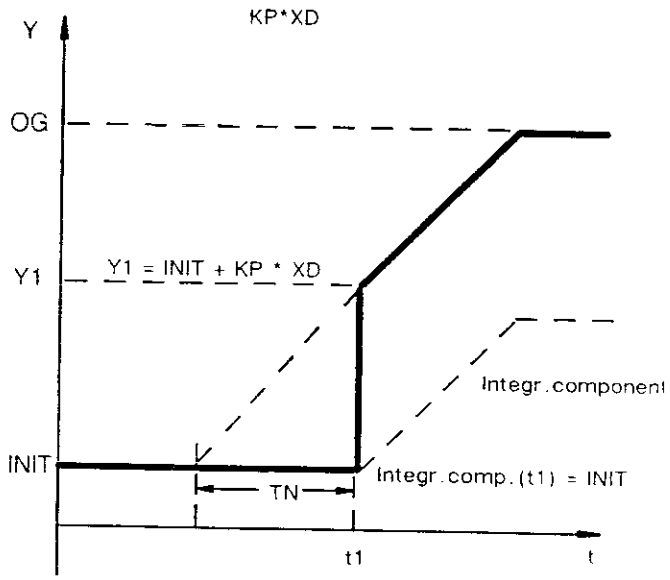
$$F(s) = KP * \left(1 + \frac{1}{s * TN}\right)$$

The inputs and outputs can neither be duplicated nor negated/inverted.









PI controller: Surge-free transition from the specified initial value to control operation

PI controller: Surging transition from the specified initial value to control operation

w WORD
The command variable (setpoint) is specified at the input w.

x WORD
The controlled variable (actual value) is specified at the input x.

KP WORD
The proportional coefficient is specified at the input KP. This value is specified as a percentage and may be positive or negative.

Example:

1	=	1	Percent
55	=	55	Percent
100	=	100	Percent
1000	=	1000	Percent
-500	=	-500	Percent

- 1 percent means that the block multiplies the system deviation by the factor 0.01 (see also control algoritm)

- 100 percent means that the block multiplies the system deviation by the factor 1 (see also control algoritm)

- 1000 percent means that the block multiplies the system deviation by the factor 10 (see also control algoritm)

Generally, proportional coefficients of more than 1000% are not meaningful in control systems.

TN/T WORD
The integral action time TN is scaled to the PLC cycle time T and is specified at the input TN/T.

Value range: $0 \leq TN/T \leq 328$

- If values are specified which are beyond the admissible value range the PLC uses the value 328.

- A large integral action time TN can be achieved by choosing a great cycle time T, too. If the block is used within a run number block, the cycle time of the run number block is valid for block PI and not the cycle time (KD 0,0) of the PLC program.

Limiting the manipulated variable y

OG WORD
UG WORD
The output y (manipulated variable) of the controller can be limited

- To a maximum value by specifying a limit at the input OG (high limit);

- To a minimum value by specifying a limit at the input UG (low limit)

The high and low limits *also* apply to the controller's internal I component. That is to say, the I component can only assume values between the high and low limits. If the manipulated variable y reaches one of the two limits, the controller's I component is *no* longer altered. This prevents the I component from running amock in the event of limiting of the controller output y, assuming meaningless values from the point of view of control and, in certain circumstances, not returning to the operating range until after a very long time. This response of a controller is also referred to as a "special anti-reset windup measure".

Setting and resetting the controller

S BINARY
INIT WORD
R BINARY
Setting the controller to an initial value

- The output y of the controller is set to the initial value specified at the INIT input by means of a 1 signal at the input S (set).

- A 1 signal at the input R (reset) is equivalent to specifying the initial value 0 (see above).

Surge-free setting/resetting

- The output y of the controller is set to the initial value specified at the input INIT by means of a 1 signal at the binary input S (set).

- A 1 signal at the input R (reset) is equivalent to specifying the initial value 0.

  In doing so, adjustment to the initial value takes place internally in the controller. This adjustment is a shift of the controller output from the momentary value to the required initial value. Now, the controller continues operating from this initial value precisely as it would have done at the old operating point before the shift, i.e. without surges. The controller's I component is defined so that the sum of the P component and the I component just results in the initial value.

Advantage of surge-free operation:

- Control as from the new initial value is devoid of surges.

Disadvantage of surge-free operation:

- The following equation applies:
  I_component = INIT - P_component

  In certain circumstances, the I component is set to high values and may take very long before this "wrong" I component from the point of view of control is dissipated again.

Surging setting/resetting

- The output y of the controller is set to the initial value specified at the INIT input by means of a 1 signal at the input S (set).
- A 1 signal at the input R (reset) is equivalent to specifying the initial value 0.

In the event of surging setting or resetting of the controller the I component is set equal to the initial value. To do this, the P component must be suppressed during setting.

Where: I component = INIT

Surging setting to an initial value is achieved by the following measure during setting:

- Specifying the value 0 at the input KP.

This measure renders the P component inactive. The controller output y assumes the initial value during the set cycle.

The P component is enabled again after the set cycle. From the initial value, the controller output y jumps according to the P component of the controller.

Advantage of surging setting:

- The I component is not set to "wrong" values from the point of view of control.

Disadvantage of surging setting:

- No freedom from surging

y WORD
The controller's manipulated variable y is output through the output y.

y=OG BINARY
The output y=OG signals whether or not the value at the output y has exceeded the specified high limit.

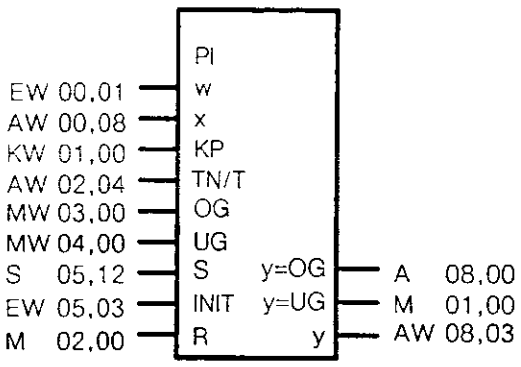
y=OG = 0 limit has *not* been reached.
y=OG = 1 limit has been reached.

y=UG BINARY
The output y=UG signals whether or not the value at the output y has reached the specified low limit.

y=UG = 0 limit has *not* been reached.
y=UG = 1 limit has been reached.

Example

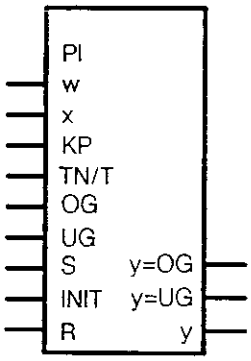
FBD/LD



IL

```
!BA   0
PI
      EW 00,01
      AW 00,08
      KW 01,00
      AW 02,04
      EW 05,03
      S  05,12
      M  02,00
      MW 03,00
      MW 04,00
      AW 08,03
      A  08,03
      M  01,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
w	E	W	N	P	Y	0	0
x	E	W	N	P	Y	0	0
KP	E	W	N	P	Y	0	0
TN/T	E	W	N	P	Y	0	0
OG	E	W	N	P	Y	0	0
UG	E	W	N	P	Y	0	0
S	E	L	N	P	Y	0	0
INIT	E	W	N	P	Y	0	0
R	E	L	N	P	Y	0	0
y=OG	A	L	N	P	Y	0	0
y=UG	A	L	N	P	Y	0	0
y	A	W	N	P	Y	0	0

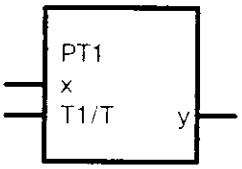
CE IL Definition

```
00000   IBA   0        Nr     Block No. (preset to 0)
00001   PI
00002         PP 0     w      Input  WORD (command variable)
00003         PP 0     x      Input  WORD (controlled variable)
00004         PP 0     KP     Input  WORD (proportional coefficient)
00005         PP 0     TN/T   Input  WORD (integral action time)
00006         PP 0     INIT   Input  WORD (initial value)
00007         PP 0     S      Input  BIT    (set to INIT)
00008         PP 0     R      Input  BIT    (reset to 0)
00009         PP 0     OG     Input  WORD (high limit)
00010         PP 0     UG     Input  WORD (low limit)
00011         PP 0     y      Output BINARY (manipulated variable)
00012         PP 0     y=OG   Output BINARY (OG reached)
00013         PP 0     y=UG   Output BINARY (UG reached)
```

PT1 ELEMENT

This function block realizes a delay element of the first order.

FBD/LD



IL

```
IBA   0
PT1
      x
      T1/T
      y
```

Parameters

x	WORD	EW, AW, MW, KW	Controlled variable
T1/T	WORD	EW, AW, MW, KW	Time constant
y	WORD	AW, MW	Manipulated variable

CE Data

Runtime:	
Basic runtime:	106 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	2 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

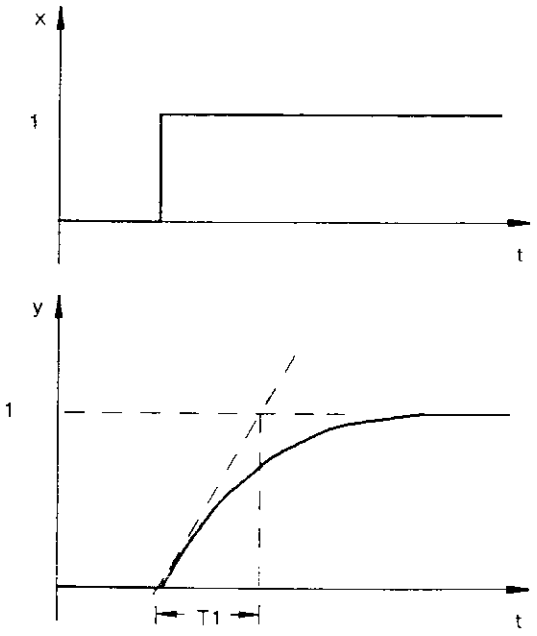
This function block realizes a delay element of the first order.

Transfer function:

$$F = \frac{1}{1 + T1 * s}$$

The inputs and the output can neither be duplicated nor negated.

Transfer funktion:



x WORD
The controlled variable is specified at the input x.

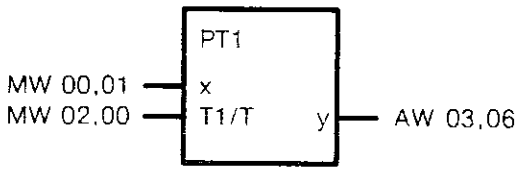
T1/T WORD
The delay time T1 is specified at the input T1/T. At the same time, the delay time T1 must be scaled to the cycle time T.

T1/T $\geq$ 0 must apply
If a negative time value is specified erroneously, the PLC automatically sets the value 32767 for T1/T.

y WORD
The result of the delay element (manipulated variable) is output through the output y.

Example

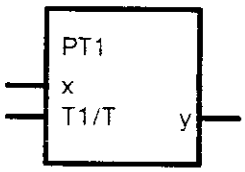
FBD/LD



IL

```
!BA   0
PT1
      MW 00,01
      MW 02,00
      AW 03,06
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
x	E	W	N	P	Y	0	0
T1/T	E	W	N	P	Y	0	0
y	A	W	N	P	Y	0	0

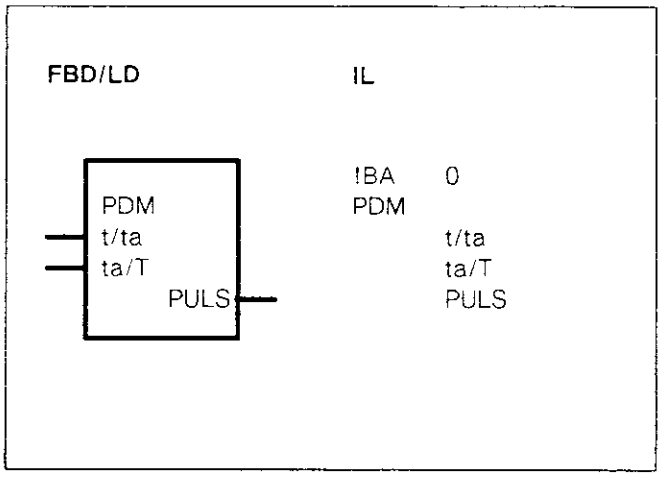
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	PT1			
00002		PP 0	x	Input WORD (controlled variable)
00003		PP 0	T1/T	Input WORD (time constant)
00004		PP 0	y	Output WORD (manipulated variable)

PULSE DURATION MODULATOR PDM

This function block generates a pulse duration-modulated binary signal at its PULS output.

The duty ratio is specified at the t/ta input and the period for the output signal is specified at the ta/T input.



Parameters

t/ta	WORD	EW, MW, AW, KW	Duty ratio
ta/T	WORD	EW, MW, AW, KW	Period referred to the cycle time
PULS	BINARY	A, M	Pulse duration modulated signal

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	3 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block generates a pulse duration-modulated binary signal at its PULS output.

The duty ratio is specified at the t/ta input and the period for the output signal is specified at the ta/T input.

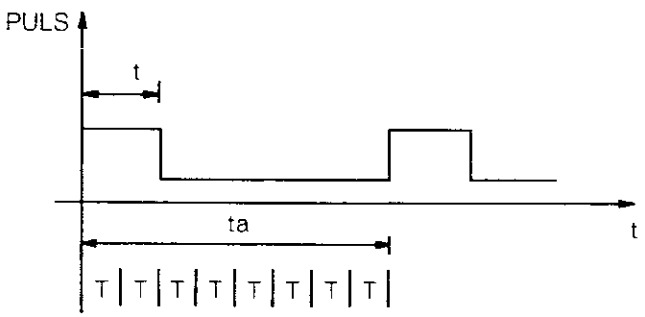
The inputs and the output can neither be duplicated nor negated nor inverted.

t/ta WORD Duty ratio

The required duty ratio for the output signal PULS is specified at the input t/ta. At the same time, ta is the period of the signal at the output PULS and t is the time within the period ta during which the output signal assumes a 1 level. The specified value for the required duty ratio at the input t/ta must be specified in scaled form. To do this, the required duty ratio must be multiplied by the value 32767 and rounded to a whole number. The resulting numerical value is then specified at the input t/ta.

Marginal condition for t: t > T

That is to say, the required duty cycle of the output signal must be higher than the cycle time of the PLC program.



The following relationship applies to specification of the keying ratio at the input t/ta:

Scaled value at the t/ta input	Results in duty ratio at the output PULS
Negative value	0 (0 %)
0 (0 * 32767)	0 (0 %)
.	. .
.	. .
16384 (0,5 * 32767)	0,5 (50 %)
.	. .
.	. .
32767 (1 * 32767)	1 (100 %)

ta/T WORD

The required period ta for the signal at the output PULS is specified at the input ta/T. At the same time, the period ta must be scaled to the cycle time T.

Marginal condition for ta:
- ta must be an integral multiple of T (ta = n * T)
- ta >> T > 0; the higher ta is in relation to T, the more exactly the required duty ratio is kept to
  E.g. ta ≥ 10 * T -> inaccuracy of the duty ratio at the output PULS ≤ 10%.

If a value ta/T < 0 is specified for ta/T, the function block automatically replaces this meaningless value by 32767.

PULS BINARY

The pulse duration modulated signal is available at the PULS output.

Combination of the PDM block with a controller

If the PDM function block is connected to the output of a controller in order to realize a "switching" controller, the following marginal conditions apply:

- Period ta of the PDM = sampling time of the controller
- Period of the pulse signal must be synchronous with the period of the controller's sampling time.

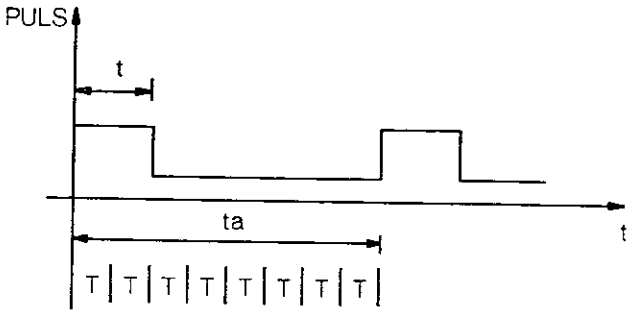
These marginal conditions are fulfilled by planning the controller in the same PLC program as the PDM, but within one run number block. By means of the run number block, the sampling time of the controller is prolonged by an integral multiple of the cycle time. Therefore, the controller is processed less frequently within the run number block than the PDM outside of the run number block.

Example:

Required:
- Duty ratio: t/ta = 0,25 (25 %)
- Period: ta = 800 ms (only an integral multiple of the PLC cycle time is possible)
- Cycle time: T = 100 ms

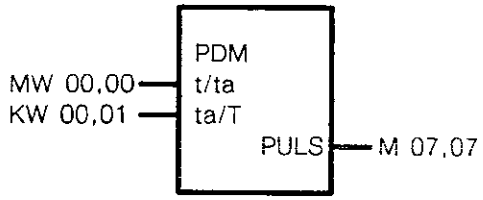
Block parameters to be specified:

- Value at the input t/ta: 8192 (0,25 * 32767)
- Value at the input ta/T: 8 (800 ms/100 ms)



Example

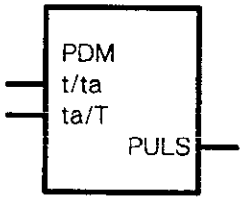
FBD/LD



IL

```
!BA   0
PDM
      MW 00,00
      KW 00,01
      M  07,07
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
t/ta	E	W	N	P	Y	0	0
ta/T	E	W	N	P	Y	0	0
PULS	A	L	N	P	Y	0	0

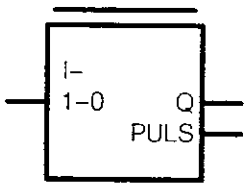
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	PDM			
00002		PP 0	t/ta	Input WORD (Duty ratio)
00003		PP 0	ta/T	Input WORD (period)
00004		PP 0	PULS	Output BINARY

A negative edge (1-0) at the input 1-0 generates a pulse at the output PULS which has the duration of 1 PLC program cycle.

IMPORTANT:
To some extent, functioning of this connection element did not correspond to the description. The functional scope has been modified as from 907 PC 31 Vers. 2.2 and now conforms to the description.

FBD/LD



IL

```
!     1-0
=S    Q
IN    1-0
&     Q
=R    Q
=     PULS
```

Parameters

1-0	BINARY	E, M, A, S, K	Input for 1 - 0 edge
Q	BINARY	M, A	Output for interrogation of the direct flag
PULS	BINARY	A, M	Pulse output

CE Data

Runtime:	
Basic runtime:	13.2 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

A negative edge (1-0) at the input 1-0 generates a pulse at the output PULS which has the duration of 1 PLC program cycle.

IMPORTANT:
To some extent, functioning of this connection element did not correspond to the description. The functional scope has been modified as from 907 PC 31 Vers. 2.2 and now conforms to the description.

Description of the error in 907 PC 31 Vers. 1.1 and older:
When the PLC program is activated (when the connection element is run through for the first time), a pulse is erroneously output through the PULS output.

1-0 BINARY

The value of the operand at this input is monitored for a 1-0 edge.

Q BINARY

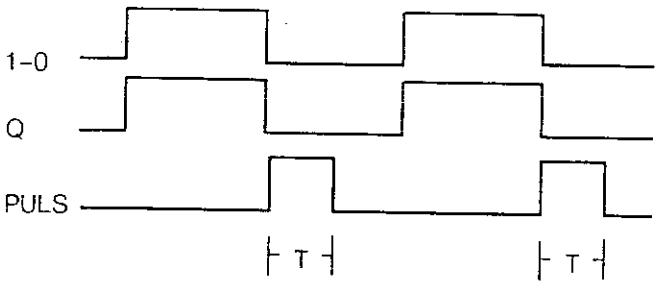
This additional auxiliary flag, which is specified as a direct flag above the connection element, is needed for edge detection. This flag must not be used again in the PLC program. It can be interrogated via the output Q. That is to say, the output Q can be connected to another connection element or a binary operand (A, M) can also be planned at the output Q.

PULS BINARY

A pulse is present at this output for 1 program cycle time if a 1-0 edge occurs at the input 1-0.

Duration of the pulse

From recognition of the 1-0 edge by the connection element up to renewed processing of this connection element in the next program cycle.

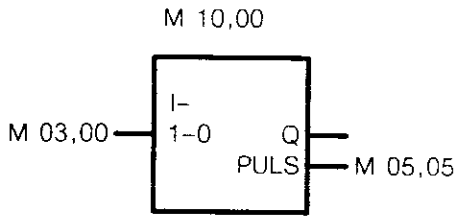


T: Program cycle time of the PLC

The input and the ouputs can neither be duplicated nor inverted.

Example

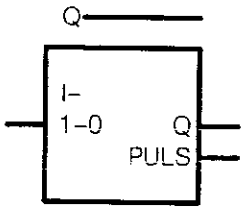
FBD/LD



IL

```
!      M 03,00
=S     M 10,00
!N     M 03,00
&      M 10,00
=R     M 10,00
=      M 05,05
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
1–0	E	L	N	P	Y	0	0
Q	A	L	N	P	Y	0	0
PULS	A	L	N	P	Y	0	0

CE IL Definition

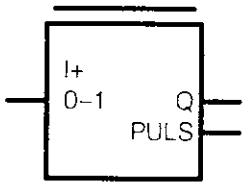
00000	!	PP 0	1–0	Input BINARY (1–0 edge)
00002	=S	PP 0	Q	Auxiliary flag BINARY
00004	!N	PP 0	1–0	Input BINARY (1–0 edge)
00006	&	PP 0	Q	Auxiliary flag BINARY
00008	=R	PP 0	Q	Auxiliary flag BINARY
00010	=	PP 0	PULS	Output BINARY (pulse)

PULSE (POSITIVE EDGE)

I+

A positive edge at the input 0-1 generates a pulse with the duration of one PLC program cycle at the PULS output.

FBD/LD



IL

```
!N    0-1
=R    Q
!     0-1
&N    Q
=S    Q
=     PULS
```

Parameters

0-1	BINARY	E, M, A, S, K	Input for 0 - 1 edge
Q	BINARY	A, M	Output for interrogation of the direct flag
PULS	BINARY	A, M	Pulse output

CE Data

Runtime:	
Basic runtime:	13.2 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

A positive edge at the input 0-1 generates a pulse with the duration of one PLC program cycle at the PULS output.

0-1 BINARY

The value of the operand at this input is monitored for a 0-1 edge.

Q BINARY

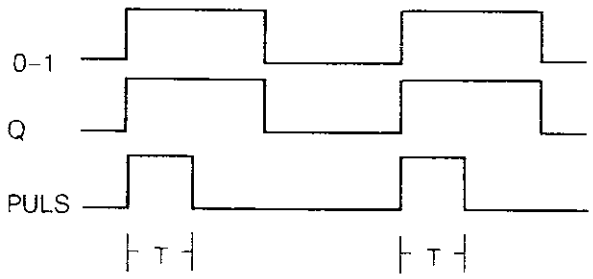
This additional auxiliary flag, which is specified as a direct flag above the connection element, is needed for edge detection. This flag must not be used again in the PLC program. It can be interrogated via the output Q. That is to say, the output Q can be connected to another connection element or a binary operand (A, M) can also be planned at the output Q.

PULS BINARY

A pulse is present at this output for 1 program cycle time if a 0-1 edge occurs at the input 0-1.

Duration of the pulse

From recognition of the 0-1 edge by the connection element until renewed processing of this connection element in the next program cycle.

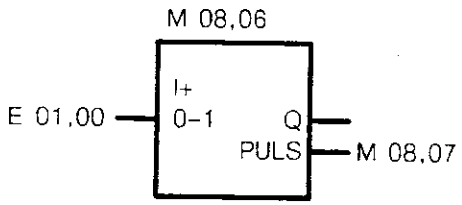


T: Program cycle time.

The input and the outputs can neither be duplicated nor inverted.

Example

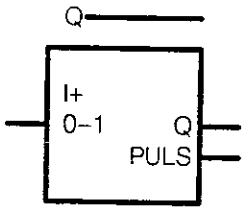
FBD/LD



IL

```
!N      E 01,00
=R      M 08,06
!       E 01,00
&N      M 08,06
=S      M 08,06
=       M 08,07
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0-1	E	L	N	P	Y	0	0
Q	A	L	N	P	Y	0	0
PULS	A	L	N	P	Y	0	0

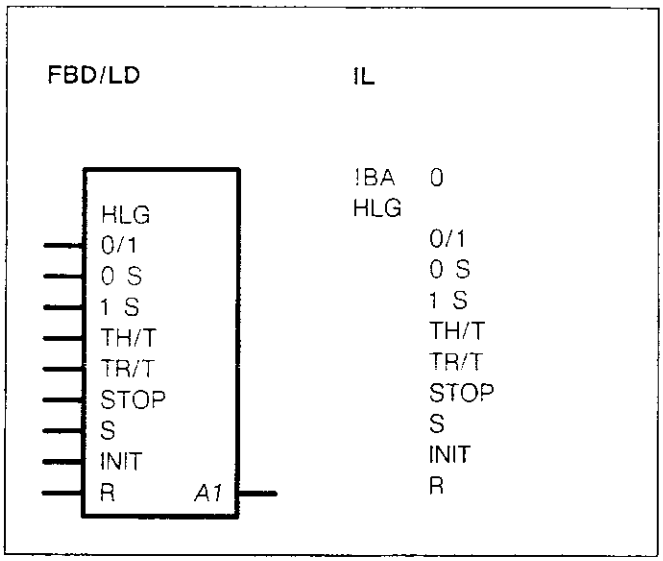
CE IL Definition

00000	!N	PP 0	0-1	Input BINARY (0-1 edge)
00002	=R	PP 0	Q	Auxiliary flag BINARY
00004	!	PP 0	0-1	Input BINARY (0-1 edge)
00006	&N	PP 0	Q	Auxiliary flag BINARY
00008	=S	PP 0	Q	Auxiliary flag BINARY
00010	=	PP 0	PULS	Output BINARY (pulse)

The ramp-function generator serves to provide ramp-shaped adaption of the current actual value at the output to a specified setpoint.

The value at the output of the HLG is adapted linearly from the current actual value to the specified setpoint with the slope y'.

In doing so, the value at the output precisely runs through the amount of the setpoint during the time TH or TR. If the value at the output of the HLG has reached the setpoint, it no longer changes unless a new setpoint is specified.



Parameters

0/1	BINARY	E, A, M, S, K	Selection of the setpoint 0 or the setpoint 1
0 S	WORD	EW, AW, MW, KW	Setpoint 0
1 S	WORD	EW, AW, MW, KW	Setpoint1
TH/T	WORD	EW, AW, MW, KW	Start up time scaled to the cycle time
TR/T	WORD	EW, AW, MW, KW	Return time scaled to the cycle time
STOP	BINARY	E, A, M, S, K	Latching of the output at the current value
S	BINARY	E, A, M, S, K	Setting of the output to the INIT value
INIT	WORD	EW, AW, MW, KW	INIT value to which the output can be set
R	BINARY	E, A, M, S, K	Reset of the output to the value 0
A1	WORD	MW, AW	Output

CE Data

Runtime:	
Basic runtime:	126 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	2 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701/R801 / 35 ZE 93 R101

Description

The ramp-function generator serves to provide ramp-shaped adaption of the current actual value at the output to a specified setpoint.

The value at the output of the HLG is adapted linearly from the current actual value to the specified setpoint with the slope y'.

In doing so, the value at the output precisely runs through the amount of the setpoint during the time TH or TR. If the value at the output of the HLG has reached the setpoint, it no longer changes unless a new setpoint is specified.

The inputs and the output can neither be duplicated nor negated/inverted.

The slope y' of the ramp results from the specified time TH (start up time) or TR (return time) and the amount of the setpoint:

$$\text{Slope } y' = \frac{\text{Setpoint amount}}{\text{TH or TR}}$$

The slope is

- positive, if setpoint > actual value
- negative, if setpoint < actual value
- 0 if setpoint = 0

Therefore, the specified setpoint has two functions:

- its amount defines the slope of the ramp together with the specified time TH or TR;
- it represents the value to which the current actual value must be adapted in a ramp shape.

The user may specify the start up time TH and the return time TR separately. The direction of the slope is defined on the basis of the setpoint. The direction of the slope then defines whether or not the running time TH or TR is used.

Slope y' positive -> TH, i.e. the ramp runs upwards
Slope y' negative -> TR, i.e. the ramp runs downwards.

The start up time TH and the return time TR must be scaled to the program cycle time TZ, i.e. the following must be specified at the corresponding inputs of the block:

- Start up time: TH/TZ
- Return time: TR/TZ

The times are specified in milliseconds. The following applies to the time constants TH or TR:
$0 \leq TH \leq 32767$
$0 \leq TR \leq 32767$

Two setpoints can be planned (0 S and 1 S), whereby one of these setpoints is selected by the binary input 0/1 (setpoint selection).

The setpoints may assume the following values:
$-32767 \leq$ setpoint $\leq +32767$

At any time, the output of the ramp-function generator can be
- stopped at the current value
- set to an initial value
- reset (output = 0)

The STOP input has the highest priority and the R input has the lowest.

The values at the inputs of the HLG can be altered at any time in the user program. In this way, any (non-linear) adaptation to the setpoint can be realized on the basis of the linear adaptation of the actual value.

Important
Setpoint = 0 means that the slope of the ramp is also 0, i.e. the current actual value does not change. If it is intended to switch from an actual value unequal to 0 to an actual value of 0, a setpoint unequal to 0 must be specified and the output of the ramp-function generator must be limited to 0 by a subsequent limiter. (On interpolation, the rounding transitions are based on calculation of integral numbers only).

0/1 BINARY
One of the two setpoints is selected with the input 0/1.
0/1 = 0 -> setpoint 0 S
0/1 = 1 -> setpoint 1 S

0 S WORD
The setpoint 0 is specified at the input 0 S.

1 S WORD
The setpoint 1 is specified at the input 1 S.

TH/T WORD
The startup time is specified at the input TH/T. At the same time, the start up time TH must be scaled to the cycle time T.

TR/T WORD
The start up time is specified at the input TH/T. At the same time, the start up time TH must be scaled to the cycle time T.

STOP BINARY
The output can be latched to the current value by means of the STOP input.
STOP = 0 -> Output not latched
STOP = 1 -> Output is latched
The STOP input has higher priority than the inputs S and R.

S BINARY
With the input S, the output can be set to the initial value specified at the input INIT.
S = 0 -> Output is not set to initial value.
S = 1 -> Output is set to initial value.

INIT WORD
The initial value to which the output is to be set if required is specified at the input INIT.

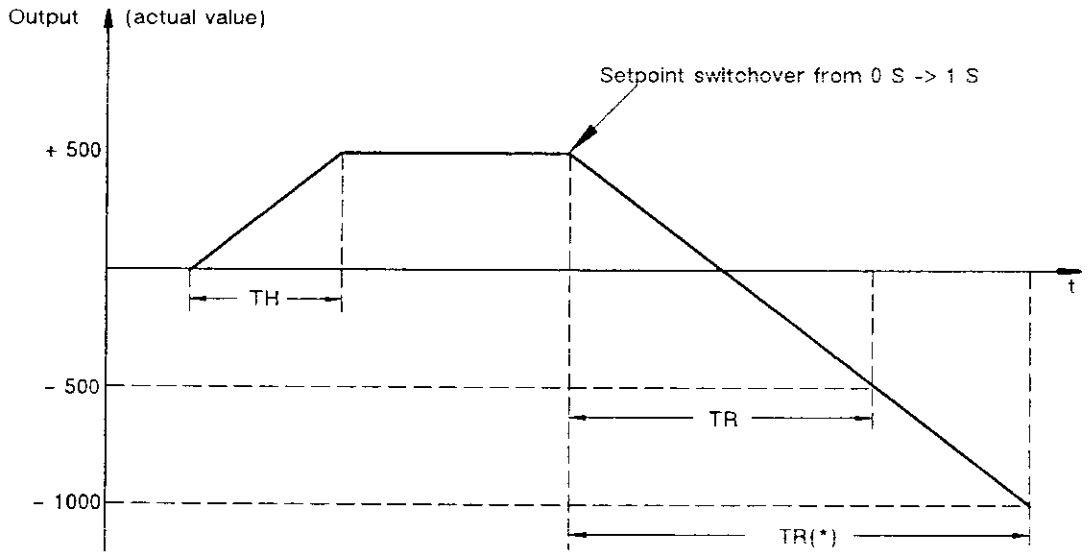
R BINARY
The output can be set to the value 0 with the input R.
R = 0 -> Output is not reset
R = 1 -> Output is reset to the value 0.

A1 WORD
Output of the block.

Example 1

The actual value is to be changed from 0 to the setpoint + 500 (setpoint amount 500) and then from +500 to the setpoint -1000 (setpoint amount 1000).

The start up time is TH and the return time TR.
0 S: 500
1 S: -1000



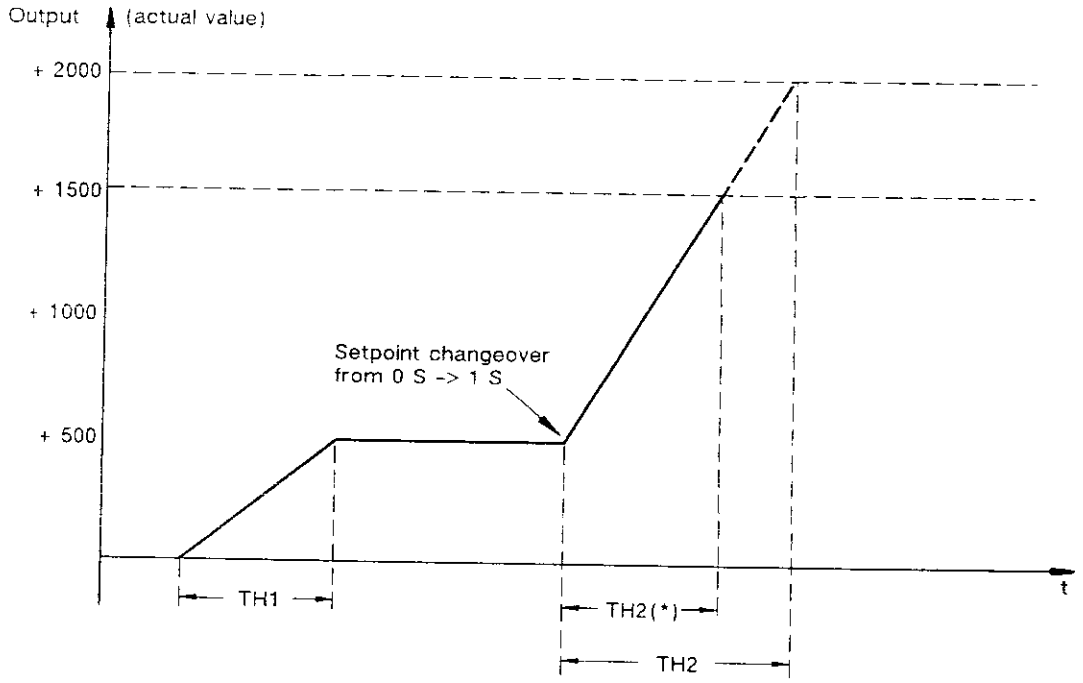
TR(*) is the actual time until the actual value has reached -1000.
During the time TH or TR, the actual value changes by the amount of the applied setpoint.

Example 2

The actual value is to be changed from 0 to the setpoint +500 (setpoint amount 500) and then from +500 to the setpoint 1500 (setpoint amount 1500).

The startup time is TH and the return time TR.

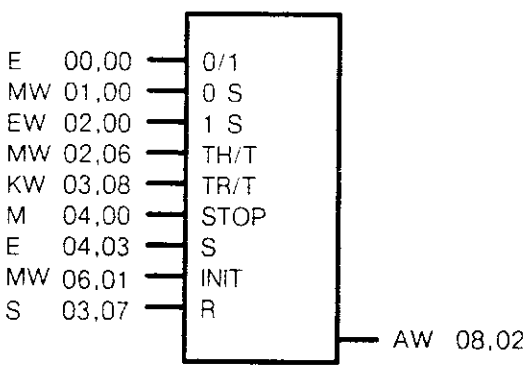
0 S: 500
1 S: 1500



TR(*) is the actual time until the actual value has reached 1500.
During the time TH or TR, the actual value changes by the amount of the applied setpoint.

Example

FBD/LD

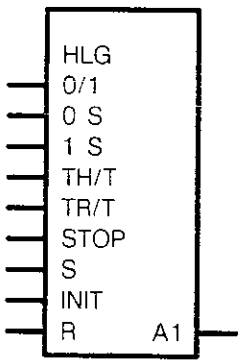


IL

```
!BA
HLG
```

```
E    00,00
MW   01,00
EW   02,00
MW   02,06
KW   03,08
M    04,00
E    04,03
MW   06,01
S    03,07
AW   08,02
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0/1	E	L	N	P	Y	0	0
0 S	E	W	N	P	Y	0	0
1 S	E	W	N	P	Y	0	0
TH/T	E	W	N	P	Y	0	0
TR/T	E	W	N	P	Y	0	0
STOP	E	L	N	P	Y	0	0
S	E	L	N	P	Y	0	0
INIT	E	W	N	P	Y	0	0
R	E	L	N	P	Y	0	0
A1	A	W	N	P	N	0	0

CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)
00001	HLG			
00002		PP 0	0/1	Input BIT (setpoint selection)
00003		PP 0	0 S	Input WORD (setpoint No. 0)
00004		PP 0	1 S	Input WORD (setpoint No. 1)
00005		PP 0	TH/T	Input WORD (start up time)
00006		PP 0	TR/T	Input WORD (return time)
00007		PP 0	STOP	Input BIT (stop ramp-function generator)
00008		PP 0	S	Input BIT (set to INIT)
00009		PP 0	INIT	Input WORD (initial value)
00010		PP 0	R	Input BIT (reset to 0)
00011		PP 0	A1	Output WORD

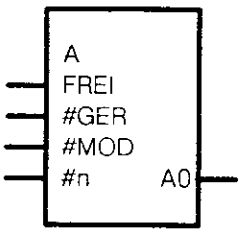
READ ANALOG VALUE FROM 35 EA 90

This function block serves to read 1...8 analog values from *one* analog input module 35 EA 90. 1...4 analog input modules 35 EA 90 can be plugged onto one carrier board 35 TP 90. *One* function block is needed for *each* analog input module.

The values of the analog channels read are allocated to the operands at the outputs A0...An–1.

CAUTION: Only as many output variables can be planned as channels are planned (1...8). Under no circumstances must more than one function block access one analog module, as otherwise malfunctions may occur.

FBD/LD



IL

```
!BA    0
ANAEIN
       FREI
       #GER
       #MOD
       #n
       A
```

Parameters

FREI	BINARY	E, M, A, K, S	Enabling for processing of the block
#GER	DIRECT CONSTANT	#, #H	Module address of the carrier board 35 TB 90
#MOD	DIRECT CONSTANT	#, #H	Number of the 35 EA 90 module (0 ... 3)
#n	DIRECT CONSTANT	#, #H	Number of assigned channels of the module 35 EA 90 (1...8)
A0	WORD	AW, MW	Output for the analog value of channel 1; the output can be duplicated

CE Data

Runtime:	
Basic runtime:	42 µs
Additional runtime:	48 µs per channel
Output updating:	yes if FREI = 1 and if the input module yields values continously
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block serves to read 1...8 analog values from *one* analog input module 35 EA 90. 1...4 analog input modules 35 EA 90 can be plugged onto one carrier board 35 TP 90. *One* function block is needed for *each* analog input module.

The values of the analog channels read are assigned to the operands at the outputs A0...An-1.

CAUTION: Only as many output variables may be planned as channels are also planned (1...8). Under no circumstances may more than *one* function block access *one* analog module as otherwise malfunctions may occur.

The output A0 can be duplicated. The inputs and outputs can be neither inverted nor negated.

The analog input module consists of:

- 1 basic board (carrier board 35 TP 90) and
- up to 4 analog modules (35 EA 90)

The block performs complete handling of *one* analog module of the analog input module, thus making plannig very simple for the user.

One analog module may only be operated by *one* block A as otherwise malfunctions may occur.

The user only needs to plan as many channels on an analog module as actually needed.

The block

- Successively activates the channels of an analog module planned by the user
- Reads the valid values
- Corrects the data format to 16 bit INTEGER (PLC format)
- and assigns the values to the variables specified at the block's outputs.

The waiting time for a valid analog value is limited to approximately 63 microseconds for each channel. If this time is exceeded, no value is allocated to the affiliated output of the block and the next channel is processed.

Value allocation at the block's outputs

Valid for: 935 PC 81, V3.0
(GJP5 1200 R701/R801)
935 PC 83, V5.0 (GJP5 1229 00 R201)

+10 V (+5 V) -> 7FF Hex = +2047 Decimal
0 V -> 0 Hex = 0 Decimal
-10 V (-5 V) -> 800 Hex = -2048 Decimal

Valid for: 935 PC 83, as of V6.0 (GJP5 1229 00 R301)

+10 V (+5 V) -> 7FF0 Hex = +32752 Decimal
0 V -> 0 Hex = 0 Decimal
-10 V (-5 V) -> 8010 Hex = -32752 Decimal

FREI BINARY

Processing of the function block is enabled with the signal at the FREI input.
FREI = 0 -> processing not enabled
FREI = 1 -> processing enabled

#GER DIRECT CONSTANT (#, #H)

The physical address of the analog input module (carrier board 35 TP 90) is specified at the #GER input. This is specified as a direct constant. Address parameters are generally specified as hexadecimal values.

#MOD DIRECT CONSTANT (#,#H)

Up to 4 analog modules 35 EA 90 can be plugged onto 1 carrier board. The modules have the numbers 0...3. The number of the analog module to be operated by the function block is specified at the #MOD input. This number is specified as a direct constant.

#n DIRECT CONSTANT (#,#H)

One analog module 35 EA 90 has 8 analog input channels. The number (1...8) of analog channels to be read by the function block is specified at the #n input. This number is specified as a direct constant. The following applies to n: $1 \leq n \leq 8$.

A0 WORD

The output A0 can be duplicated (A0...An-1). The analog values entered by the function block are allocated to the outputs A0...An-1. The following applies:
Channel 1 -> A0
Channel 2 -> A1
. .
. .
Channel 8 -> A7

CAUTION: Only as many outputs may exist as have been specified at the #n input.

Brief description of the analog input module

1. Analog input unit (35 TP 90, 35 EA 90)

The analog input unit consists of a carrier board (35 TP 90) and one to four analog modules (35 EA 90). Each of the analog modules (modules 0 to 3) has eight input channels and so the analog input unit has a total of 32 input channels if 4 modules are fitted.

Each of the 8 input channels of one analog module can be switched to the A/D converter via a multiplexer. The channel to be selected is specified by writing a control word into the analog module.

The unit is a passive station on the MPST BUS.

1.1 Conversion range

The conversion range can be adjusted by means of a jumper on the analog module.

Range 1: -10 V to +10 V
Range 2: - 5 V to + 5 V

1.2 Value allocation on the module 35 EA 90

The following affiliations apply between the analog value and the digital value:

+10 V (+5 V) -> 7FF Hex = +2047 Decimal
0 V -> 0 Hex = 0 Decimal
-10 V (-5 V) -> 800 Hex = -2048 Decimal

Note: As from PLC version V6, different affiliations apply at the outputs of the function block (see previous page).

1.3 Data format

On the input unit: 12 bit INTEGER (twos complement), right justified
Bit 15 = 0 -> Value valid
Bit 15 = 1 -> Value invalid

1.4 Conversion time

The time between activation of a channel and provision of the valid converted analog value is < 40 microseconds.

2. Settings of jumper panels

There are three jumper panels on the carrier board (35 TP 90) which have the following functions:

Jumper panel 1 (JF1): Connection of the module enable signals to the individual analog modules

Jumper panel 2 (JF2): Connection of the clock pulse to the individual analog modules

Jumper panel 3 (JF3): Setting of the unit address

There is a jumper on the analog module for setting the analog voltage range.

2.1 Setting the unit address

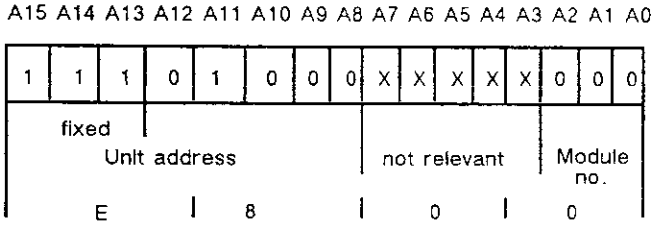
The unit address is set by means of jumper panel 3 (JF3).

The possible address range is from E0XXH ... FFXXH. Only the two most significant hex digits, i.e. A8...A15, are relevant to addressing the unit. The 3 most significant address bits A13...A15 are each permanently set to the value 1. The remaining 5 address bit A8...A12 can be set on jumper panel 3.

Affiliations between jumper and address bit on JF3:

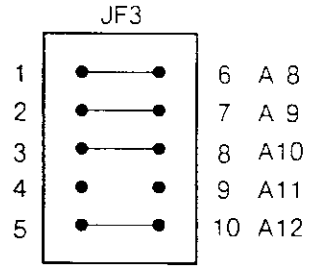
Address bit = 0 -> Jumper is fitted
Address bit = 1 -> Jumper is not fitted

MPST-BUS-Address:



Example:

Unit address E8XX corresponds to the following settings on JF3:

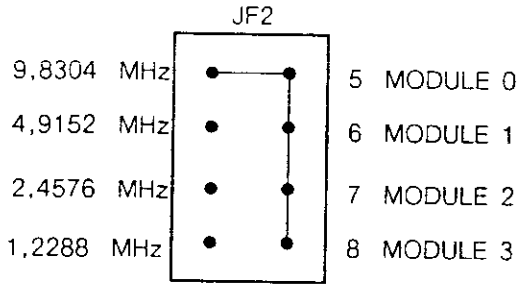


2.2 Wiring the clock pulse

Each analog module must be supplied with a clock pulse from the carrier board. To do this, one of the frequencies available on the basis board must be selected and switched through to *all* modules by means of jumper panel 2 (JF2).

The jumper panel must be wired as follows:

Frequency of 9.8304 MHz on all modules.



2.3 Module enabling

Each of the four analog modules is selected by two module enable signals each.

The module enable signals are generated from the address bits A0...A2 and are connected to the individual analog modules via jumper panel 1 (JF1). The following affiliations apply between the address bits A0...A2 and the modules:

MODULE	A0 ... A2
MODULE 0	0
MODULE 1	2
MODULE 2	4
MODULE 3	6

All jumpers must be fitted on jumper panel 1 to set the above.

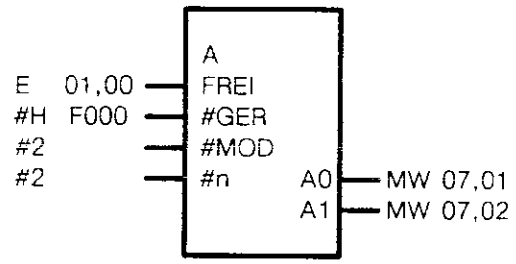
2.4 Setting the voltage range

The conversion range can be set by means of a jumper on the analog module.

Range 1: −10 V to +10 V -> Jumper 1–2
Range 2: − 5 V to + 5 V -> Jumper 2–3

Example

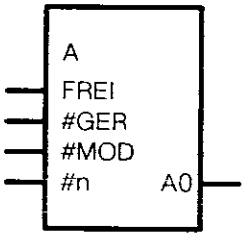
FBD/LD



IL

```
!BA          0
ANAEIN
        E    01,00
        #H   F000
        #2
        #2
        MW 07,01
        MW 07,02
```

CE FBD Definition



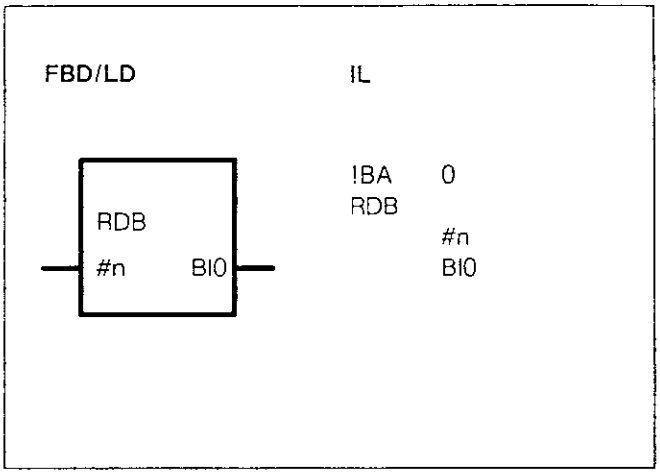
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
#GER	K	W	N	P	Y	0	0
#MOD	K	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
A	A	W	N	P	Y	1	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	ANAEIN			
00002		PP 0	FREI	Input BINARY (block enable)
00003		PP 0	#GER	# CONSTANT (module address)
00004		PP 0	#MOD	# CONSTANT (module No.)
00005		PP 0	#n	# CONSTANT (No. of channels)
	[	1		
00006		PP 1	A	Output WORD (analog value)
	]	1		

READ BINARY VALUES FROM HISTORICAL VALUES MEMORY — RDB

This function block reads binary values out of its historical values memory and allocates these to the outputs BI0...BIn-1. The binary values are written beforehand into the historical values memory by means of the WRB block.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of outputs BI0 ... BIn-1
BI0	BINARY	A, M	Output for the binary values, capable of duplication (BI0 ... BIn-1)

CE Data

Runtime:	
Basic runtime:	48 µs. The runtime of the WRB block is already contained.
Additional runtime:	14 µs per variable
Output updating:	yes
Number of historical values:	#n even -> #n/2 words; #n odd -> (#n + 1)/2 words
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block reads binary values out of its historical values memory and allocates these to the outputs BI0...BIn-1. The binary values are written beforehand into the historical values memory by means of the WRB block.

The RDB and WRB function blocks always occur in pairs. The WRB block writes the required values into the historical values memory. The RDB block reads them again.

#n DIRECT CONSTANT

The number of outputs BI0...BIn-1 is specified at the input #n. This is specified as a direct constant.

Note:

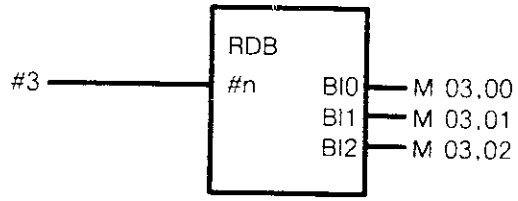
The value specified at the input #n must also agree with the number of inputs of the affiliated WRB block.

BI0...BIn-1 BINARY

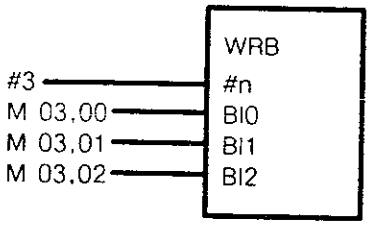
The output BI0 is capable of duplication (BI0...BIn-1). The block allocates the values read out of the historical values memory to the outputs BI0...BIn-1.

Example

FBD/LD



The program part, which is used multiply and in which the variables M 03.00 ... M 03.02 are needed in the next cycle, is located here. For this purpose, the values are written into the historical values memory by means of the WRB block and are read out again in the next cycle by means of the RDB block.



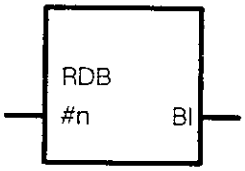
IL

```
IBA   0
RDB
      #3
      M 03,00
      M 03,01
      M 03,02
```

```
IBA   0
WRB
      # 0
      # 3
      M 03,00
      M 03,01
      M 03,02
```

READ BINARY VALUES FROM HISTORICAL VALUES MEMORY

CE FBD Definition



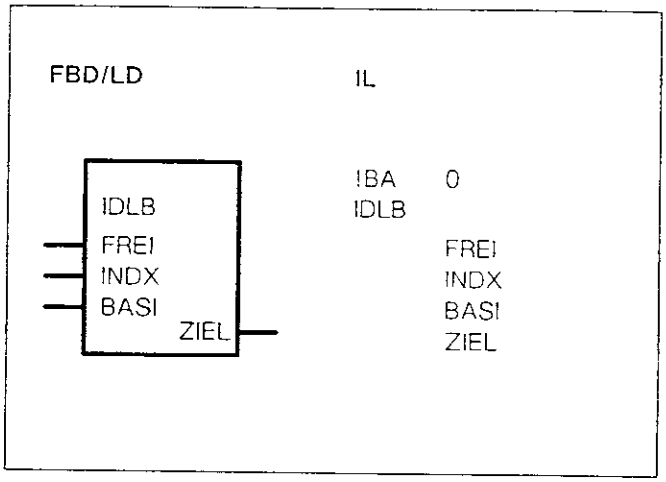
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
BI	A	L	N	P	Y	1	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	RDB			
00002		PP 0	#n	# DIRECT CONSTANT (number of bits)
	[	1		
00003		PP 1	BI	Output BINARY
	]	1		

READ BINARY VARIABLE, INDEXED — IDLB

This function block serves the purpose of indexed reading of binary variables.



Parameters

FREI	BINARY	E, A, M, K, S	Enable for the block Frei = 0: Block is not processed Frei = 1: The value of the source variable is read and allocated to the target variable.
INDX	WORD	EW, AW, MW, KW	The index and the basic variable result in the source variable
BASI	BINARY	E, A, M, K	Basic variable
ZIEL	BINARY	A, M	Target variable

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T300 V8.5

Description

This function block serves the purpose of indexed reading of binary variables.

The source variable to be read is obtained from indexing the basic variable. The value of the source variable read is allocated to the target variable.

The inputs and outputs can neither be duplicated nor inverted nor negated.

Note
EPROM programming in case of this block is not possible via the programming system. EPROM programming must be carried out in terminal emulation as operating function using command PU. At the same time the EPROM programming unit must be connected to serial interface 1.

The group and channel numbers of the source flag (source variable) are determined from the basic flag and the index INDX.

The source flag is:
M (G_Basis + A) . (K_Basis + B)

where:

G_Basis: Group number of the basic flag
K_Basis: Channel number of the basic flag

Formula:

$$\frac{INDX}{16} = A \quad \text{Remainder } B$$

Group No. of the source flag:
Group No. of the basic flag + A

Channel No. of the source flag:
Channel No. of the basic flag + B

Example:

Basic variable: M 00,00
INDX = 10 -> 10 : 16 = A = 0, Remainder B = 10
->Source variable:
M(00+A),(00+B) = M(00+0),(00+10) = M 00,10

Further examples

Basic variable		INDX	Source variable
M	00,00	0	M 00,00
M	00,00	2	M 00,02
M	00,00	16	M 01,00
M	00,02	18	M 01,04
E	00,00	3	E 00,03
E	01,01	5	E 01,06
A	05,05	6	A 05,11
K	00,04	8	K 00,12

FREI BINARY
Enable block

FREI= 0 -> Block is not processed
FREI= 1 -> The value of the source variable is read and allocated to the target variable ZIEL.

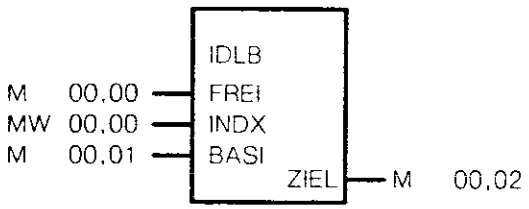
INDX WORD
The index value is specified at the input INDX. The source variable (see above for a calculation) results from the index INDX and the basic variable.
Value range: $-16383 \leq INDX \leq +16383$

BASI BINRAY
The basic variable is specified at the input BASI. The source variable (see above for a calculation) results from the index INDX and the basic variable.

ZIEL BINARY
The target variable is specified at the output ZIEL. The value of the selected source variable is allocated to the target variable ZIEL.

Example

FBD/LD



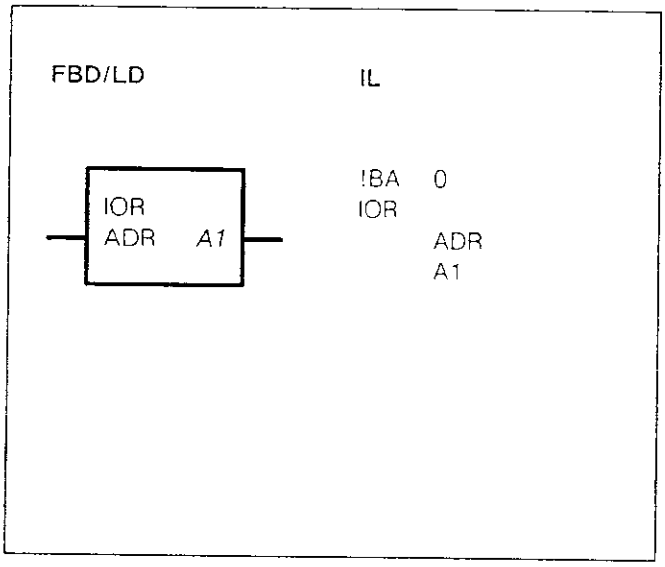
IL

```
!BA   0
IDLB
      M   00.00
      MW  00.00
      M   00.01
      M   00.02
```

READ BYTE VALUE FROM I/O ADDRESS

IOR

This function block reads a byte out of the I/O area and allocates it to the operand at the output.



Parameters

ADR	WORD	EW, AW, MW, KW	Address from the I/O area whose BYTE value is to be read.
A1	WORD	AW, MW	The BYTE value read is allocated to the output A1.

CE Data

Runtime:	
Basic runtime:	24 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block reads a byte out of the I/O area and allocates it to the operand at the output.

The input and output can neither be duplicated nor negated.

ADR WORD
The value of the operand at the input ADR represents the I/O address to be read.

A1 WORD
The byte read out of the I/O area is allocated to the LOW BYTE of the operand at the output A1.

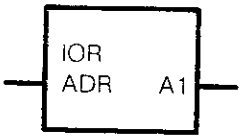
Example

FBD/LD

IL

```
!BA   0
IOR
      KW  00.00
      AW  02.00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
ADR	E	W	N	P	Y	0	0
A1	A	W	N	P	N	0	0

CE IL Definition

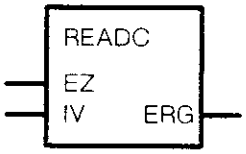
00000	!BA	0	Nr	Block No. (preset to 0)
00001	IOR			
00002		PP 0	ADR	Input WORD (I/O address)
00003		PP 0	A1	Output WORD (value)

READ COUNTER RESULTS (OMS-F) READC

This function block reads from an iconic processor 35 IV 90 the result values of all counters or the result value of a single counter including the counter status word.

This block is needed when using the video sensor OMS-F.

FBD/LD



IL

```
!BA   0
READC
      IV
      EZ
      ERG
```

Parameters

EZ	WORD	EW, AW, MW, KW	Number of the single counter
IV	WORD	EW, AW, MW, KW	Number of the 35 IV 90 module
ERG	WORD	AW, MW	Result values of the counters

CE Data

Runtime:	
Basic runtime:	115 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block reads from an iconic processor 35 IV 90 the result values of all counters or the result value of a single counter including the counter status word.

The status word contains the overflow flags of the counters belonging to the 35 IV 90 module. If the counters are previously set with event values (offsets), it is possible to derive from the status word the information as to which counters have elapsed (event occurrence). The mode register must be set at least 20 ms before calling the READC block, i. e. in the previous cycle.

EZ WORD

Number of the single window

EZ = 0: All windows of the window sequence are selected.
EZ = 1...8: Number of the window selected.
Value range: 0...8

IV WORD

Number of the 35 IV 90 module.

The number of the required 35 IV 90 module with which measurement is to take place is specified with the operand at this input.
Value range: 0...5.

ERG WORD

Result values of the counter

The result values of the counters belonging to the 35 IV 90 module and the status word for all counters of this module are provided at the block's ERG output. The result values are stored as from the operand specified at the ERG output. The status word is stored in the operand specified at ERG and the result values of the counters are stored in the operands with the consecutive numbering. That is to say:

Result of counter 1 in the first operand after the status word.
Result of counter 2 in the second operand after the status word.
Result of counter 3 in the third operand after the status word.

Result of counter 8 in the eighth operand after the status word

Important:
Therefore, apart from the operand specified at the ERG output, as many additional operands must be kept free in the program as the number of counters to be read.

Result values of a length or area measurement:
The result value is the total of all white or black pixels within the measurement window affiliated to the counter.

Result values in a histogram analysis:
The result value is the frequency with which specific gray tones occur within one measurement window.

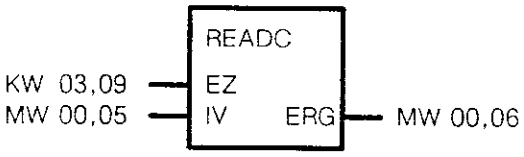
Structure of the status word:
The status word occupies bit positions 0...7 in the operand specified at the output ERG. Bit position 0 is assigned to counter 1 and bit position 1 is assigned to counter 2 etc.

Bit position	7	6	5	4	3	2	1	0
X X X X X X X X	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0

1: Counter has not overflown
0: Counter has overflown

Example

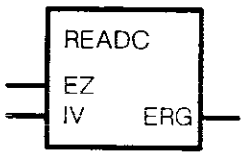
FBD/LD



IL

```
!BA    0
READC
       MW 00,05
       KW 03,09
       MW 00,06
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
EZ	E	W	N	P	Y	0	0
IV	E	W	N	P	Y	0	0
ERG	E	W	N	P	Y	0	0

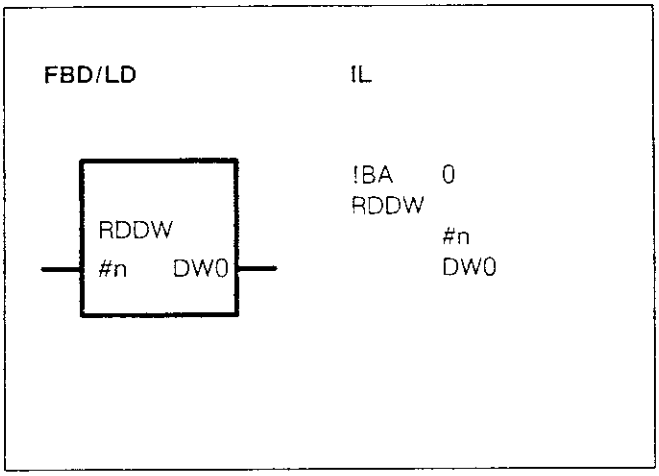
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	READC			
00002		PP 0	IV	Input WORD (IV No.)
00003		PP 0	EZ	Input WORD (counter number)
00004		PP 0	ERG	Output WORD (result)

READ DOUBLE WORD VALUES FROM HISTORICAL VALUES MEMORY

RDDW

This function block reads double word values out of its historical values memory and allocates them to the outputs DW0...DWn-1. Beforehand, the WRDW block writes the double word values into the historical values memory.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of outputs DW0 ... DWn-1
DW0	DOUBLE WORD	MD	Output for the double word values, capable of duplication (DW0 ... DWn-1)

CE Data

Runtime:	
Basic runtime:	34 µs. The runtime of the WRDW block is already contained here.
Additional runtime:	28 µs per additional output DW1 ... DWn-1
Output updating:	yes
Number of historical values:	2 * #n words
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block reads double word values out of its historical values memory and allocates them to the outputs DW0...DWn-1. Beforehand, the WRDW block writes the double word values into the historical values memory.

The RDDW and WRDW function blocks always occur in pairs. The WRDW block writes the required values into the historical values memory. The RDDW block reads them out again.

#n DIRECT CONSTANT

The number of the outputs DW0...DWn-1 is specified at the input #n. This is specified as a direct constant.

Note:

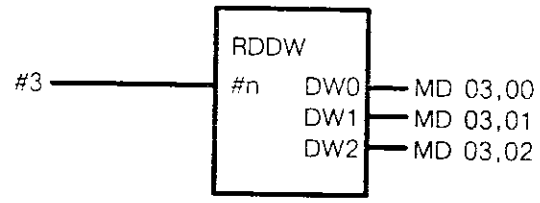
The value specified at the input #n must also agree with the number of inputs of the affiliated WRDW block.

DW0...DWn-1 DOUBLE WORD

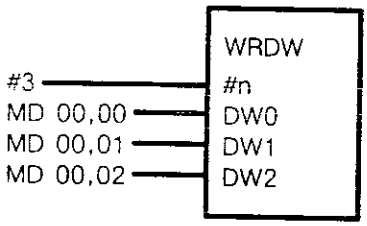
The DW0 output can be duplicated (DW0...DWn-1). The block allocates the values read out of the historical values memory to the outputs DW0...DWn-1.

Example

FBD/LD



The program part, which is used multiply and in which the variables MD 03.00 ... MD 03.02 are needed in the next cycle, is located here. For this purpose, the WRDW block writes the values into the historical values memory and the RDDW block reads them out again in the next cycle.



IL

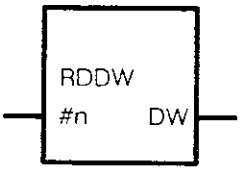
```
!BA    0
RDDW
       #   3
       MD 03,00
       MD 03,01
       MD 03,02
```

```
!BA    0
WRDW
       #   0
       #   3
       MD 00,00
       MD 00,01
       MD 00,02
```

READ DOUBLE WORD VALUES FROM HISTORICAL VALUES MEMORY

RDDW

CE FBD Definition



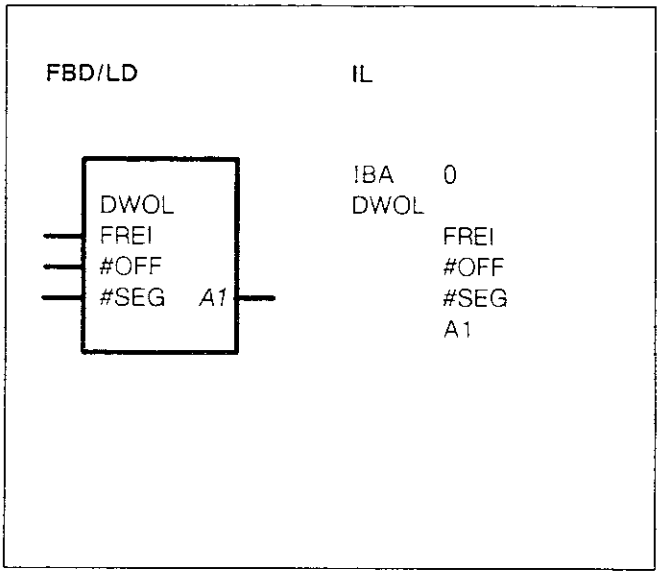
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
DW	A	D	N	P	Y	1	0

CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)
00001	RDDW			
00002		PP 0	#n	# CONSTANT (number of double words)
	[	1		
00003		PP 1	DW	Output DOUBLE WORD
	]	1		

READ DOUBLE WORD WITH ENABLE DWOL

When a 1 signal is present at the FREI input, the value of the specified physical address is read and is allocated to the operand at the output A1.



Parameters

FREI	BINARY	E, A, M, S, K	Block enable
#OFF	DIRECT CONSTANT	#, #H	Offset address of the memory location whose double word value is to be read.
#SEG	DIRECT CONSTANT	#, #H	Segment address of the memory location whose double word value is to be read.
A1	DOUBLE WORD	MD	Output to which the value read is allocated.

CE Data

Runtime:	
Basic runtime:	26.5 µs; 19.0 µs in case of no enable.
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

When a 1 signal is present at the FREI input, the value of the specified physical address is read and is allocated to the operand at the output A1.

No double word is read if there is a 0 signal at the FREI input.

The physical address consists of a segment and an offset.

FREI BINARY
Processing of the block is enabled or disabled with the operand at the input FREI. The following applies:
FREI = 0 -> Processing disabled
FREI = 1 -> Processing enabled

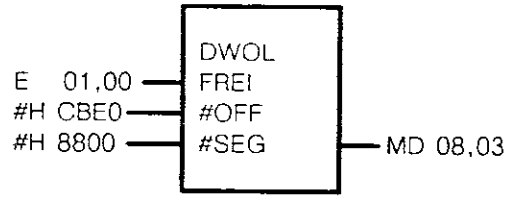
#OFF DIRECT CONSTANT (#, #H)
The offset component of the address to be read is specified at the input #OFF. This is specified as a direct constant.

#SEG DIRECT CONSTANT (#, #H)
The segment component of the address to be read is specified at the input #SEG. This is specified as a direct constant.

A1 WORD
The value read is allocated to the operand at the output A1.

Example

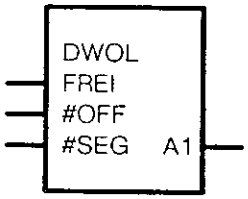
FBD/LD



IL

```
!BA    0
DWOL
       E   01,00
       #H CBE0
       #H 8800
       MD 08,03
```

CE FBD Definition



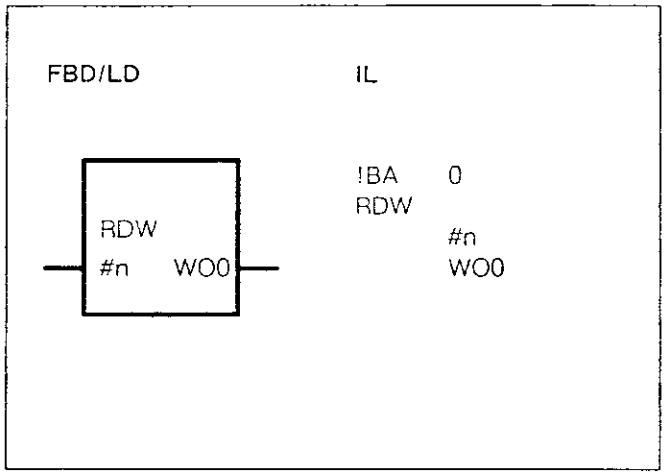
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
#OFF	K	W	N	P	Y	0	0
#SEG	K	W	N	P	Y	0	0
A1	A	D	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	DWOL			
00002		PP 0	FREI	Input BINARY (block enable)
00003		PP 0	#OFF	# CONSTANT (offset address)
00004		PP 0	#SEG	# CONSTANT (segment address)
00005		PP 0	A1	Output DOUBLE WORD (value)

READ WORD VALUES FROM HISTORICAL VALUES MEMORY — RDW

This function block reads word values out of its historical values memory and allocates these to the outputs WO0...WOn–1. The word values are written beforehand into the historical values memory by means of the WRW block.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of outputs WO0 ... WOn–1
WO0	WORD	AW, MW	Output for the word values, capable of duplication (WO0 ... WOn–1)

CE Data

Runtime:	
Basic runtime:	38 µs. The runtime of the WRW block is already contained here.
Additional runtime:	19 µs per additional output WO1 ... WOn–1
Output updating:	yes
Number of historical values:	#n words
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block reads word values out of its historical values memory and allocates these to the outputs WO0...WOn–1. The word values are written beforehand into the historical values memory by means of the WRW block.

The function blocks RDW and WRW always occur in pairs. The WRW block writes the required values into the historical values memory. The RDW block reads them out again.

#n DIRECT CONSTANT

The number of outputs WO0...WOn–1 is specified at the input #n. This is specified as a direct constant.

Note:

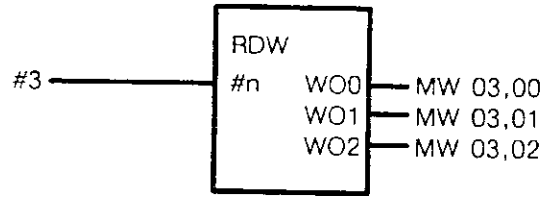
The value specified at the input #n must agree with the number of inputs of the affiliated WRW block.

WO0 ... WOn–1 WORD

The output WO0 can be duplicated (WO0...WOn–1). The block allocates the values read out of the historical values memory to the outputs WO0...WOn–1.

Example

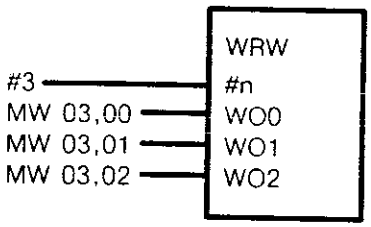
FBD/LD



IL

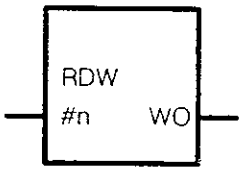
```
!BA     0
RDW
        #3
        MW 03,00
        MW 03,01
        MW 03,02
```

The program part which is used multiply and in which the variables MW 03.00 ... MW 03.02 are used in the next cycle is located here. For that the values are written into the historical values memory by the WRW block and are being read out by the RDW block in the next cycle.




```
!BA     0
WRW
        #    0
        #    3
        MW 03,00
        MW 03,01
        MW 03,02
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
WO	A	W	N	P	Y	1	0

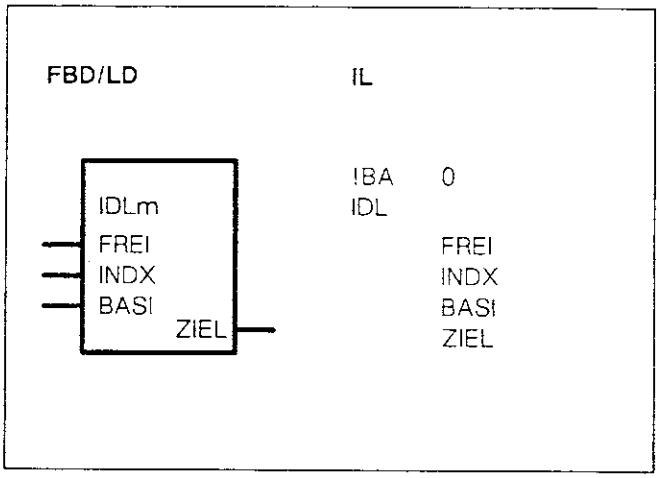
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	RDW			
00002		PP 0	#n	DIRECT CONSTANT
	[	1		
00003		PP 1	WO	Output
	]	1		

READ WORD VARIABLE, INDEXED

IDLm

This function block serves the purpose of indexed reading of word variables.



Parameters

FREI	BINARY	E, A, M, K, S	Enable for the block Frei = 0: Block is not processed Frei = 1: The value of the source variable is read and allocated to the target variable.
INDX	WORD	EW, AW, MW, KW	The index and the basic variable result in the source variable
BASI	WORD	EW, AW, MW, KW	Basic variable
ZIEL	WORD	AW, MW	Target variable

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block serves the purpose of indexed reading of word variables.

The source variable to be read is obtained from indexing the basic variable. The value of the source variable read is allocated to the target variable.

The inputs and outputs can neither be duplicated nor inverted nor negated.

The group and channel numbers of the source flag (source variable) are determined from the basic flag and the index INDX.

The source flag is:

MW (G_Basis + A) , (K_Basis + B)

where:

G_Basis: Group number of the basic flag
K_Basis: Channel number of the basic flag

Formula:

$$\frac{INDX}{16} = A \quad \text{Remainder } B$$

Group No. of the source flag:
Group No. of the basic flag + A

Channel No. of the source flag:
Channel No. of the basic flag + B

Example:

Basic variable: MW 00,00
INDX = 10 -> 10 : 16 = A = 0, Remainder B = 10
->Source variable:
MW(00+A),(00+B) = MW(00+0),(00+10) = MW 00,10

Further examples

Basic variable	INDX	Source variable
MW 00,00	0	MW 00,00
MW 00,00	2	MW 00,02
MW 00,00	16	MW 01,00
MW 00,02	18	MW 01,04
EW 00,00	3	EW 00,03
EW 01,01	5	EW 01,06
AW 05,05	6	AW 05,11
KW 00,04	8	KW 00,12

FREI BINARY
Enable block

FREI= 0 -> Block is not processed
FREI= 1 -> The value of the source variable is read and allocated to the target variable ZIEL.

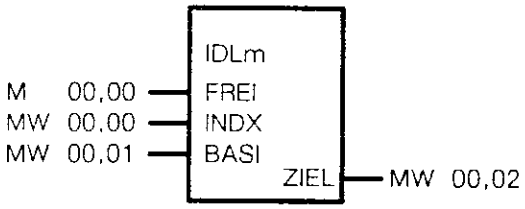
INDX WORD
The index value is specified at the input INDX. The source variable (see above for a calculation) results from the index INDX and the basic variable.
Value range: $-16383 \leq INDX \leq +16383$

BASI WORD
The basic variable is specified at the input BASI. The source variable (see above for a calculation) results from the index INDX and the basic variable.

ZIEL WORD
The target variable is specified at the output ZIEL. The value of the selected source variable is allocated to the target variable ZIEL.

Example

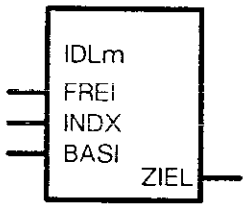
FBD/LD



IL

```
!BA   0
IDL
      M    00,00
      MW   00,00
      MW   00,01
      MW   00,02
```

CE FBD Definition



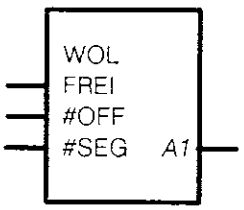
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
INDX	E	W	N	P	Y	0	0
BASI	E	W	N	P	Y	0	0
ZIEL	E	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	IDL			
00002		PP 0	FREI	Enable (BINARY)
00003		PP 0	INDX	Index relatet to basis (WORD)
00004		PP 0	BASI	Basic variable (WORD)
00005		PP 0	ZIEL	Target variable (WORD)

When the FREI input has a 1 signal, the value of the specified physical address is read and is allocated to the operand at the output A1.

FBD/LD



IL

```
!BA   0
WOL
      FREI
      #OFF
      #SEG
      A1
```

Parameters

FREI	BINARY	E, M, A, S, K	Enable block
#OFF	DIRECT CONSTANT	#, #H	Offset address of the memory location whose value must be read
#SEG	DIRECT CONSTANT	#, #H	Segment address of the memory location whose value must be read
A1	WORD	AW, MW	Output to which the read value is allocated.

CE Data

Runtime:	
Basic runtime:	20 µs, no reading; 33 µs, reading
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

When the FREI input has a 1 signal, the value of the specified physical address is read and is allocated to the operand at the output A1.

No reading and no allocation take place if the input FREI has a 0 signal.

The physical address consists of a segment and offset. Thus, the attainable address space is 1 MByte.

The inputs and the output can neither be duplicated nor negated.

FREI BINARY

Processing of the block is enabled or disabled with the operand at the FREI input. The following applies:
FREI = 0 -> Processing disabled
FREI = 1 -> Processing enabled

#OFF DIRECT CONSTANT (#,#H)

The offset of the address to be read is specified at the input #OFF. This is specified as a direct constant.

#SEG DIRECT CONSTANT (#,#H)

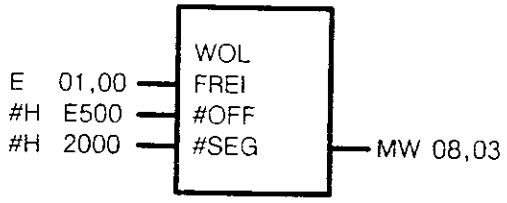
The segment of the address to be read is specified at the input #SEG. This is specified as a direct constant.

A1 WORD

The read value is allocated to the operand at the output A1.

Example

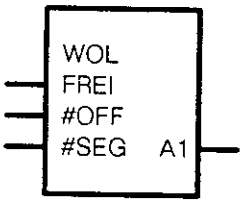
FBD/LD



IL

```
!BA   0
WOL
      E   01,00
      #H  E500
      #H  2000
      MW  08,03
```

CE FBD Definition



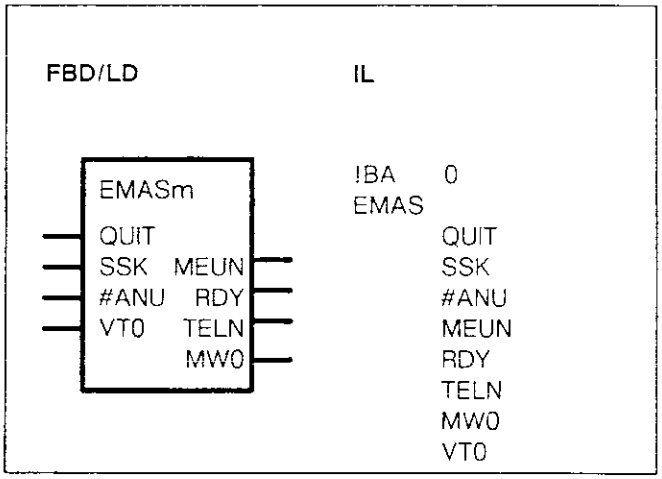
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
#OFF	K	W	N	P	Y	0	0
#SEG	K	W	N	P	Y	0	0
A1	A	W	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WOL			
00002		PP 0	FREI	Input BINARY (block enable)
00003		PP 0	#OFF	# CONSTANT (offset address)
00004		PP 0	#SEG	# CONSTANT (segment address)
00005		PP 0	A1	Output WORD (value)

The function block EMAS

- receives telegrams through a serial interface of the PLC
- compares these telegrams to comparison telegrams stored in the user program
- and, if these agree, provides the user data of the telegram received at the block's outputs.



Parameters

QUIT	BINARY	A, E, M, S, K	Reception of telegrams not enabled.
SSK	WORD	AW, EW, MW, KW	Interface identifier; number of the serial interface
#ANU	DIRECT		Number of outputs for user information
	CONSTANT	#, #H	
VT0	ALL	texts	Comparison telegram; input capable of duplication
MEUN	BINARY	A, M	Data invalid
RDY	BINARY	A, M	Telegram has been received
TELN	WORD	AW, MW	Number of the comparison telegram with which the received one agrees
MW0	WORD	AW, MW	useful information; output capable of duplication

CE Data

Runtime:	
Basic runtime:	not available
Additional runtime:	not available
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V5 / 935 PC 83 R201 / 35 ZE 93 R101

Description

The function block EMAS

- receives telegrams through a serial interface of the PLC
- compares these telegrams to comparison telegrams stored in the user program
- and, if these agree, provides the user data of the telegram received at the block's outputs.

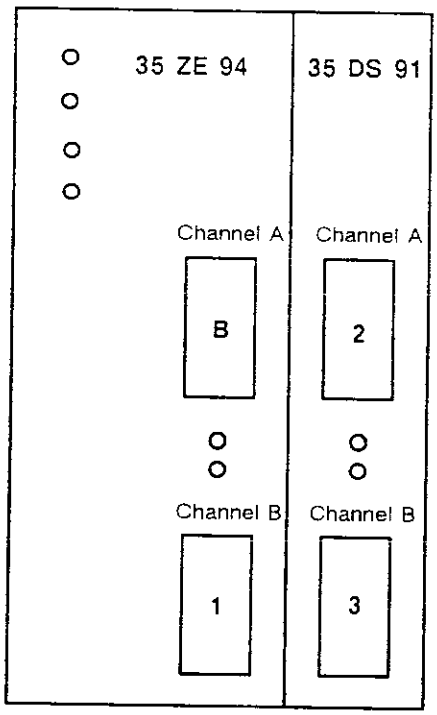
The received telegrams are fetched from the serial interface by an interface driver and are provided in a BUFFER for further processing by EMAS. The driver recognizes the end of the telegram by the end of telegram character. This end of telegram character is planned in the SINIT block.

Initialization of the serial interface

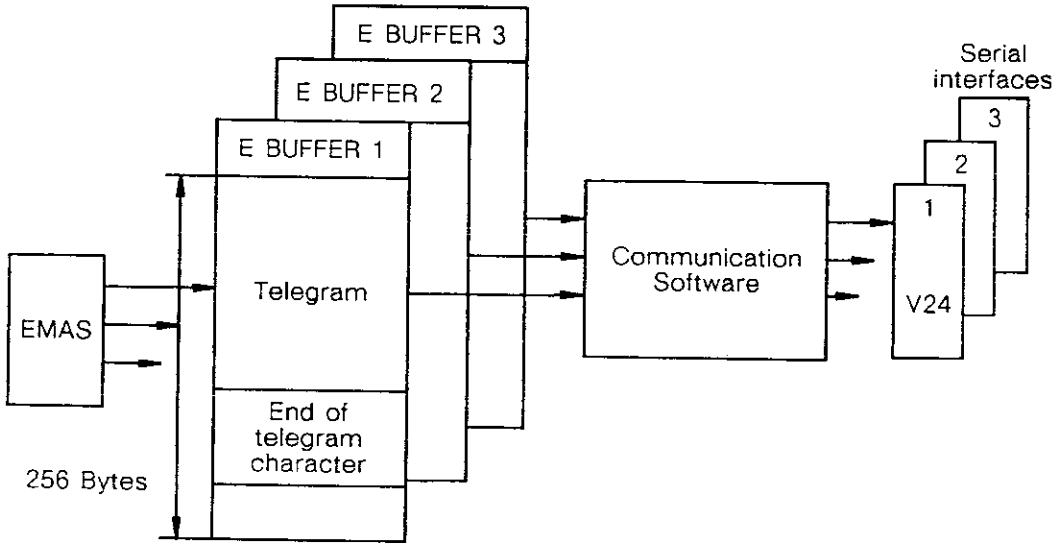
The EMAS must be initialized with the SINIT block *before* it can communicate with one of the three possible serial interfaces. Besides the required interface parameters, the following are specified in the SINIT block.

- An end of telegram character, which can be planned freely, in order to recognize the end of the telegram to be received
- Whether or not a character received through the interface is to be echoed. Synchronization of the data flow is basically achieved by way of the RTS and CTS signals.

Interfaces of the PLC



PLC with serial interfaces 1 to 3



The E buffers 1 to 3 are the receive buffers, which are assigned to serial interfaces 1 to 3.

Brief overview of the block's parameters:

- QUIT
- RDY
- MEUN

QUIT	MEUN	RDY	Meaning
1	0	0	The EMAS is disabled by QUIT = 1. In doing so, the outputs MEUN and RDY are permanently set to 0.
0	1	0	EMAS is enabled for reception, but still no telegram has been received and evaluated.
0	0	1	EMAS has received a valid telegram and is ready to receive a new telegram.
0	1	1	EMAS has received an invalid telegram. An acknowledgement at the QUIT input is necessary in order to be able to receive a new telegram. QUIT: 0->1
0->1	0	0	Acknowledgement after reception of an invalid telegram. *After* acknowledgement, the EMAS is enabled again by QUIT = 0

QUIT BINARY

The input QUIT controls reception of telegrams and also serves the purpose of acknowledgement in the event of an error occurring.

- QUIT = 0: Reception of telegrams enabled.
- QUIT = 1: Reception of telegrams *not* enabled. Acknowledgment after reception of an invalid telegram.

If agreement with *none* of the stored comparison telegrams is ascertained on comparison of a received telegram, the EMAS automatically assumes the "error" state. In this case, EMAS no longer processes any new telegrams until the error is acknowledged with a 1 signal at the input QUIT and reception of telegrams is enabled again (next cycle) with a 0 signal at the input QUIT.

SSK WORD

The number of the interface through which the block receives its telegrams is specified at the SSK input (interface identifier).
The following applies: 1 ≤ number ≤ 3

#ANU DIRECT CONSTANT

The number of outputs MW0...MWn at which the block provides the received user information is specified at the input #ANU (number of user information items). This is specified as a direct constant.

VT0 ... VTn ALL

The input VT0...VTn is capable of duplication. The comparison telegrams to be stored in the PLC program are specified at the inputs VT0...VTn. The block is capable of processing 1...99 telegrams. One telegram occupies 2 inputs, each telegram number being specified at one input and the actual telegram text being specified at the next one. The exact syntax and handling of the comparison telegrams are described separately in this description.

MEUN BINARY

The output MEUN (flag invalid) indicates whether or not the data at the outputs MW0...MWn is valid or invalid.

If a telegram is received and processed properly, the data at the outputs MW0...MWn is declared *valid*. The data at the outputs MW0...MWn is declared *invalid* if the received telegram does not agree with any of the stored comparison telegrams or if the received telegram cannot be processed properly.

MEUN = 0 -> Data at the outputs MW0...MWn is *valid*

MEUN = 1 -> Data at the outputs MW0...MWn is *invalid*

RDY BINARY

The output RDY (ready) indicates that a telegram has been received and processed.
The output RDY does not provide any information as to whether or not a valid or invalid telegram has been received.

RDY = 0 -> Still no telegram has been received

RDY = 1 -> A telegram has been received and processed

TELN WORD

If a valid telegram is received, the number of the affiliated comparison telegram is output through the output TELN (telegram number).

MW0 ... MWn WORD

The output MW0 can be duplicated (MW0...MWn). The user data communicated in the telegram currently received is output through the outputs MW0...MWn. This user data may consist of numerical values or any characters. This depends on which kind of dummy parameters have been planned in the comparison telegram. The user data of a telegram is stored beginning with the output MW0 and in the sequence in which they are planned in the comparison telegram. As many outputs MW0...MWn must be provided as are sufficient for the telegram with the *most* user data.

Detailed description of the functional scope of function block EMAS

The function block EMAS

- receives telegrams through a serial interface of the PLC
- compares these telegrams to the comparison telegrams stored in the user program
- and, if they agree, provides the user data of the telegram received at the block's outputs.

The telegrams received are fetched from the serial interface by an interface driver and are provided for further processing in a BUFFER for the EMAS. The driver recognizes the end of the telegram by the end of telegram character. This end of telegram character is planned in the SINIT block.

Initialization of the serial interface

The EMAS must be initialized with the SINIT block *before* it can communicate with one of the three possible serial interfaces. Besides the required interface parameters, the following are specified in the SINIT block:

- An end of telegram character that can be planned freely in order to recognize the end of the telegram to be received
- Whether or not a character received at the input is to be echoed. Synchronization of the data flow is basically achieved by way of the RTS and CTS signals.

Comparison telegrams

1...99 comparison telegrams are stored directly after the EMAS block.

The comparison telegrams serve to identify

- the current telegrams received
- and the user data contained in the telegrams received

The stored comparison telegrams each have a telegram number to identify them. Each comparison telegram may comprise up to 255 characters.

The comparison telegrams consist of:

- ASCII characters serving only to identify the telegram received,
- Dummy parameters for the user information to be received and to be output through the block's outputs.

As regards the dummy parameters for the user information, EMAS distinguishes between dummy parameters for *digits* and dummy parameters for *characters*.

Dummy parameter for digits:	# (1 # for each digit)
Dummy parameter for characters:	* (1 * for each character/byte)

Dummy parameter for digits

For each dummy digit parameter (#) of the comparison telegram, EMAS expects precisely *one* ASCII coded decimal digit in the telegram to be received.

Up to 5 dummy digits constitute one dummy parameter group. Such a group of dummy digits represents the numerical value of a decimal number comprising up to 5 digits.

No dummy parameter is specified for the decimal number's sign because EMAS takes it into account automatically.

The EMAS allocates one user information output to each numerical value belonging to a dummy parameter group. In doing so, EMAS automatically limits the value range to ±32767.

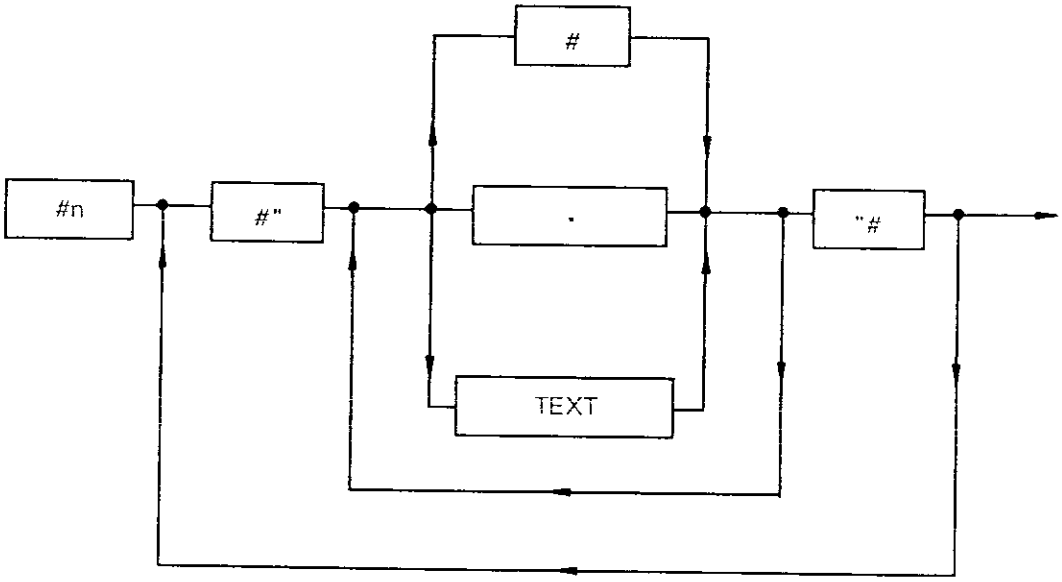
Dummy parameter for characters

For each dummy character (*) of the comparison telegram, EMAS expects *any* one character/byte in the telegram to be received. These may comprise ASCII characters of letters, but also all other hex values from 1...FF.

The length of a dummy character group is up to 255. If this were the case, the complete comparison telegram would consist of dummy characters only.

EMAS allocates the characters/bytes received without change and successively to its user information outputs MW0...MWn.

Syntax diagram: Structure of comparison telegrams



#n : Successive telegram number (direct constant 1...99)

#" : Start identifier for text input

"# : End identifier for text input in terminal mode

Important: This end identifier is *not* specified when using the ABB Procontic programming system. Only when texts are entered directly in the PLC in terminal mode do these have to be concluded by this end identifier.

* : Dummy parameter for character/byte

: Dummy parameter for digits

TEXT: All ASCII characters (1 ... FF) except * and #

Input of comparison telegrams

As regards input of comparison telegrams, two cases must be distinguished:
- Input in the ABB Procontic programming system
- Input directly in the PLC with a terminal

Input of comparison telegrams in the ABB Procontic programming system

Each comparison telegram consists of:
- the telegram number
- the telegram text

The telegram number and the telegram text are each separate operands. This is why the telegram number and the telegram text each occupy one separate input in the FBD symbol of the EMAS block. Therefore, two inputs are needed for one comparison telegram.

Example:
Input VT0: #1
(No. of the first comparison telegram)
Input VT1: #"PRINT### IDENTIFIER****
(Text of the first comparison telegram)

Apart from the ASCII characters for * and #, all ASCII characters are possible in the telegram text.

When entering special ASCII characters such as "start of line" <CR>, the following must be observed when using the ABB programming system:
Special characters are entered by:
\Numerical value of the character
The character's numerical value is specified as a three-digit decimal number.

Example:
The following telegram text is to be entered in the ABB programming system:
Temperature <CR> boiler 1
The following applies: <CR> = 013
The text input in the programming system is as follows:
#"temperature\013boiler 1

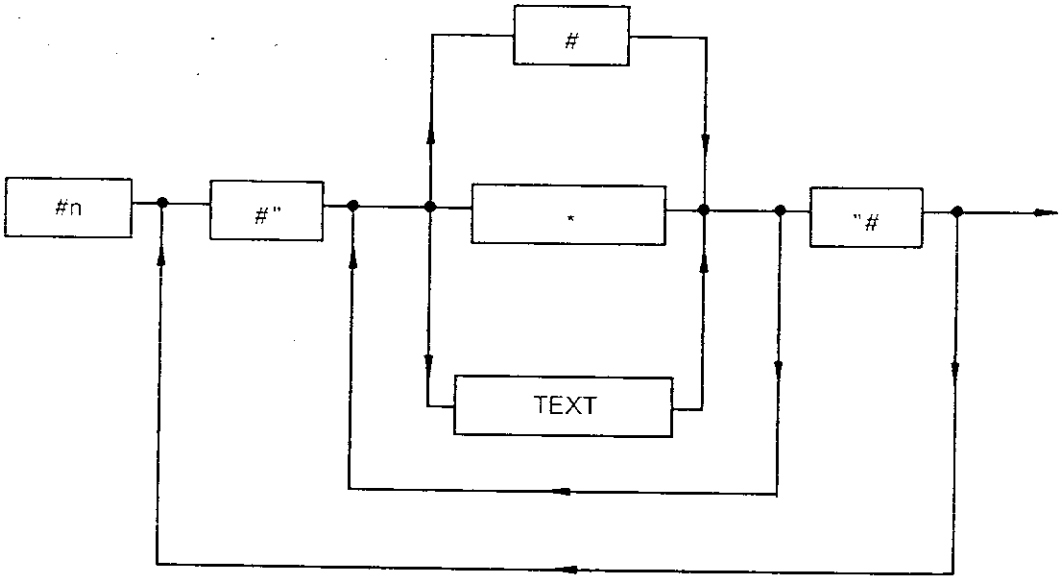
Input of comparison telegrams directly in the PLC with a terminal

The telegrams are entered directly after the EMAS block, i.e. the first comparison telegram begins at the program memory address following the operand at the output MWn.

The telegram number is specified as a direct constant. When entered, the telegram text is embedded in the special characters #" (start of text input) and "# (end of text input).

Apart from the ASCII characters for * and #, all ASCII characters from 1...FF are possible; the ASCII character <NUL> is used as a prefix for the format identifier in the send buffer (see also PLC description, chapter entitled "language scope").

Syntax diagramm: Text input directly in the PLC with a terminal



#n : Successive telegram number (direct constant 1 ... 99)

#" : Start identifier for text input

"# : End identifier for text input in terminal mode

* : Dummy parameter for character/byte

: Dummy parameter for digits

TEXT: All ASCII characters (1 ... FF) except * and #

Correcting a character within a text input in terminal mode

An incorrectly entered character can *not* be corrected when entering a text. To correct the text input it must be terminated with a text input end identifier "# and program input must be terminated by <CR>. Continue with program input at the address of the character to be corrected. Text input then begins with the text input start identifier #".

Input of texts containing more than 80 characters on a terminal

When making a text input, no more than 80 characters can be displayed on each line of the screen. If the text is longer than 80 characters, text input must be terminated at the end of the screen line by means of the text input end identifier "# and you must jump to the next line on the screen by means of the ; character. Now continue text input with the text input start identifier #".

Receiving and processing telegrams

Receiving telegrams

The QUIT input controls reception of telegrams and also serves the purpose of acknowledgement in the event of an error.

- QUIT = 0: Telegram reception enabled.
- QUIT = 1: Telegram reception *not* enabled. Acknowledgement after reception of an invalid telegram.

The EMAS automatically assumes the "error" state if agreement with *none* of the stored comparison telegrams is ascertained when comparing a telegram received. In this case, EMAS will not process any new telegram until the error is acknowledged with a 1 signal at the QUIT input and reception of telegrams is enabled again (next cycle) with a 0 signal at the QUIT input.

Processing telegrams

The EMAS compares a received telegram character by character to the comparison telegrams stored. In doing so, agreement between the dummy parameters in the comparison telegram and the affiliated current characters in the received telegram is checked.

Important: For EMAS, the telegram end identifier indicates the end of the telegram to be received. This is why the telegram end identifier must *not* occur *within* a telegram. This also applies to user characters received on the basis of a dummy parameter (# or *).

Agreement

If the EMAS ascertains agreement between the received telegram and one of its comparison telegrams, the following applies to the outputs:

MEUN = 0, because it has been possible to receive and process the telegram *properly*

RDY = 1, because a telegram has been *received* and *processed*

TELN = Number of the relevant comparison telegram

MW0 ... MWn= Current user data from the telegram received

The coherent digits defined by a dummy digit group in the comparison telegram are read out of the received telegram, are combined in one numerical value and this is output to *one* word operand (user information output). Text characters marked by dummy text characters (*) in the comparison text are read out of the received telegram and each *single* character/byte is output without change to a word operand (user information output).

The user information is allocated to the outputs MW0...MWn in the same sequence as the user information occurs in the currently received telegram (see also examples).

The maximum number of user information word operands to be output is specified at the input #ANU. The maximum number results from the telegram with the most user information word operands to be reserved (up to 256 word operands, in which case the telegram would consist of user information *only*).

Special case:
If more user information is planned in a comparison telegram than user information outputs are available on the block, i.e. specified at the input #ANU, the received user data is output through the user information outputs until the end of the outputs has been reached. The remaining user data received will not be output.

The following applies in this error case:

MEUN = 1, because it has *not* been possible to process the telegram properly

RDY = 1, because a telegram has been received and processed

TELN = Text No. of the relevant comparison telegram

MW0...MWn = The part of the current user data from the received telegram for which the number of planned user information outputs suffices.

No agreement

The received telegram agrees with *none* of the stored comparison telegrams.

The following then applies to the outputs:

MEUN = 1 because the received telegram is invalid

RDY = 1 because a telegram has been received and processing of it has been terminated

The EMAS is now in the "ERROR" state. In this case, EMAS does not process any new telegram until the error is acknowledged with a 1 signal at the QUIT input and the reception of telegrams is enabled again (next cycle) with a 0 signal at the QUIT input.

Acknowledgement of the error

Set QUIT = 1 and then, in the next cycle QUIT = 0.

After reception of telegrams has been enabled, it may happen that still no telegram is available for reception or that a telegram has not yet been received and evaluated completely.

In this case, the outputs are set as follows

- MEUN = 1 and
- RDY = 0

until a telegram has been received and processed completely.

Communication between several EMAS blocks and the same serial interface

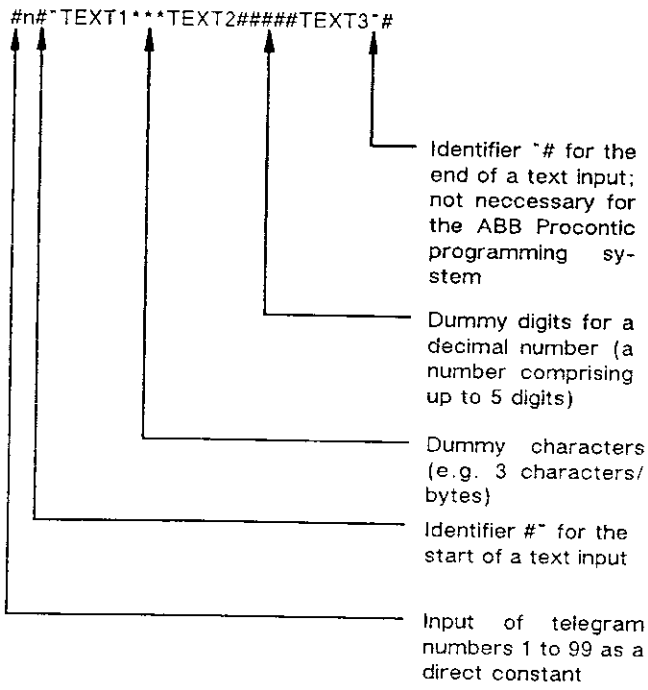
- EMAS blocks of a user program which access *the same* serial interface must be interlocked so that only ever *one* EMAS block is active. If this is not done, telegrams may be processed by the wrong EMAS and declared invalid.
- If both user program 1 and also user program 2 contain EMAS blocks which access *the same* serial interface, they must be interlocked so that only ever *one* EMASm block is active. If this is not done, telegrams may be processed by the wrong EMAS and declared invalid.

A telegram loss can be avoided by interlocking of the EMAS blocks. Interlocking must be planned so that only the EMAS block is enabled for which the telegram arriving through the interface is intended.

Communikation by an EMAS block and a DRUCK block with the same serial interface

An EMAS and a DRUCK block can use the *same* serial interface without special precautions having to be taken.

Example of a comparison telegram



Notes:

- The texts designated TEXT1, TEXT2 and TEXT3 in the example are permanently planned texts which EMAS does not output to user information outputs. They serve exclusively to identify an arriving telegram.

  Any sequence of texts and dummy parameters for characters and digits is possible.

- Up to five # may be specified as dummy parameters for digits. Precisely *one* dummy parameter (#) must be specified in the comparison telegram for each digit of the decimal number. *No* dummy parameter must be specified for the decimal number's sign. EMAS takes its sign into account automatically.

E.g.:	Decimal number	Dummy parameters
	1234	####
	+1234	####
	−1234	####

The EMAS block checks the received decimal number in relation to its significant range. Only numbers within the ±32767 range can be processed in the PLC. If the received decimal number exceeds the significant range, the EMAS will automatically insert the maximum respective limit. The limit for a positive number is +32767 and the limit is −32767 for a negative number.

Further examples of comparison telegrams

Input with ABB Procontic programming system

Example 1: Text 1: #1#"RO104

Example 2: Text 2: #2#"D0##"#"###

Terminate text input and restart, see also special case

Example 3: Text 3: #3#"CO##****

Example 4: Text 4: #4#"*****###**

Special case, example 2:

If the text character (") is used and a dummy parameter for digits (#) follows the text character, the input must be terminated after the text character ("). A subsequent text or dummy parameter begins with the input identifier of a new input (#").

Input directly in the PLC in terminal mode

Example 1: Text 1: #1#"RO104"#

Example 2: Text 2: #2#"D0##""##"###"#

Terminate text input and restart, see also special case

Example 3: Text 3: #3#"CO##****"#

Example 4: Text 4: #4#"*****###**"#

Special case, example 2:

If the text character (") is used and a dummy parameter for digits (#) follows the text character, the input must be terminated after the text character ("). A subsequent text or dummy parameter begins with the input identifier of a new input (#").

Input of comparison telegrams and the corresponding received telegram

User input	Meaning	Telegram received through serial interface
#1	Telegram No. 1, direct constant	
#"	Identifier for the start of text input	
R	Letter in ASCII	52H
O	" "	4FH
1	Number in ASCII	31H
0	" "	30H
4	" "	34H
	End of text character e. g. <CR>	0DH
"#	Identifier for the end of text input *(dropped when using the programming system)*	
#2	Telegram No. 2, direct constant	
#"	Identifier for the start of text input	
D	Letter in ASCII	44H
O	" "	4FH
	Sign (+)	2BH
#	1st of two digits	xxH
#	2nd of two digits	xxH
"	Character in ASCII	22H
"#	Identifier for the end of text input *(dropped when using the programming system)*	
#"	Identifier for the start of text input	
#	1st of 3 digits	yyH
#	2nd of 3 digits	yyH
#	3rd of 3 digits	yyH
"	Character in ASCII	22H
"#	Identifier for the end of text input *(dropped when using the programming system)*	
	End of text character, e. g. <CR>	0DH
#3	Telegram No. 3, direct constant	
#"	Identifier for the start of text input	
C	Letter in ASCII	43H
O	" "	4FH
	Sign (-)	2DH
#	1st of 2 digits	zzH
#	2nd of 2 digits	zzH
*	1st of 4 characters	nnH
*	2nd of 4 characters	nnH
*	3rd of 4 characters	nnH
*	4th of 4 characters	nnH
	End of text character, e.g. <CR>	0DH
"#	Identifier for the end of text input *(dropped when using the programming system)*	
#4	Telegram No. 4, direct constant	
#"	Identifier for the start of text input	
*	1st of 5 characters	mmH
*	2nd of 5 characters	"
*	3rd of 5 characters	"
*	4th of 5 characters	"
*	5th of 5 characters	"
#	1st of 3 digits	xxH
#	2nd of 3 digits	"
#	3rd of 3 digits	xxH
*	1st of 2 characters	nnH
*	2nd of 2 characters	nnH
	End of text character <CR>	0DH
"#	Identifier for the end of text input *(dropped when using the programming system)*	

The following telegrams have been received and evaluated:

Example 1

RO104<CR>

The number 1 is output through the TELN output.

The outputs MW0...MWn are *not* written because no user information is planned.

As the received telegram has been evaluated and agrees with a comparison telegram, the following applies:

MEUN = 0
RDY = 1
TELN = 1

Example 2

DO+12"457"<CR>

The number 2 is output through the TELN output

The value 12 is present at the output MW0 as the contents of a word flag.
The value 457 is present at the output MW1 as the contents of a word flag.
As the received telegram has been evaluated and agrees with a comparison telegram, the following applies:

MEUN = 0
RDY = 1
TELN = 2
MW0 = 12
MW1 = 457

Example 3

CO-11AUTO<CR>

The number 3 is output through the TELN output.

-11 is present at the output MW0 as the contents of a word flag.

41H (A in ASCII) is present at the output MW1 as the contents of a word flag.

55H (U in ASCII) is present at the output MW2 as the contents of a word flag.

54H (T in ASCII) is present at the output MW3 as the contents of a word flag.

4FH (O in ASCII) is present at the output MW4 as the contents of a word flag.

As the received telegram has been evaluated and agrees with a comparison telegram, the following applies:

MEUN = 0
RDY = 1
TELN = 3
MW0 = -11
MW1 = A
MW2 = U
MW3 = T
MW4 = O

Example 4

TEMPO120KM<CR>

The number 4 is output through the TELN output.

54H (T in ASCII) is present at the output MW0 as the contents of a word flag.

45H (E in ASCII) is present at the output MW1 as the contents of a word flag.

4DH (M in ASCII) is present at the output MW2 as the contents of a word flag.

50H (P in ASCII) is present at the output MW3 as the contents of a word flag.

4FH (O in ASCII) is present at the output MW4 as the contents of a word flag.

The value 120 is present at the output MW5 as the contents of a word flag.

4BH (K in ASCII) is present at output MW6 as the contents of a word flag.

4DH (M in ASCII) is present at output MW7 as the contents of a word flag.

As the received telegram has been evaluated and agrees with a comparison telegram, the following applies:

MEUN = 0
RDY = 1
TELN = 4
MW0 = T
MW1 = E
MW2 = M
MW3 = P
MW4 = O
MW5 = 120
MW6 = K
MW7 = M

Example 5

XY25OTTO<CR>

The received telegram has been evaluated, but agrees with *none* of the comparison telegrams.

Therefore the following applies:

MEUN = 1
RDY = 1
TELN = is not allocated

Reception of further telegrams is blocked until reception of the invalid telegram is acknowledged at the QUIT input.

End of telegram character

The receive driver of the EMAS block recognizes the end of the received telegram by virtue of the end of telegram character.

Examples of end of telegram characters to be specified in the SINIT block:

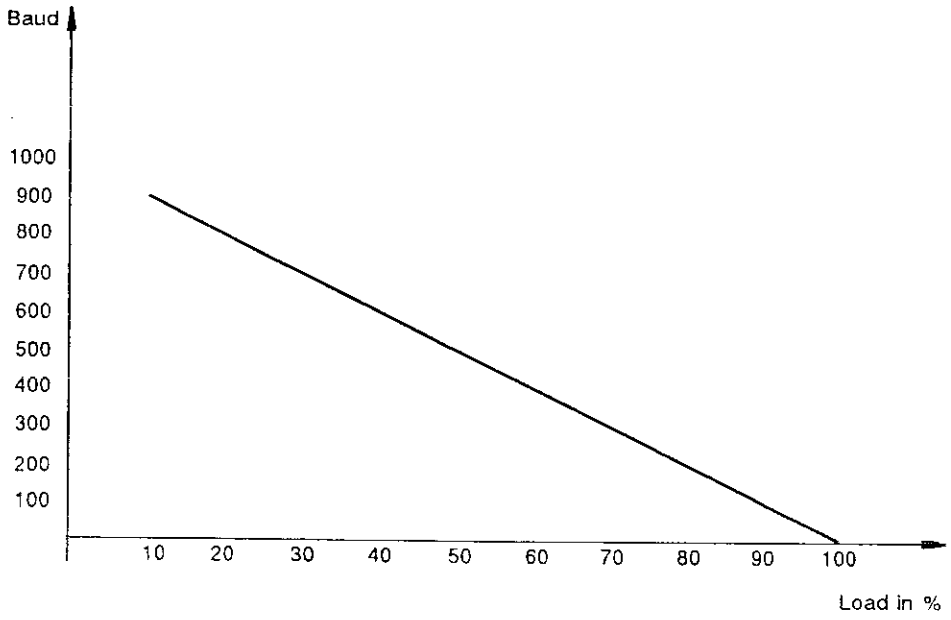
<ETX> = 03H = 3
<EOT> = 04H = 4
<CR> = 0DH = 13
<LF> = 0AH = 10
<SP> = 20H = 32
. . .
. . .

Effective data transmission rate in the EMAS block

The effective data transmission rate (characters actually read through the serial interface) depends on the load on the PLC and the baud rate set on the jumper X8.

The following relationship roughly applies:

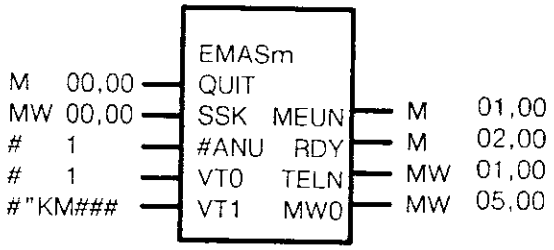
Marginal condition: Baud rate set at X8: 9600 baud



Important: The load indication (command AL) is correct only if no communication is taking place through the serial interface at the time of indication. Communication adulterates the load indication and renders it useless. If a user program contains the blocks DRUCK or EMAS, these must be disabled while determining the load.

Example

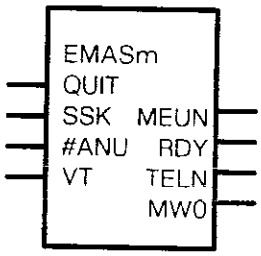
FBD/LD



IL

```
!BA    0
EMAS
       M    00,00
       MW   00,00
       #    1
       M    01,00
       M    02,00
       MW   01,00
       MW   05,00
       #    1
       #"KM###
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
QUIT	E	L	N	P	Y	0	0
SSK	E	W	N	P	Y	0	0
#ANU	K	W	N	P	Y	0	0
VT	E	X	N	P	Y	2	0
MEUN	A	L	N	P	Y	0	0
RDY	A	L	N	P	Y	0	0
TELN	A	W	N	P	Y	0	0
MW	A	W	N	P	Y	1	0

CE IL Definition

```
00000    !BA     0          Nr      Block No. (preset to 0)
00001    EMAS
00002            PP  0      QUIT    Input BINARY (acknowledgement, no enable)
00003            PP  0      SSK     Input WORD (interface identifier)
00004            PP  0      #ANU    DIRECT CONSTANT (number of MWs)
00005            PP  0      MEUN    Output BINARY (data invalid)
00006            PP  0      RDY     Output BINARY (reception of text)
00007            PP  0      TELN    Output WORD (telegram number)
     [           1
00008            PP  1      MW      Output WORD (useful information)
     ]           1
     [           2
00009            PP  2      VT      Input ALL (comparison text)
     ]           2
```

RESET MEMORY, DOMINATING — SR

This connection element realizes a binary memory with the "dominating reset" characteristic.

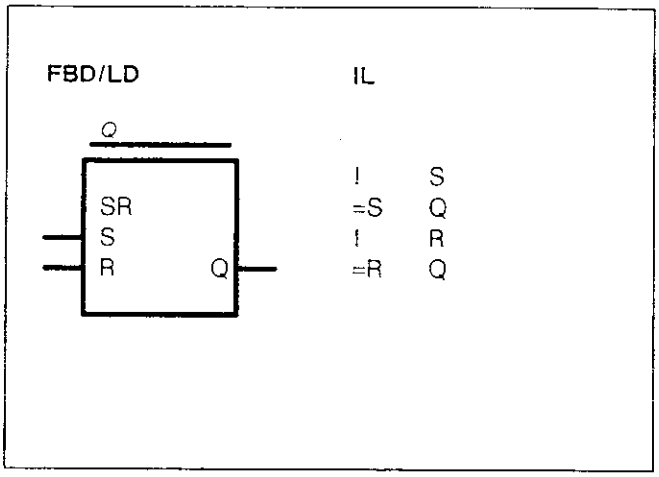
The operand Q, which performs the storage function, is entered above the graphical symbol. The status of the operand Q is allocated to the operand at the output Q.

A status 1 at the input S sets the operand Q to a status 1.

A status 1 at the input R resets the operand Q to the status 0.

A simultaneous 1 status at the inputs S and R resets the operand to a status 0 (dominating reset).

A status 0 at the input S or R has no influence on the operand Q.



Parameters

S	BINARY	E, M, A, S, K	Set input
R	BINARY	E, M, A, S, K	Reset input
Q	BINARY	M, A	Flip-flop output

CE Data

Runtime:	
Basic runtime:	8.6 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The operand Q, which performs the storage function, is entered above the graphical symbol. The status of the operand Q is allocated to the operand at the output Q.

A status 1 at the input S sets the operand Q to a status 1.

A status 1 at the input R resets the operand Q to the status 0.

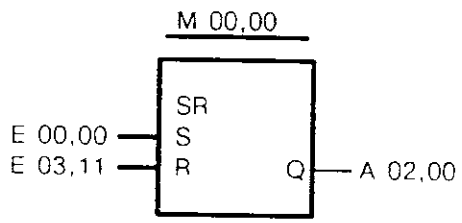
A simultaneous 1 status at the inputs S and R resets the operand to a status 0 (dominating reset).

A status 0 at the input S or R has no influence on the operand Q.

The inputs and the output cannot be duplicated.
The inputs R and S can be inverted.

Example

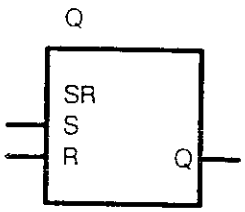
FBD/LD



IL

```
!     E 00,00
=S    M 00,00
!     E 03,11
=R    M 00,00
!     M 00,00
=     A 02,00
```

CE FBD Definition



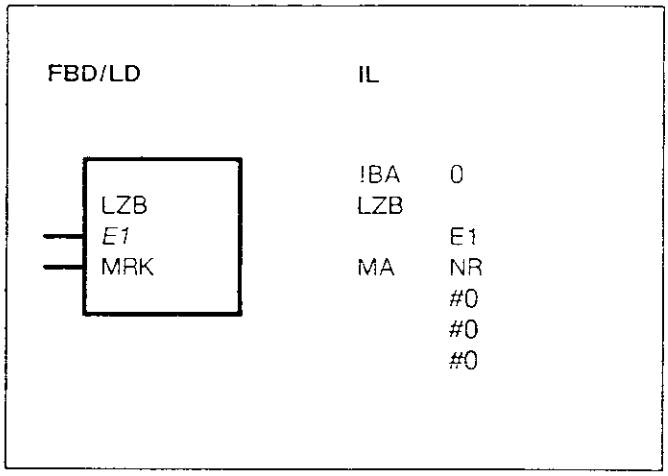
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
S	E	L	Y	P	Y	0	0
R	E	L	Y	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	I	PP 0	S	Input BINARY
00002	=S	PP 0	Q	Output BINARY
00004	I	PP 0	R	Input BINARY
00006	=R	PP 0	Q	Output BINARY

This function block controls processing of a program part. This program part is called run number block and begins with the function block LZB and ends with the affiliated target label specified at the input MRK. This program part is processed as follows depending on the value of the operand at the input E1:

E1 = 0: Program part is not processed
E1 = 1: Program part is processed during every cycle
E1 = 2: Program part is processed during every second cycle
E1 = n: Program part is processed during every nth cycle



Parameters

E1	WORD	EW, MW, AW, KW	Run number
MRK	SPECIAL	MRK	Target label

CE Data

Runtime:
- Basic runtime: 29 µs
- Additional runtime: ---

Output updating: not applicable
Number of historical values: 1 word
Available as of: ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block controls processing of a program part. This program part is called run number block and begins with the function block LZB and ends with the affiliated target label specified at the input MRK. This program part is processed as follows depending on the value of the operand at the input E1:

E1 = 0: Program part is not processed
E1 = 1: Program part is processed during every cycle
E1 = 2: Program part is processed during every second cycle
E1 = n: Program part is processed during every nth cycle

FBD/LD

Designation of the target label: MRK No.

Where:	MRK	Key word
	0 ≤ Nr ≤ 999	Number of the label

IL

Designation of the target label: MA No.

Where:	MA	Key word
	0 < Nr < 999	Number of the label

The PLC automatically calculates:

- The number of historical values to be skipped
- The address of the target label
- The pointer to the own historical value

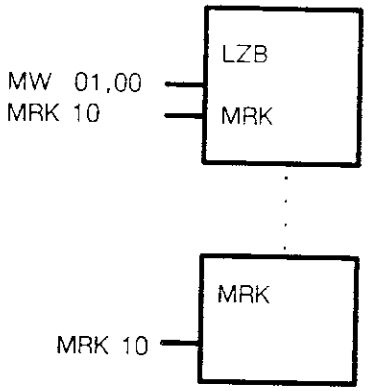
The PLC enters the 3 computed values in the intended positions of the LZB block. When planned in IL, the user can basically fill these 3 program words with any contents. However, it is suggested that the numeric value 0 be entered here.

Caution:

The 3 program words should not be written with NOPs because these are removed when optimizing the user program.

Example

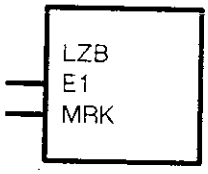
FBD/LD



IL

```
!BA     0
LZB
        MW 01,00
MA      10
        # 0
        # 0
        # 0
        .
        .
        .
        .
      MA 10
```

CE FBD Definition



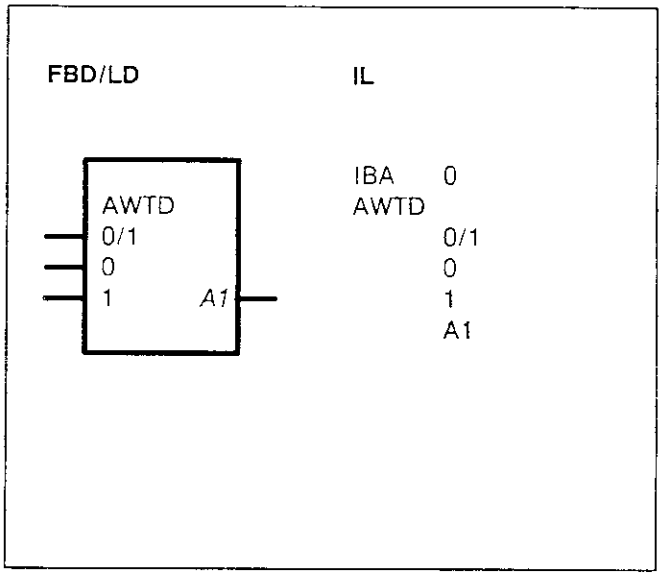
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
MRK	S	S	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	LZB			
00002		PP 0	E1	Input WORD (run number)
00003	MA	PP 0	MRK	Targel Label
00004		# 0		
00005		# 0		
00006		# 0		

A 0 signal at the binary input 0/1 allocates the value of the double word operand at the input 0 to the double word operand at the output A1.

A 1 signal at the binary input 0/1 allocates the value of the double word operand at the input 1 to the double word operand at the output A1.



Parameters

0/1	BINARY	E, M, A, S, K	Switchover input
0	DOUBLE WORD	MD, KD	Double word input for 0/1 = 0
1	DOUBLE WORD	MD, KD	Double word input for 0/1 = 1
A1	DOUBLE WORD	MD	Double word output

CE Data

Runtime:	
Basic runtime:	41 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

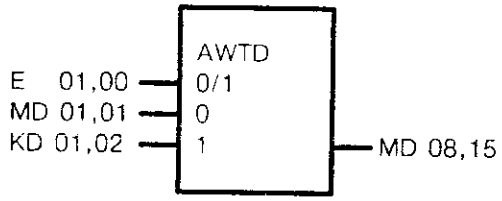
A 0 signal at the binary input 0/1 allocates the value of the double word operand at the input 0 to the double word operand at the output A1.

A 1 signal at the binary input 0/1 allocates the value of the double word operand at the input 1 to the double word operand at the output A1.

The inputs and the output can neither be duplicated nor negated/inverted.

Example

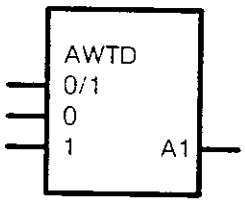
FBD/LD



IL

```
!BA    0
AWTD
       E  01,00
       MD 01,01
       KD 01,02
       MD 08,15
```

CE FBD Definition



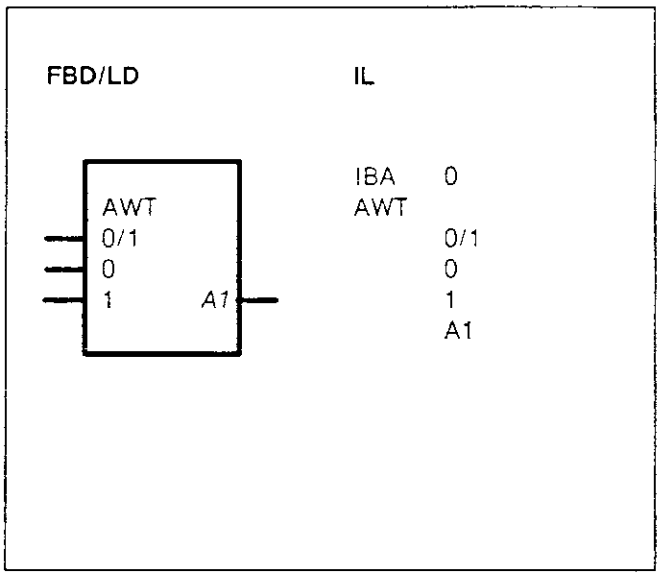
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0/1	E	L	N	P	Y	0	0
0	E	D	N	P	Y	0	0
1	E	D	N	P	Y	0	0
A1	A	D	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	AWTD			
00002		PP 0	0/1	Input BINARY (Selector)
00003		PP 0	0	Input DOUBLE WORD
00004		PP 0	1	Input DOUBLE WORD
00005		PP 0	A1	Output DOUBLE WORD

A 0 signal at the binary input 0/1 allocates the value of the word operand at the input 0 to the word operand at the output A1.

A 1 signal at the binary input 0/1 allocates the value of the word operand at the input 1 to the word operand at the output A1.



Parameters

0/1	BINARY	E, M, A, S, K	Switchover input
0	WORD	EW, MW, AW, KW	Word input for 0/1 = 0
1	WORD	EW, MW, AW, KW	Word input for 0/1 = 1
A1	WORD	AW, MW	Word output

CE Data

Runtime:	
Basic runtime:	30 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

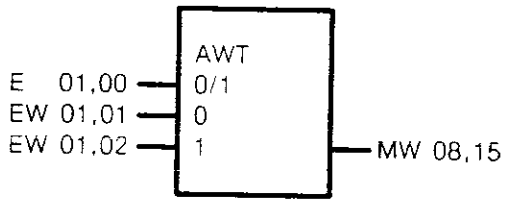
A 0 signal at the binary input 0/1 allocates the value of the word operand at the input 0 to the word operand at the output A1.

A 1 signal at the binary input 0/1 allocates the value of the word operand at the input 1 to the word operand at the output A1.

The inputs and the output can neither be duplicated nor negated/inverted.

Example

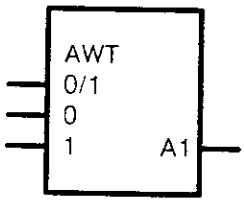
FBD/LD



IL

```
!BA   0
AWT
      E  01,00
      EW 01,01
      EW 01,02
      MW 08,15
```

CE FBD Definition



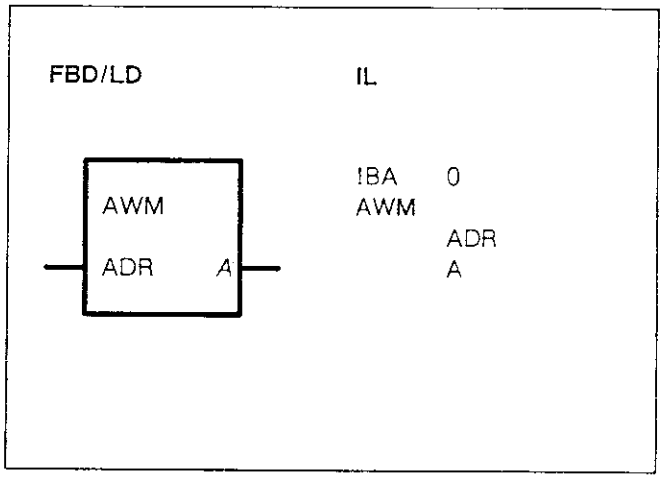
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0/1	E	L	N	P	Y	0	0
0	E	W	N	P	Y	0	0
1	E	W	N	P	Y	0	0
A1	A	W	N	P	N	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	AWT			
00002		PP 0	0/1	Input BINARY (Selector)
00003		PP 0	0	Input WORD
00004		PP 0	1	Input WORD
00005		PP 0	A1	Output WORD

This function block reads the value of an operand, using the method of indirect addressing. The value read is allocated to the output A.

Note: The AWM block can only be used meaningfully in conjunction with the ADRWA block.



Parameters

ADR	WORD	EW, AW, MW, KW	Indirect address of the operand to be read
A	WORD	AW, MW	Value of the operand read

CE Data

Runtime:

Basic runtime:	26 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V4 / 935 PC 82 R101 / 35 ZE 93 R101

Description

This function block reads the value of an operand, using the method of indirect addressing. The value read is allocated to the output A

Note: The AWM block can only be used meaningfully in conjunction with the ADRWA block.

The value of the operand at the input ADR is interpreted as an address of the operand to be read (indirect addressing).

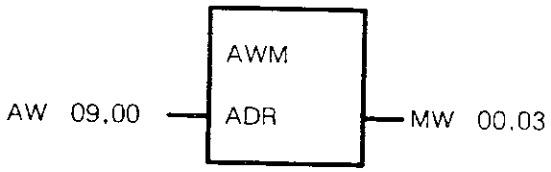
Therefore, the operand at the input ADR represents an indirect address together with its value. This indirect address is generated by the ADRWA function block.

The input and output can neither be duplicated nor negated.

Note: An explanation of the indirect addressing method and the possibilities of using the AWM function block are described in the section dealing with the ADRWA function block.

Example

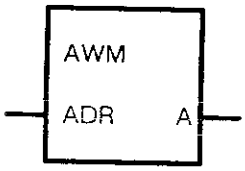
FBD/LD



IL

```
!BA   0
AWM
      AW  09,00
      MW  00,03
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
ADR	E	W	N	P	Y	0	0
A	A	W	N	P	N	0	0

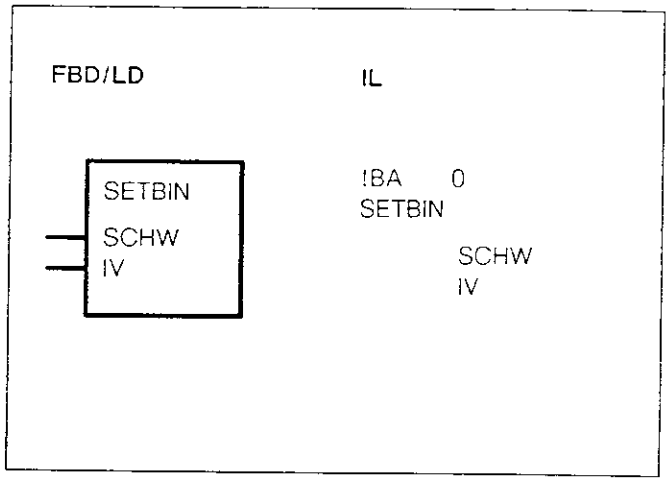
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	AWM			
00002		PP 0	ADR	Input WORD (operand address)
00003		PP 0	A	Output WORD (operand value)

SET BINARY VIDEO THRESHOLD (OMS-F) SETBIN

This function block sets the binary video threshold on the camera interface 35 KI 90.

This block is needed when using the video sensor OMS-F.



Parameters

SCHW	WORD	EW, AW, MW, KW	Value of the binary video threshold
IV	WORD	EW, AW, MW, KW	Number of the 35 IV 90 module on which the camera interface is plugged

CE Data

Runtime:	
Basic runtime:	60 µs
Additional runtime:	no
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block sets the binary video threshold on the camera interface 35 KI 90.

Binary video threshold:
All gray tones below this threshold are interpreted by the 35 KI 90 as black and all gray tones above it as white.

The binary video threshold is normally entered in the PLC in "teach-in mode" by means of the PLC command X-binary (XB) and is stored there. On initialization of the PLC, this binary threshold is then loaded to the camera interface 35 KI 90. In some applications it is necessary to modify the binary video threshold for a specific measurement. This is then done with the SETBIN block.

SCHW WORD

Value of the binary video threshold.

The value of the required binary video threshold is specified with the operand at this input.
Value range: 1...128.

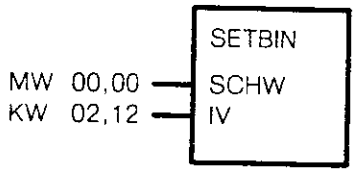
IV WORD

Number of the 35 IV 90 module.

The number of the 35 IV 90 module on which the camera interface is plugged is specified with the operand at this input.
Value range: 0...5.

Example

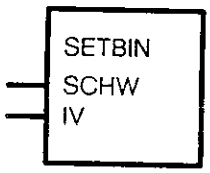
FBD/LD



IL

```
!BA    0
SETBIN
       MW 00,00
       KW 02,12
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
SCHW	E	W	N	P	Y	0	0
IV	E	W	N	P	Y	0	0

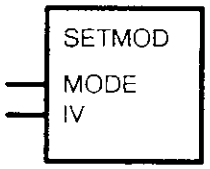
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	SETBIN			
00002		PP 0	SCHW	Input WORD (video threshold)
00003		PP 0	IV	Input WORD (IV No.)

This function block sets the required image processing mode on the selected iconic processor (35 IV 90 module).

This block is needed when using the video sensor OMS-F.

FBD/LD



IL

```
!BA     0
SETMOD
        MODE
        IV
```

Parameters

MODE	WORD	EW, AW, MW, KW	Value entered in the MODE register
IV	WORD	EW, AW, MW, KW	Number of the 35 IV 90 module

CE Data

Runtime:	
Basic runtime:	51 µs
Additional runtime:	no
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block sets the required image processing mode on the selected iconic processor (35 IV 90 module).

MODE WORD

Value entered in the mode register.

The value of the operand at this input is entered in the mode register of the selected iconic processor (IV). This sets the mode of the iconic processor for image processing.

Bit
15 0

0	0	FRE	REF	MXY	HIS	0	0	0	INV	0	MON	INT	MM	SBA	0

Bit 13, FRE= 0: Reference area fixed in the 1st pixel row
1: Reference areas free in windows 6 and 8

Bit 12, REF= 0: Adaptive A/D conversion without reference
1: Adaptive A/D conversion with reference

Bit 11, MXY=0: Definition of MX
1: Definition of MY

Bit 10, HIS= 0: Length and area measurement
1: Histogram definition

Bit 6, INV = 0: Length and area measurement for white areas
1: Length and area measurement for black areas

Bit 4, MON= 0: Gray tone image display
1: Binary display

Bit 3, INT= 0: End of image recording without interrupt
1: End of image recording with interrupt

Bit 2, MM= 0: Single measurement mode
1: Continuous measurement mode

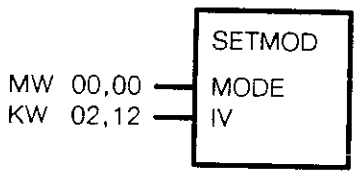
Bit 1, SBA= 1: Start of image recording; the bit is automatically set to 0 after image recording.

IV WORD
Number of the 35 IV 90 module.

The number of the required 35 IV 90 module with which measurement is to take place is specified with the operand at this input.
Value range: 0...5

Example

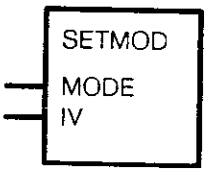
FBD/LD



IL

```
!BA     0
SETMOD
        MW 00,00
        KW 02,12
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
MODE	E	W	N	P	Y	0	0
IV	E	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	SETMOD			
00002		PP 0	MODE	Input WORD (mode register)
00003		PP 0	IV	Input WORD (IV No.)

This connection element realizes a binary memory with the characteristic "dominating set".

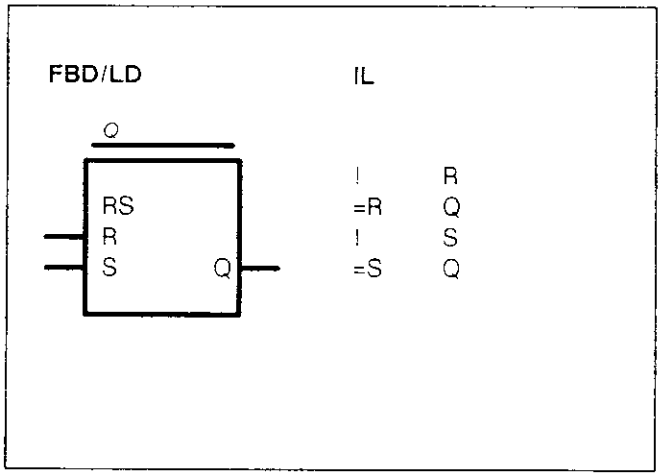
The operand Q, which performs the storage function, is entered above the graphical symbol. The status of the operand Q is allocated to the operand at the output Q.

A status 1 at the input R resets the operand Q to a status 0.

A status 1 at the input S sets the operand Q to a status 1.

A simultaneous 1 status at the inputs S and R sets the operand Q to a status 1 (dominating set).

A status 0 at the input S or R has no influence on the operand Q.



Parameters

R	BINARY	E, M, A, S, K	Reset input
S	BINARY	E, M, A, S, K	Set input
Q	BINARY	M, A	Flip-flop output

CE-Data

Runtime:	
Basic runtime:	8.6 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The operand Q, which performs the storage function, is entered above the graphical symbol. The status of the operand Q is allocated to the operand at the output Q.

A status 1 at the input R resets the operand Q to a status 0.

A status 1 at the input S sets the operand Q to a status 1.

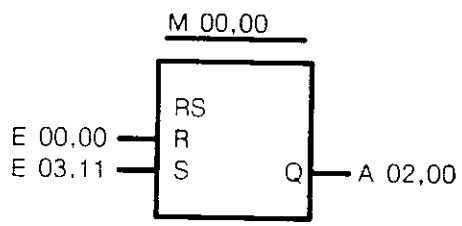
A simultaneous 1 status at the inputs S and R sets the operand Q to a status 1 (dominating set).

A status 0 at the input S or R has no influence on the operand Q.

The inputs and the output cannot be duplicated.
The inputs R and S can be inverted.

Example

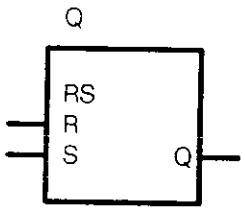
FBD/LD



IL

```
!     E 00,00
=R    M 00,00
!     E 03,11
=S    M 00,00
!     M 00,00
=     A 02,00
```

CE FBD Definition

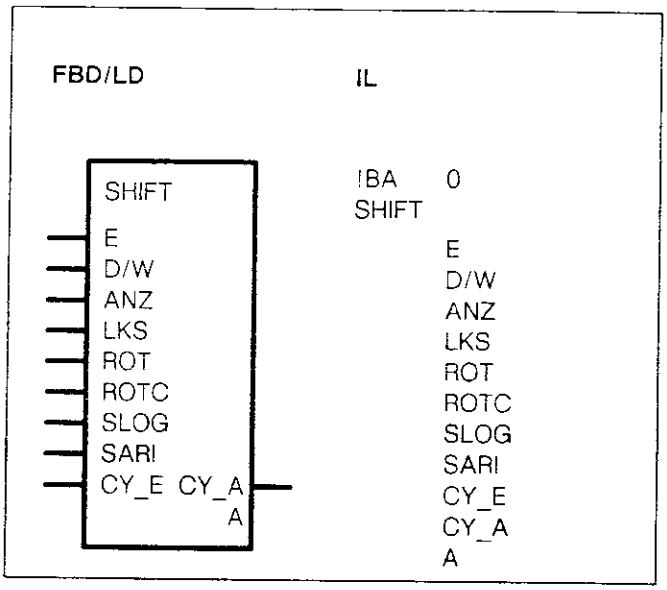


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
R	E	L	Y	P	Y	0	0
S	E	L	Y	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	!	PP 0	R	Input BINARY
00002	=R	PP 0	Q	Output BINARY
00004	!	PP 0	S	Input BINARY
00006	=S	PP 0	Q	Output BINARY

This function block shifts the operand present at the input by a specified number of bit positions to the left or right.



Parameters

E	WORD	EW, AW, MW, KW	Operand to be shifted
	DOUBLE WORD	KD, MD	Word or double word
D/W	BINARY	E, A, M, K, S	Format selection, double word or word D/W = 0 -> Word D/W = 1 -> Double word
ANZ	WORD	EW, AW, MW, KW	Number of bit positions to be shifted
LKS	BINARY	E, A, M, K, S	Shift direction, left or right LKS = 0 -> shift to right LKS = 1 -> shift to left
ROT	BINARY	E, A, M, K, S	Shift-Art: ROTATE
ROTC	BINARY	E, A, M, K, S	Shift-Art: ROTATE by the CARRY FLAG
SLOG	BINARY	E, A, M, K, S	SHIFT type: LOGICAL SHIFT
SARI	BINARY	E, A, M, K, S	SHIFT type: ARITHMETIC SHIFT
CY_E	BINARY	E, A, M, K, S	Initial value for the CARRY FLAG in the case of shift type ROTC
CY_A	BINARY	A, M	Status of the CARRY FLAG after the shift
A	WORD	AW, MW	Result of the shift
	DOUBLE WORD	MD	

CE Data

Runtime:	
Basic runtime:	37 ... 78 µs according to the shift type
Additional runtime:	1 µs per bit position to be shifted
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block shifts the operand present at the input by a specified number of bit positions to the left or right.

The result of the shift and the CARRY flag are available at the block's outputs. The operand at the input remains unchanged.

The required shift type is planned at the inputs:

- ROT
- ROTC
- SLOG
- SARITH

If several shift types are specified simultaneously, the shift type located furthest to the front in the sequence of the block's inputs applies.

The block can shift both word and also double word operands.

Important: In certain circumstances, the value 8000 H or 8000 0000 H forbidden for arithmetic operations may be present at this block's output.

The inputs and outputs can neither be duplicated nor inverted nor negated.

E WORD/DOUBLE WORD
Operand to be shifted
The planned SHIFT operation is applied to the input operand, which is not changed.

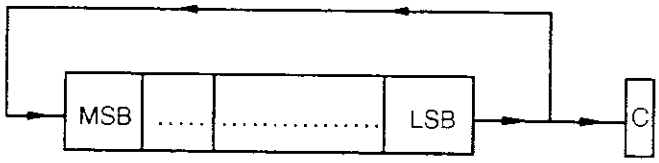
D/W BINARY
Format selection for the input operand
D/W = 0 -> WORD
D/W = 1 -> DOUBLE WORD

ANZ WORD
Number of bit positions to be shifte
Meaningful range for the quantity n:
Word operands: $0 \leq n \leq 16$
Double word operands: $0 \leq n \leq 32$

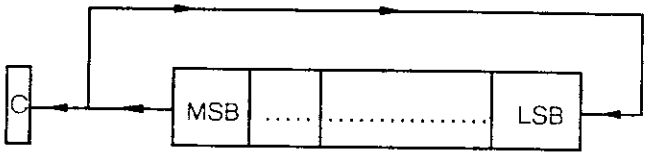
LKS BINARY
Direction in which shifting takes place
LKS = 0 -> shift to right
LKS = 1 -> shift to left

ROT BINARY
Shift type: ROTATE
The bit position released by the shift is replaced by the bit shifted out. The contents of the CARRY FLAG are additionally replaced by the bit shifted out. After the shift, the result and the contents of the CARRY FLAG are available at the block's outputs.

ROT to right:



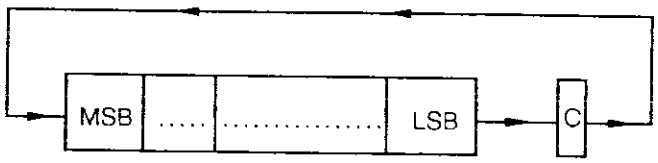
ROT to left:



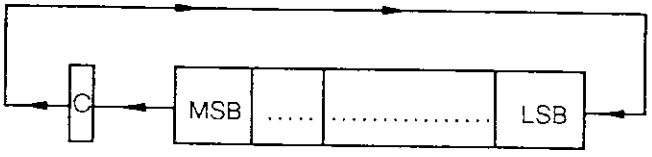
LSB: Least significant bit
MSB: Most significant bit

ROTC BINARY
Shift type: ROTATE by the CARRY FLAG
The bit position released by the shift is replaced by the contents of the CARRY FLAG. The CARRY FLAG is then replaced by the bit shifted out. After the shift, the result and the contents of the CARRY FLAG are available at the block's outputs.

ROTC to the right:



ROTC to the left:

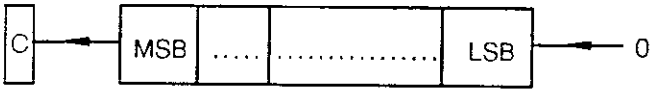


LSB: Least significant bit
MSB: Most significant bit

SLOG BINARY
Shift type: LOGICAL SHIFT
The bit position released by the shift is replaced by the value 0.
The contents of the CARRY FLAG are replaced by the bit shifted out.
After the shift, the result and the contents of the CARRY FLAG are available at the block's outputs.

SLOG to the right:

SLOG to the left:



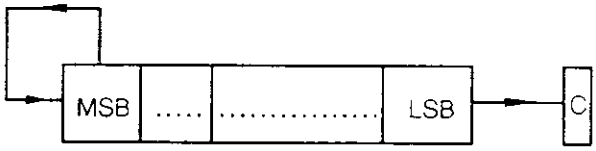
LSB: Least significant bit
MSB: Most significant bit

SARI BINARY
Shift type: ARITHMETIC SHIFT

ARITHMETIC SHIFT to the right:
The bit position 15 (MSB, sign bit) released by the shift operation is replaced by itself. The contents of the CARRY FLAG are replaced by the bit shifted out. After the shift, the result and the contents of the CARRY FLAG are available at the block's outputs

ARITHMETIC SHIFT to the left (identical with SLOG left):
The bit position 0 released by the shift is replaced by the value 0. The contents of the CARRY FLAG are replaced by the bit shifted out. After the shift, the result and the contents of the CARRY FLAG are available at the block's outputs.

SARI to the right:



SARI to the left: (Identical with SLOG to the left)

LSB: Least significant bit
MSB: Most significant bit

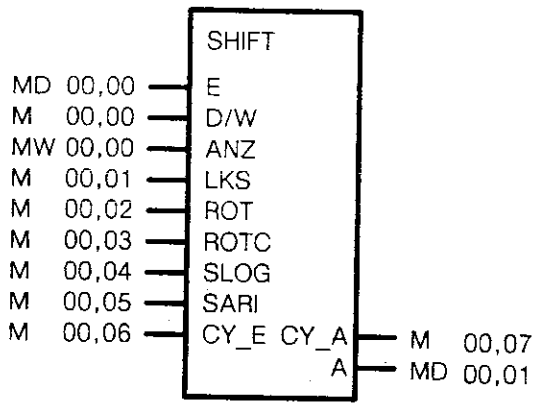
CY_E BINARY
Initial value for the CARRY FLAG with shift type ROTC. For the shift type "ROTATE by the CARRY FLAG", an initial value is needed for the CARRY FLAG. This initial value is specified at the CY_E input.

CY_A BINARY
State of the CARRY FLAG after the shift. After the shift, the current value of the CARRY FLAG is available at this output.

A WORD/DOUBLE WORD
Result of the shift
After the shift, the result is available at this output.

Example

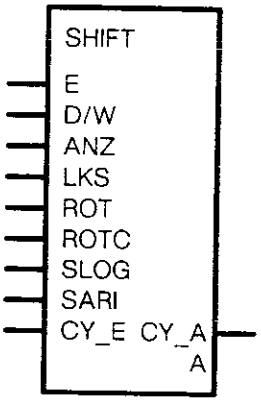
FBD/LD



IL

```
!BA    0
SHIFT
       MD 00,00
       M  00,00
       MW 00,00
       M  00,01
       M  00,02
       M  00,03
       M  00,04
       M  00,05
       M  00,06
       M  00,07
       MD 00,01
```

CE FBD Definition

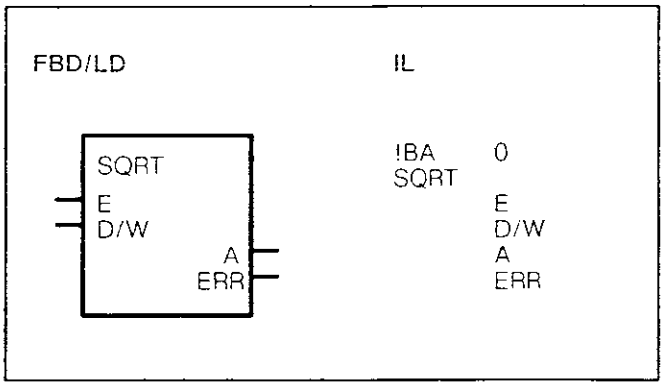


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	X	N	P	Y	0	0
D/W	E	L	N	P	Y	0	0
ANZ	E	W	N	P	Y	0	0
LKS	E	L	N	P	Y	0	0
ROT	E	L	N	P	Y	0	0
ROTC	E	L	N	P	Y	0	0
SLOG	E	L	N	P	Y	0	0
SARI	E	L	N	P	Y	0	0
CY_E	E	L	N	P	Y	0	0
CY_A	A	L	N	P	Y	0	0
A	A	X	N	P	Y	0	0

CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)
00001	SHIFT			
00002		PP 0	E	Operand to be shifted (WORD/DOUBLE WORD)
00003		PP 0	D/W	Format selection (BINARY)
00004		PP 0	ANZ	Number of bit positions (WORD)
00005		PP 0	LKS	Shift direction (BINARY)
00006		PP 0	ROT	Rotate (BINÄR)
00007		PP 0	ROTC	Rotate by the Carry flag (BINARY)
00008		PP 0	SLOG	Logical shift (BINARY)
00009		PP 0	SARI	Arithmetic shift (BINARY)
00010		PP 0	CY_E	Carry input (BINARY)
00011		PP 0	CY_A	Carry output (BINARY)
00012		PP 0	A	Result of the shift (WORD/DOUBLE WORD)

The function block SQRT generates the square root of the value at the input E. The result is available at the output A and is always *rounded down* to an integral number. The value at the input E must be a positive number. If the value at the input is negative, the value 'zero' is output through the output A and the value '1' is output through the ERR output.



Parameters

E	WORD, DOUBLE WORD	EW, AW, MW, KW, MD, KD	Input
D/W	BINARY	E, A, M, K, S	Format of the input/output
A	WORD,	EW, AW, MW, KW,	
	DOUBLE WORD	MD, KD	Square root of the input value
ERR	BINARY	E, A, M, K	Error if the input value is negative

CE Data

Runtime:	
Basic runtime:	55 µs ... 193 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T300 (PLC) V8.0 / 35 ZE 94 R0101

Description

The function block SQRT generates the square root of the value at the input E. The result is available at the output A and is always *rounded down* to an integral number. The value at the input E must be a positive number. If the value at the input is negative, the value 'zero' is output through the output A and the value '1' is output through the ERR output.

E WORD/DOUBLE WORD
The square root of the value at the input operand E is generated and is available as a value of the output operand A.

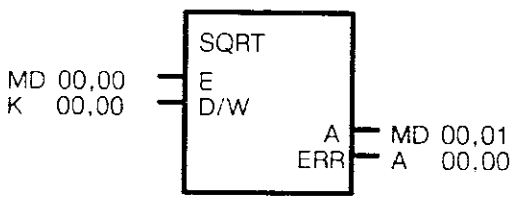
D/W BINARY
The input D/W defines the format of the input/output operand.
D/W = 0 -> WORD
D/W = 1 -> DOUBLE WORD

A WORD/DOUBLE WORD
The value of the square root is available at the output A.

ERR BINARY
The ERR output indicates whether the value of the input operand E is positive (≥ 0) or negative (< 0).
Input E $\geq$ 0 -> ERR = 0 and A = square root
Input E $<$ 0 -> ERR = 1 and A = 0

Example

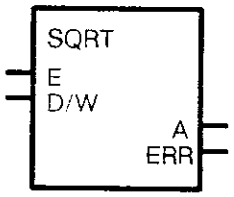
FBD/LD



IL

```
!BA    0
SQRT
       MD 00,00
       K  00,00
       MD 00,01
       A  00,00
```

CE FBD Definition

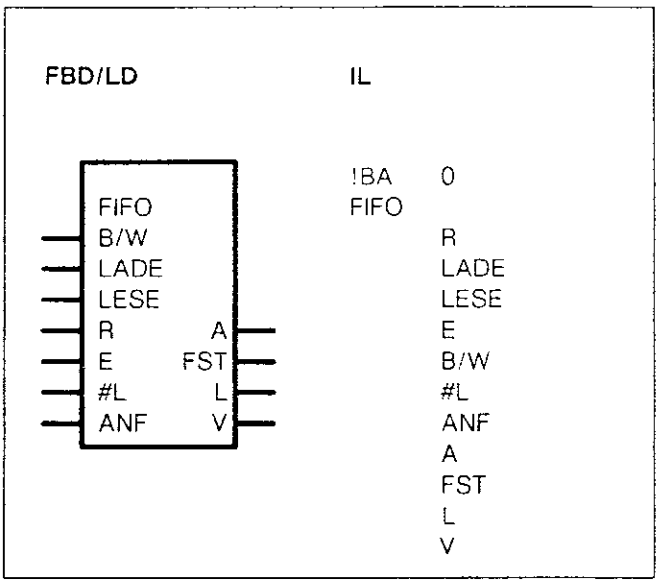


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	X	N	P	Y	0	0
D/W	E	L	N	P	Y	0	0
A	A	X	N	P	Y	0	0
ERR	A	L	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr.	Block number (pre-assigned 0)
00001	SQRT			
00002		PP 0	E	Input (WORD, DOUBLE WORD)
00003		PP 0	D/W	Format (BINARY)
00004		PP 0	A	Output (WORD, DOUBLE WORD)
00005		PP 0	ERR	Error (BINARY)

This function block realizes a stack for binary or word data from which the data written in first is again read out first (first in/first out).



Parameters

B/W	BINARY	E, A, M, S, K	Binary/word data
LADE	BINARY	E, A, M, S, K	Load FIFO
LESE	BINARY	E, A, M, S, K	Read FIFO
R	BINARY	E, A, M, S, K	Reset FIFO, 0->1 edge
E	BINARY, WORD	E, A, M, S, K EW, AW, MW, KW	Input of data in the FIFO
#L	DIRECT CONSTANT	#, #H	Number of memory locations (bytes)
ANF	BINARY WORD	E, A, M, K EW, AW, MW, KW	Start of the FIFO in the flag area
A	BINARY, WORD	E, M AW, MW	Output of data from the FIFO
FST	WORD	AW, MW	Filling level of the FIFO
L	BINARY	A, M	FIFO empty
V	BINARY	A, M	FIFO full

VE-Data

Runtime:	
Basic runtime:	64 µs
Additional runtime:	0 ... 114 µs according to the function to be carried out
Output updating:	no
Number of historical values:	4 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block realizes a stack for binary or word data from which the data written in first is again read out first (first in/first out).

The inputs and outputs can neither be duplicated nor negated/inverted.

B/W BINARY

The input BW serves to preselect binary or word processing (FIFO for binary or word data)

B/W = 0 -> word processing
B/W = 1 -> binary processing

Binary or word processing is defined once and must not be changed during operation of the FIFO.

LADE BINARY

The value present at the input E is transferred to the next free position of the FIFO by means of a 1 signal at the LADE input.
If the FIFO is "full" and a "load" signal is present, the new value will *not* be read in. A new value can only be read if a value has been read out first. It is then transferred to the first position of the FIFO.

LESE BINARY

A 1 signal at the input LESE results in output of the current FIFO value through the output A. If the FIFO is "empty", a pending read order is ignored and the value 0 is output through the output A. The output A is always set to 0 if there is no read order.

LADE and **LESE**

If load and read orders are present simultaneously, the value to be loaded is forwarded directly from the input E to the output A provided the FIFO is empty.
If the FIFO is not empty, the current FIFO value is output and the new load information is transferred to it.
The FIFO does not change its filling level in this mode. The value output through the output FST remains constant. If the FIFO is empty, the output L remains 1 and the output FST is set to 0.

R BINARY

A 0->1 edge at the input R results in the reset of the block. Therefore, values read in before are no longer available.

The output L assumes the value 1 and the outputs FST and A assume the value 0.

R and **LADE**

If R and LADE signals are present simultaneously the reset is performed first and then directly afterwards the load operation. Therefore, the FIFO is reset and the new value to be read in is then immediately stored in the FIFO as the first value.

R and **LADE** and **LESE**

The value at the input E is forwarded directly to the output A. The output L is set permanently to 1 and the output FST is set permanently to 0.

E BINARY/WORD

The value to be transferred into the FIFO is specified at the input E.

#L DIRECT CONSTANT

The number of required memory locations (bytes) of the FIFO is specified at the input A. This quantity is specified as a direct constant and results from the following formula:

BINARY data:
#L = Number of binary values to be stored

WORD data:
#L = 2 * Quantity of word values to be stored

E.g.: BINARY data: 3 values -> #L = 3
WORD data: 3 values -> #L = 6

The FIFO length parameter is subjected to a plausibility check for the value 0 and, in word processing mode, also in relation to an odd byte parameter. If the parameter specified at the input #L is incorrect, the FIFO will assume the initial state (as after R).

ANF BINARY/WORD

The FIFO memory start address is specified as a binary or word flag at the input ANF. The FIFO begins with the specified flag.

A BINARY/WORD

When the FIFO is read, the current value is available at the output A. If *no* read order is available, the value 0 is output.

FST WORD

The output FST indicates the current filling level of the FIFO at any time. The filling level is the number of binary or word values stored in the FIFO.

L BINARY

The output L indicates whether or not the FIFO is empty.
L = 0 -> FIFO is not empty
L = 1 -> FIFO is empty

V BINARY

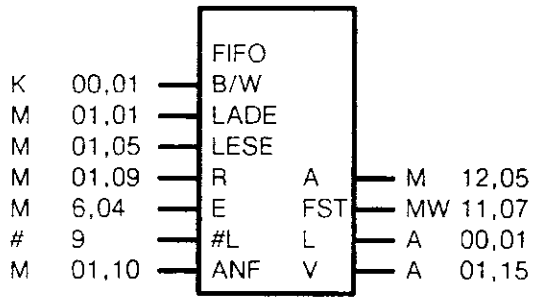
The output V indicates whether or not the FIFO is full.
V = 0 -> FIFO is not full
V = 1 -> FIFO is full

No more values can be read in if the FIFO is full. A value can only be read in once a value has been read out. This then takes place as from the start of the FIFO.

Example

FIFO for 9 binary data, stored as of M 01,10

FBD/LD

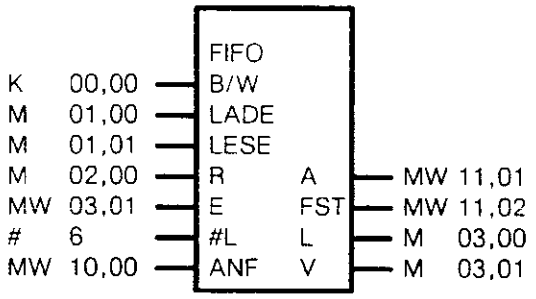


IL

```
!BA      0
FIFO
         M   01,09
         M   01,01
         M   01,05
         M   06,04
         K   00,01
         #   9
         M   01,10
         M   12,05
         MW  11,07
         A   00,01
         A   01,15
```

FIFO for 3 word data, stored as of MW 10,00

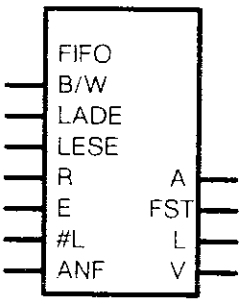
FBD/LD



IL

```
!BA      0
FIFO
         M   02,00
         M   01,00
         M   01,01
         MW  03,01
         K   00,00
         #   6
         MW  10,00
         MW  11,01
         MW  11,02
         M   03,00
         M   03,01
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
B/W	E	L	N	P	Y	0	0
LADE	E	L	N	P	Y	0	0
LESE	E	L	N	P	Y	0	0
R	E	L	N	P	Y	0	0
E	E	X	N	P	Y	0	0
#L	K	W	N	P	Y	0	0
ANF	E	X	N	P	Y	0	0
A	A	X	N	P	Y	0	0
FST	A	W	N	P	Y	0	0
L	A	L	N	P	Y	0	0
V	A	L	N	P	Y	0	0

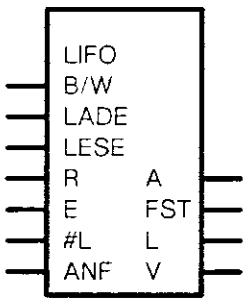
CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)
00001	FIFO			
00002		PP 0	R	Input BINARY (Reset)
00003		PP 0	LADE	Input BINARY (load FIFO)
00004		PP 0	LESE	Input BINARY (read FIFO)
00005		PP 0	E	Input BINARY/WORD (input)
00006		PP 0	B/W	Input BINARY (BINARY/WORD data)
00007		PP 0	#L	# CONSTANT (number of memory locations)
00008		PP 0	ANF	Input BINARY/WORD (start of memory locations)
00009		PP 0	A	Output BINARY/WORD (ouput)
00010		PP 0	FST	Output WORD (filling level)
00011		PP 0	L	Output BINARY (FIFO empty)
00012		PP 0	V	Output BINARY (FIFO full)

STACK, LAST IN/FIRST OUT — LIFO

This function block realizes a stack for binary or word data from which the data written in last is read out first (last in/first out).

FBD/LD



IL

```
IBA   0
LIFO
      R
      LADE
      LESE
      E
      B/W
      #L
      ANF
      A
      FST
      L
      V
```

Parameters

B/W	BINARY	E, A, M, S, K	Binary data/word data
LADE	BINARY	E, A, M, S, K	Load LIFO
LESE	BINARY	E, A, M, S, K	Read LIFO
R	BINARY	E, A, M, S, K	Reset LIFO, 0/1 edge
E	BINARY	E, A, M, S, K	Enter data in the LIFO
	WORD	EW, AW, MW, KW	
#L	DIRECT CONSTANT	#, #H	Number of memory locations (bytes)
ANF	BINARY	E, A, M, K	Start of the LIFO in the flag area
	WORD	EW, AW, MW, KW	
A	BINARY	E, M	Output of data from the LIFO
	WORD	AW, MW	
FST	WORD	AW, MW	Filling level of the LIFO
L	BINARY	A, M	LIFO empty
V	BINARY	A, M	LIFO full

CE Data

Runtime:	
Basic runtime:	64 µs
Additional runtime:	0–114 µs according to the function to be carried out
Output updating:	no
Number of historical values:	3 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block realizes a stack for binary or word data from which the data written in last is read out first (last in/first out).

The inputs and outputs can neither be duplicated nor negated/inverted.

B/W BINARY

The input B/W serves to preselect binary or word processing (LIFO for binary or word data)

B/W = 0 -> word processing
B/W = 1 -> binary processing

Binary or word processing is defined once and must not be changed during operation of the LIFO.

LADE BINARY

The value present at the input E is transferred to the next free position of the LIFO by means of a 1 signal at the LADE input. If the LIFO is "full" and a "load" signal is present, the new value will *not* be read in. A new value can only be read in if a value has been read out first. It is then transferred to the first position of the LIFO.

LESE BINARY

A 1 signal at the input LESE results in output of the current LIFO value through the output A. If the LIFO is "empty", a pending read order is ignored and the value 0 is output through the output A. The output A is always set to 0 if there is no read order.

LADE and LESE

If load and read orders are present simultaneously, the value to be loaded is forwarded directly from the input E to the output A. At the same time, the LIFO does not store the value. The LIFO does not change its filling level in this mode. The value output through the output FST remains constant. If the LIFO is empty, the output L remains 1 and the output FST is set to 0.

R BINARY

A 0->1 edge at the input R results in the reset of the block. Therefore, values read in before are no longer available. The output L assumes the value 1 and the outputs FST and A assume the value 0.

R and LADE

If R and LADE signals are present simultaneously the reset is performed first and then directly afterwards the load operation. Therefore, the LIFO is reset and the new value to be read in is then immediately stored in the LIFO as the first value.

R and LADE and LESE

The value at the input E is forwarded directly to the output A. The output L is set permanently to 1 and the output FST is set permanently to 0.

E BINARY/WORD

The value to be transferred into the LIFO is specified at the input E.

#L DIRECT CONSTANT

The number of required memory locations (bytes) of the LIFO is specified at the input #L. This quantity is specified as a direct constant and results from the following formula:

BINARY data:
#L = Number of binary values to be stored

WORD data:
#L = 2 * Quantity of word values to be stored

E.g.: BINARY data: 3 values -> #L = 3
WORD data: 3 values -> #L = 6

The LIFO length parameter is subjected to a plausibility check for the value 0 and, in word processing mode, also in relation to an odd byte parameter. If the parameter specified at the input #L is incorrect, the LIFO will assume the initial state (as after R).

ANF BINARY/WORD

The LIFO memory start address is specified as a binary or word flag at the input ANF. The LIFO begins with the specified flag.

A BINARY/WORD

When the LIFO is read, the current value is available at the output A. If *no* read order is available, the value 0 is output.

FST WORD

The output FST indicates the current filling level of the LIFO at any time. The filling level is the number of binary or word values stored in the LIFO.

L BINARY

The output L indicates whether or not the LIFO is empty.
L = 0 -> LIFO is not empty
L = 1 -> LIFO is empty

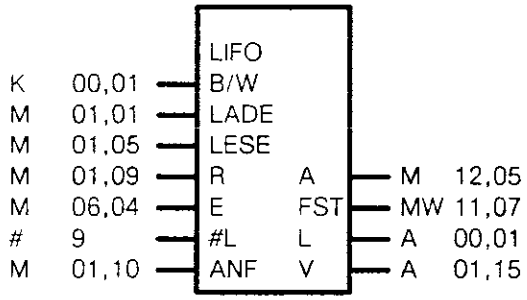
V BINARY

The output V indicates whether or not the LIFO is full.
V = 0 -> LIFO is not full
V = 1 -> LIFO is full
No more values can be read in if the LIFO is full. A value can only be read in again once a value has been read out. This then takes place as from the start of the LIFO.

Example

LIFO for 9 binary data, stored as of M 01,10

FBD/LD

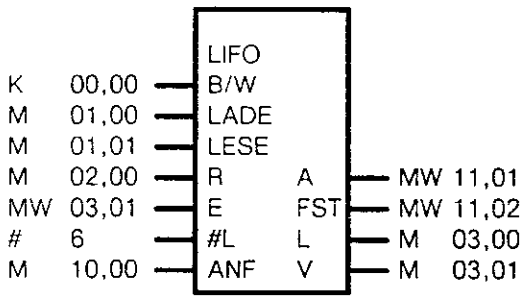


IL

```
!BA      0
LIFO
         M   01,09
         M   01,01
         M   01,05
         M   06,04
         K   00,01
         #    9
         M   01,10
         M   12,05
         MW  11,07
         A   00,01
         A   01,15
```

LIFO for 3 word data, stored as of MW 10,00

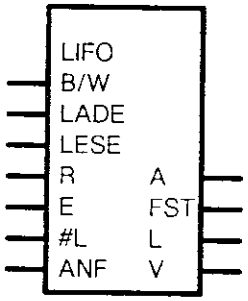
FBD/LD



IL

```
!BA      0
LIFO
         M   02,00
         M   01,00
         M   01,01
         MW  03,01
         K   00,00
         #    6
         MW  10,00
         MW  11,01
         MW  11,02
         M   03,00
         M   03,01
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
B/W	E	L	N	P	Y	0	0
LADE	E	L	N	P	Y	0	0
LESE	E	L	N	P	Y	0	0
R	E	L	N	P	Y	0	0
E	E	X	N	P	Y	0	0
#L	K	W	N	P	Y	0	0
ANF	E	X	N	P	Y	0	0
A	A	X	N	P	Y	0	0
FST	A	W	N	P	Y	0	0
L	A	L	N	P	Y	0	0
V	A	L	N	P	Y	0	0

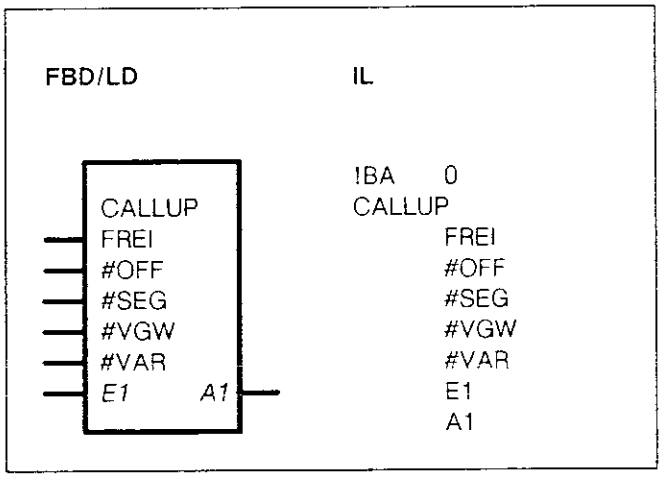
CE IL Definition

00000	IBA	0	Nr	Block No. (presest to 0)
00001	LIFO			
00002		PP 0	R	Input BINARY (R)
00003		PP 0	LADE	Input BINARY (load LIFO)
00004		PP 0	LESE	Input BINARY (read LIFO)
00005		PP 0	E	Input BINARY/WORD (input)
00006		PP 0	B/W	Input BINARY (BINARY/WORD data)
00007		PP 0	#L	# CONSTANT (number of memory locations)
00008		PP 0	ANF	Input BINARY/WORD (start of memory locations)
00009		PP 0	A	Output BINARY/WORD (output)
00010		PP 0	FST	Output WORD (filling level)
00011		PP 0	L	Output BINARY (LIFO empty)
00012		PP 0	V	Output BINARY (LIFO full)

SUBROUTINE CALL FOR AN ASSEMBLER PROGRAM — CALLUP

This function block calls a subroutine written in assembler.

After processing of the subroutine, the program branches back to the block. Before the subroutine is called, all registers are automatically saved and restored after the return.



Parameters

FREI	BINARY	E, M, A, K, S	Block enable
#OFF	DIRECT CONSTANT	#, #H	Offset address of the subroutine
#SEG	DIRECT CONSTANT	#, #H	Segment address of the subroutine
#VGW	DIRECT CONSTANT	#, #H	Number of historical values of the subroutine
#VAR	DIRECT CONSTANT	#, #H	Number of input/output variables of the subroutine
E1	X	E, M, A, K, S, EW, MW, AW, KW, MD, KD	Input variable (capable of duplication)
A1	X	M, A, MW, AW,MD	Output variable (capable of duplication)

CE Data

Runtime:	
Basic runtime:	29 µs
Additional runtime:	---
Output updating:	will be determined in the assembler program
Number of historical values:	Number of #VGW words
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block calls a subroutine written in assembler.

After processing of the subroutine, the program branches back to the block. Before the subroutine is called, all registers are automatically saved and restored after the return.

The input E1 and the output A1 are capable of duplication. The inputs and outputs can neither be negated nor inverted.

FREI BINARY

Enable function block processing
FREI = 0: Block is not processed
FREI = 1: Block is processed

#OFF DIRECT CONSTANT
#SEG DIRECT CONSTANT

The start address of the assembler subroutine is specified at these two inputs. The start address consists of an offset and a segment address. The offset and segment addresses are specified as direct constants.

#VGW DIRECT CONSTANT

The number of historical values used in the subroutine is specified at the input #VGW. This is specified as a direct constant.

#VAR DIRECT CONSTANT

The total number of input and output variables of the subroutine is specified at the input #VAR. Therefore, this is the total of all E1...En and A1...An. This is specified as a direct constant.

E1 BINARY, WORD, DOUBLE WORD

The input E1 is capable of duplication (E1...En).

The input variables for the subroutine are specified at the inputs E1...En. The subroutine performs "reading" access to these variables. When the subroutine accesses these variables, the assembler programmer must pay attention to the defined data format. Direct constants (value transfer) are not allowed at the inputs E1...En.

A1 BINARY, WORD, DOUBLE WORD

The output A1 can be duplicated (A1...An).

The output variables for the subroutine are specified at the outputs A1...An. The subroutine performs "writing" access to these variables, i.e. values calculated in the subroutine are provided to the main program with the aid of the output variables A1...An. The assembler programmer must pay attention to the defined data format when accessing these variables.

The function block's inputs and outputs can neither be inverted nor negated.

Application notes

Memory space

In the test phase, the subroutine can be stored in the comment RAM or in the user program RAM (e.g. with emulator, HEX file loader or PLC monitor). It can be loaded to an EPROM once it has been tested. There are various possibilities of doing this:

a) Comment memory (only in the case of 35 ZE 93 or 35 ZE 94 with a small program memory)
The subroutine is located in the comment RAM 2 and is stored in the comment EPROMs when they are produced.

Disadvantage: Comment 2 cannot be used
Memory space: C000:0010H to C000:FFF0H

b) User program memory
The subroutine is located in the user program memory after the end of user program 1 or user program 2. It is stored on the user program EPROMs when they are created together with the user program.

Disadvantage: Part of user program 1 or 2 cannot be used (address range of the user program RAMs: see PLC description).

c) System software-EPROM

The tested subroutine is loaded in EPROMs together with the system software. This possibility is only meaningful for internal ABB users because it is absolutely necessary to contact the development department owing to the necessary free memory space in the system EPROMs. To do this, the checksum must be recalculated and also stored. Checksum calculation is done separately for each of the two system software EPROMs. The last memory word (= old checksum, address FFFEH) must be overwritten with the value FFFFH in the EPROM programmer. Checksum calculation is then done throughout the EPROM from address 0000H to FFFFH. This checksum word is then loaded to FFFEH and programmed in the system EPROM (FFFEH: low byte; FFFFH: high byte).

Subroutine creation

1. Register

When creating the subroutine, use can be made of the following registers to suit requirements:

AX, BX, CX, DX, BP

The following registers are intended for access to variables:

DS, ES, DI, SI

However, these registers can also be changed because they are corrected again after execution of the subroutine in the CALLUP block.

2. Start address

Subroutine's start address must be specified at the #OFF and #SEG inputs in the call of the block. The offset and segment are each specified as direct constants.

3. Access to input and output variables

Access to the variables is by way of the pointers ES:SI and DS:BX.

When the subroutine is entered, the pointer to the address of the first variable in the call of the block is located in (ES:SI) and the pointer to the address of the second variable is located in (ES:SI+2) etc. Access to the contents of these variables is then by means of

DS:BX

Example 1:

The values of the first two *word variables* in the block call are to be fetched to the registers AX and CX.

```
MOV   BX,ES:[SI]    ;Fetch address for 1st va-
                    ;riable and store in BX
MOV   AX,DS:[BX]    ;fetch value to AX
MOV   BX,ES:[SI+2]  ;fetch address for second
                    ;variable and store in BX
MOV   CX,DS:[BX]    ;fetch value to CX
```

Example 2:

The value of the first two *binary variables* in the call of the block are to be fetched to the registers AL and CL.

```
MOV   BX,ES:[SI]    ;Fetch address for 1st va-
                    ;riable and store in BX
MOV   AL,DS:[BX]    ;Fetch value to AL
MOV   BX,ES:[SI+2]  ;Fetch address for second
                    ;variable and store in BX
MOV   CL,DS:[BX]    ;Fetch value to CL
```

Binary variables occupy a whole byte. This byte may only have 0 or 1 as its contents.

4. Access to historical values/RAM

If RAM space is needed in the subroutine for intermediate values, use can be made of the historical values memory.

The intermediate values are retained and are again available in the next program cycle for further processing. For each call of the subroutine, its own memory locations are available in the historical values memory. If a subroutine is called three times, for instance, in total 3 times as many historical values are occupied in the historical values memory as are specified in the call. The number of memory

words used (1 word = 16 bits) in the historical values memory must be specified as a direct constant in the call of the block. If too many historical values are occupied, the PLC issues an error message when the command PA (program editing) is used or when the program is started.

The historical values are addressed by way of the pointers
DS:DI
In doing so, DS:DI points to the first available historical value when entering the subroutine.

Example:
The value fetched to register AX in point 3. is to be compared to the first word in the historical values memory

```
CMP   AX,DS:[DI]      ;Compare value to
                      ;historical value
```

5. The stack has a depth of 40 words. In this way, up to 40 PUSH commands can be executed successively.

6. Return from the subroutine
   The return is achieved by means of **RET FAR** (8086 machine code: CB). **RET FAR** is generated by the 8086 assembler by virtue of the fact that the subroutine is declared as **PROC FAR.**

Example

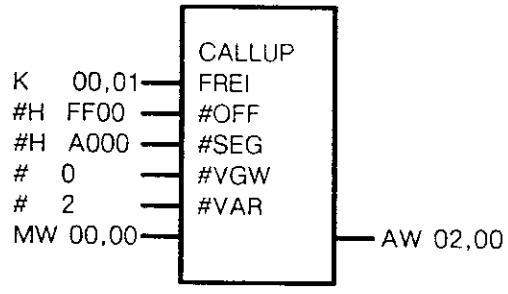
In a subroutine, the contents of the flag word MW 01,00 are to be incremented and the contents of the flag word MW 02,00 are to be decremented.

Subroutine

```
TEST   PROC FAR

MOV    BX,ES:[SI]     ;Address of MW 01,00
                      ;to BX
MOV    AX,DS:[SI]     ;Load value from MW 01,00
                      ;to register AX
INC    AX             ;Increment value
MOV    DS:[BX],AX     ;Again write value back to
                      ;MW 01,00
MOV    BX,ES:[SI+2]   ;Address of MW 02,00
                      ;to BX
MOV    AX,DS:[BX]     :Load value from MW 02,00
                      ;to register AX
DEC    AX             ;Decrement value
MOV    DS:[BX],AX     ;Again write value back
                      ;to MW 02,00
RET
TEST   ENDP
```

The subroutine should begin at address A000:FF00H in the program memory 2. It must be enabled for execution (K00,01=1). No historical values are needed and 2 variables are used.

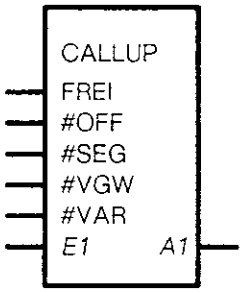
FBD/LD



IL

```
!BA          0
CALLUP
       K    00,01
       #H   FF00
       #H   A000
       #    0
       #    2
       MW   00,00
       AW   02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
#OFF	K	W	N	P	Y	0	0
#SEG	K	W	N	P	Y	0	0
#VGW	K	W	N	P	Y	0	0
#VAR	K	W	N	P	Y	0	0
E1	E	X	N	P	N	1	0
A1	A	X	N	P	N	2	0

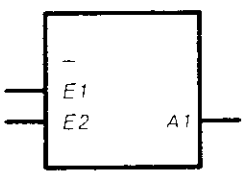
CE IL Definition

```
00000   !BA      0          Nr      Block No. (preset to 0)
00001   CALLUP
00002            PP  0      FREI    Input BINARY (block enable)
00003            PP  0      #OFF    # CONSTANT (offset address)
00004            PP  0      #SEG    # CONSTANT (segment address)
00005            PP  0      #VGW    # CONSTANT (number of historical values)
00006            PP  0      #VAR    # CONSTANT (number of variables)
        [        1
00007            PP  1      E1      Input ALL
        ]        1
        [        2
00008            PP  2      A1      Output ALL
        ]        2
```

SUBTRACTION

The value of the operand at the input E2 is subtracted from the value of the operand at the input E1 and the result is allocated to the operand at the output A1.

FBD/LD



IL

```
!   E1
-   E2
=   A1
```

Parameters

E1	WORD	EW, MW, AW, KW	Minuend
E2	WORD	EW, MW, AW, KW	Subtrahend; the input can be duplicated
A1	WORD	MW, AW	Result (difference)

CE Data

Runtime:	
Basic runtime:	< 12 µs
Additional runtime:	5 µs per additional input (E3 ... En)
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The value of the operand at the input E2 is subtracted from the value of the operand at the input E1 and the result is allocated to the operand at the output A1.

The input E2 is capable of duplication (E2...En). If it is duplicated, all values of the operands at the inputs E2...En are subtracted from the operand at the input E1.

The input E1 and the output A1 can be negated.

Number range

Integer word (16 bits)

The following specially applies here to *non-negated* input E1:
If the inadmissible value 8000 H (-32768) is present at E1, it is automatically corrected to the permissible value 8001 H (-32767) before processing.

The following generally applies:

- Low limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

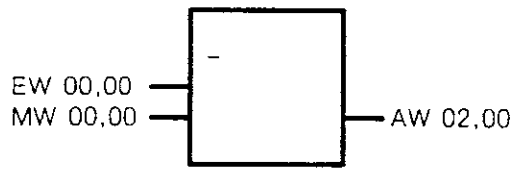
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

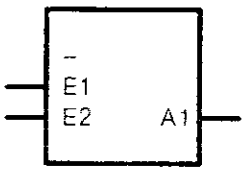
FBD/LD



IL

```
!    EW 00,00
–    MW 00,00
=    AW 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	Y	P	N	0	0
E2	E	W	N	P	N	1	0
A1	A	W	Y	P	N	0	0

CE IL Definition

00000	!	PP 0	E1	Input WORD
	[	1		
00002	–	PP 1	E2	Input WORD (capable of duplication)
	]	1		
00005	=	PP 0	A1	Output WORD

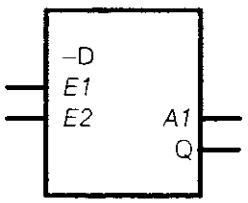
SUBTRACTION, DOUBLE WORD -D

The value of the operand at the input E2 is subtracted from the value of the operand at the input E1 and the result is allocated to the operand at the output A1.

The result is limited to the maximum or minimum value of the number range. If limiting has taken place, a 1 signal is allocated to the binary operand at the output Q. If no limiting has taken place, a 0 signal is allocated to the binary operand at the output Q.

The value of the operand at the input E2 is checked before subtraction to determine whether or not it lies outside of the permissible number range (8000 0000 H). If this is the case, calculation is done with the value -2 147 483 647 (8000 0001 H) instead of this inadmissible value.

FBD/LD



IL

```
!BA   0
SUBD
      E1
      E2
      A1
      Q
```

Parameters

E1	DOUBLE WORD	MD, KD	Minuend
E2	DOUBLE WORD	MD, KD	Subtrahend
A1	DOUBLE WORD	MD	Result (difference)
Q	BINARY	A, M	Result, limited

CE Data

Runtime:	
Basic runtime:	50 ... 55 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

The value of the operand at the input E2 is subtracted from the value of the operand at the input E1 and the result is allocated to the operand at the output A1.

The result is limited to the maximum or minimum value of the number range. If limiting has taken place, a 1 signal is allocated to the binary operand at the output Q. If no limiting has taken place, a 0 signal is allocated to the binary operand at the output Q.

The value of the operand at the input E2 is checked before subtraction to determine whether or not it lies outside of the permissible number range (8000 0000 H). If this is the case, calculation is done with the value -2 147 483 647 (8000 0001 H) instead of this inadmissible value.

The inputs and outputs can neither be duplicated nor negated.

Number range

Integer double word (32 bits).

The following especially applies here to the input E1:

- Low limit: 8000 0000 H -2 147 483 647.

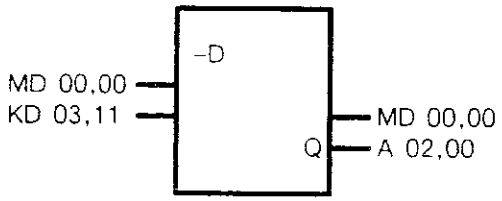
The following generally applies:

Integer double word (32 bits)

• Low limit:	8000 0001 H	-2 147 483 647
• High limit:	7FFF FFFF H	+2 147 483 647
• Inadmissible value:	8000 0000 H	---

Example

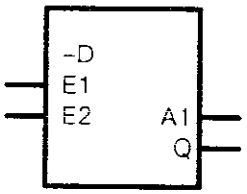
FBD/LD



IL

```
!BA   0
SUBD
      MD 00,00
      KD 03,11
      MD 00,00
      A  02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
E2	E	D	N	P	N	0	0
A1	A	D	N	P	N	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

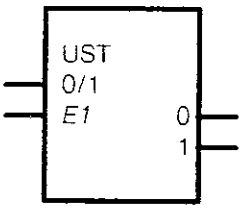
00000	!BA	0	Nr	Block No. (preset to 0)
00001	SUBD			
00002		PP 0	E1	Input DOUBLE WORD
00003		PP 0	E2	Input DOUBLE WORD
00004		PP 0	A1	Output DOUBLE WORD
00005		PP 0	Q	Output BINARY

A 0 signal at the binary input 0/1 allocates the value of the word operand at the input E1 to the word operand at the output 0.

A 1 signal at the binary input 0/1 allocates the value of the word operand at the input E1 to the word operand at the output 1.

The respective output that is not allocated retains its old value, but the old value is not updated.

FBD/LD



IL

```
!BA   0
UST
      0/1
      E1
      0
      1
```

Parameters

0/1	BINARY	E, M, A, S, K	Switchover input
E1	WORD	EW, MW, AW, KW	Input
0	WORD	AW, MW	Output for 0/1 = 0
1	WORD	AW, MW	Output for 0/1 = 1

CE Data

Runtime:	
Basic runtime:	31 µs
Additional runtime:	---
Output updating:	no
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R201

Description

A 0 signal at the binary input 0/1 allocates the value of the word operand at the input E1 to the word operand at the output 0.

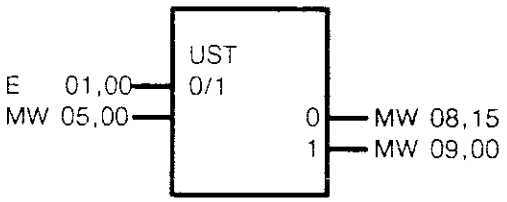
A 1 signal at the binary input 0/1 allocates the value of the word operand at the input E1 to the word operand at the output 1.

The respective output that is not allocated retains its old value, but the old value is not updated.

The inputs and outputs can neither be duplicated nor negated/inverted.

Example

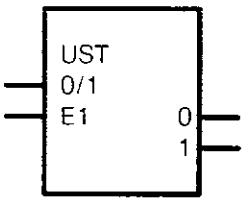
FBD/LD



IL

```
!BA   0
UST
      E   01,00
      MW 05,00
      MW 08,15
      MW 09,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0/1	E	L	N	P	Y	0	0
E1	E	W	N	P	N	0	0
0	A	W	N	P	Y	0	0
1	A	W	N	P	Y	0	0

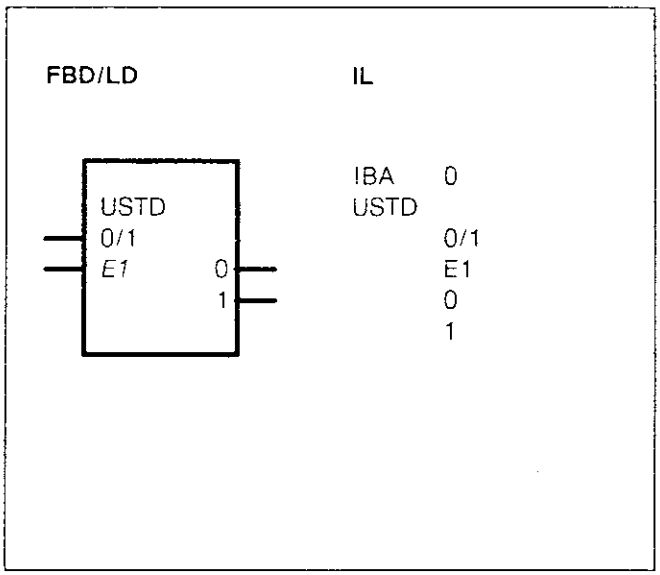
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	UST			
00002		PP 0	0/1	Input BINARY (selector)
00003		PP 0	E1	Input WORD
00004		PP 0	0	Output WORD
00005		PP 0	1	Output WORD

A 0 signal at the binary input 0/1 allocates the value of the double word operand at the input E1 to the double word operand at the output 0.

A 1 signal at the binary input 0/1 allocates the value of the double word operand at the input E1 to the double word operand at the output 1.

The respective output that is not allocated retains its old value, but the old value is not updated.



Parameters

0/1	BINARY	E, M, A, S, K	Switchover input
E1	DOUBLE WORD	MD, KD	Input
0	DOUBLE WORD	MD	Output for 0/1 = 0
1	DOUBLE WORD	MD	Output for 0/1 = 1

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	---
Output updating:	no
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

A 0 signal at the binary input 0/1 allocates the value of the double word operand at the input E1 to the double word operand at the output 0.

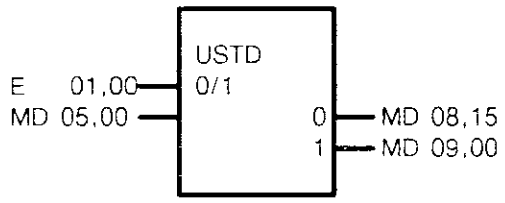
A 1 signal at the binary input 0/1 allocates the value of the double word operand at the input E1 to the double word operand at the output 1.

The respective output that is not allocated retains its old value, but the old value is not updated.

The inputs and outputs can neither be duplicated nor negated/inverted.

Example

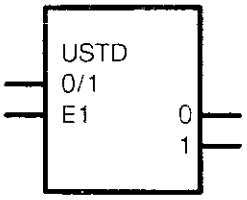
FBD/LD



IL

```
!BA   0
USTD
      E    01,00
      MD 05,00
      MD 08,15
      MD 09,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0/1	E	L	N	P	Y	0	0
E1	E	D	N	P	N	0	0
0	A	D	N	P	Y	0	0
1	A	D	N	P	Y	0	0

CE IL Definition

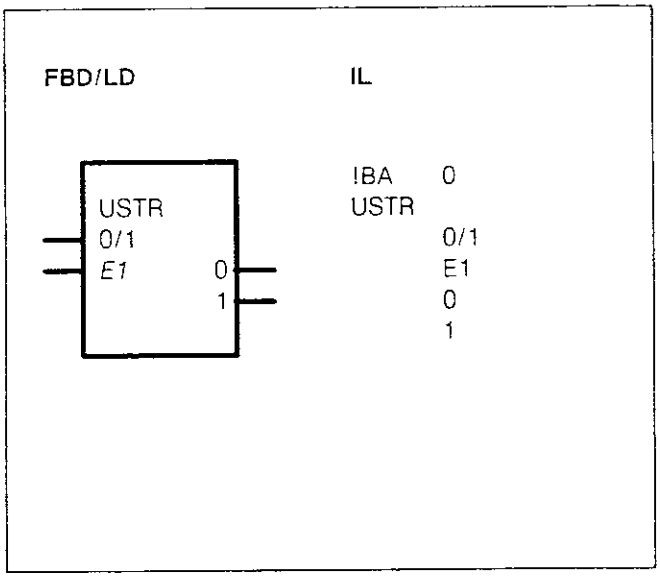
00000	!BA	0	Nr	Block No. (preset to 0)
00001	USTD			
00002		PP 0	0/1	Input BINARY (selector)
00003		PP 0	E1	Input DOUBLE WORD
00004		PP 0	0	Output DOUBLE WORD
00005		PP 0	1	Output DOUBLE WORD

SWITCHOVER GATE WITH RESET — USTR

A 0 signal at the binary input 0/1 allocates the value of the word operand at the input E1 to the word operand at the output 0.

A 1 signal at the binary input 0/1 allocates the value of the word operand at the input E1 to the word operand at the output 1.

The respective output that is not allocated is set to 0.



Parameters

0/1	BINARY	E, M, A, S, K	Switchover input
E1	WORD	EW, MW, AW, KW	Input
0	WORD	AW, MW	Output for 0/1 = 0
1	WORD	AW, MW	Output for 0/1 = 1

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R201

Description

A 0 signal at the binary input 0/1 allocates the value of the word operand at the input E1 to the word operand at the output 0.

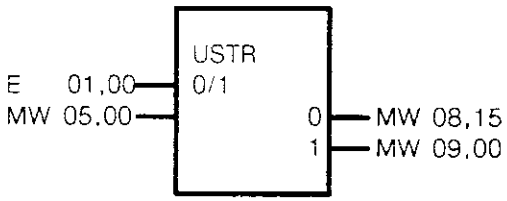
A 1 signal at the binary input 0/1 allocates the value of the word operand at the input E1 to the word operand at the output 1.

The respective output that is not allocated is set to 0.

The inputs and outputs can neither be duplicated nor negated/inverted.

Example

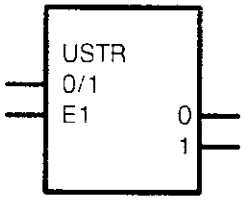
FBD/LD



IL

```
!BA   0
USTR
      E   01,00
      MW 05,00
      MW 08,15
      MW 09,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0/1	E	L	N	P	Y	0	0
E1	E	W	N	P	N	0	0
0	A	W	N	P	Y	0	0
1	A	W	N	P	Y	0	0

CE IL Definition

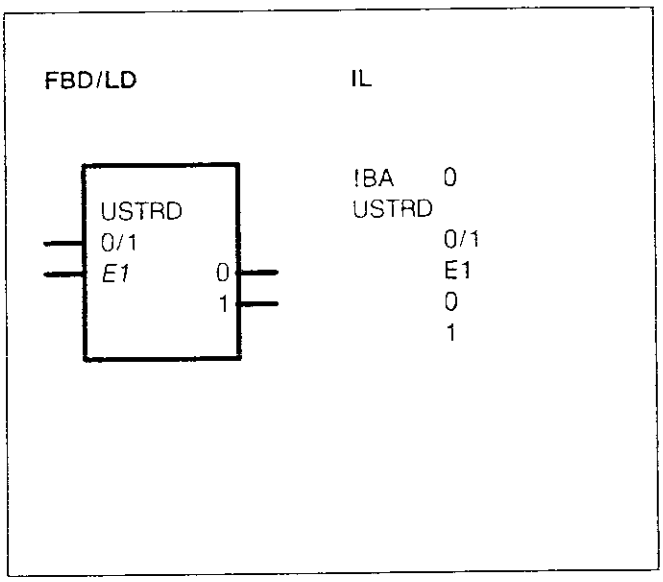
00000	!BA	0	Nr	Block No. (preset to 0)
00001	USTR			
00002		PP 0	0/1	Input BINARY (selector)
00003		PP 0	E1	Input WORD
00004		PP 0	0	Output WORD
00005		PP 0	1	Output WORD

SWITCHOVER GATE WITH RESET, DOUBLE WORD — USTRD

A 0 signal at the binary input 0/1 allocates the value of the double word operand at the input E1 to the double word operand at the output 0.

A 1 signal at the binary input 0/1 allocates the value of the double word operand at the input E1 to the double word operand at the output 1.

The respective output that is not allocated is set to 0.



Parameters

0/1	BINARY	E, M, A, S, K	Switchover input
E1	DOUBLE WORD	MD, KD	Input
0	DOUBLE WORD	MD	Output for 0/1 = 0
1	DOUBLE WORD	MD	Output for 0/1 = 1

CE Data

Runtime:	
Basic runtime:	43 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 81 R301 / 35 ZE 93 R201

Description

A 0 signal at the binary input 0/1 allocates the value of the double word operand at the input E1 to the double word operand at the output 0.

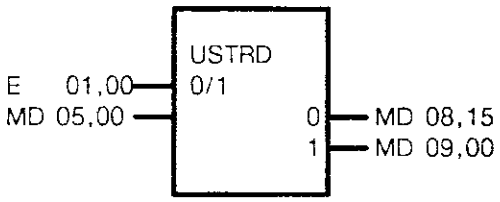
A 1 signal at the binary input 0/1 allocates the value of the double word operand at the input E1 to the double word operand at the output 1.

The respective output that is not allocated is set to 0.

The inputs and outputs can neither be duplicated nor negated/inverted.

Example

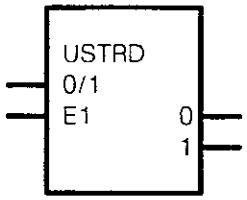
FBD/LD



IL

```
!BA    0
USTRD
       E    01,00
       MD 05,00
       MD 08,15
       MD 09,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
0/1	E	L	N	P	Y	0	0
E1	E	D	N	P	N	0	0
0	A	D	N	P	Y	0	0
1	A	D	N	P	Y	0	0

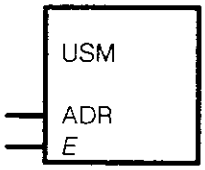
CE IL Definition

00000	!BA	0	Nr	Block No. (presest to 0)
00001	USTRD			
00002		PP 0	0/1	Input BINARY (selector)
00003		PP 0	E1	Input DOUBLE WORD
00004		PP 0	0	Output DOUBLE WORD
00005		PP 0	1	Output DOUBLE WORD

This function block allocates the value from the input E to an operand, using the method of indirect addressing.

Note: The USM block can only be used meaningfully in conjunction with the ADRWA block.

FBD/LD



IL

```
!BA   0
USM
      ADR
      E
```

Parameters

ADR	WORD	EW, AW, MW, KW	Indirect address of the operand to be written
E	WORD	AW, MW	Value to be allocated to the operand

CE Data

Runtime:	
Basic runtime:	225 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V4 / 935 PC 82 R101 / 35 ZE 93 R101

Description

This function block allocates the value from the input E to an operand, using the method of indirect addressing.

Note: The USM block can only be used meaningfully in conjunction with the ADRWA block.

The value of the operand at the input ADR is interpreted as the address of the operand to be written (indirect addressing).

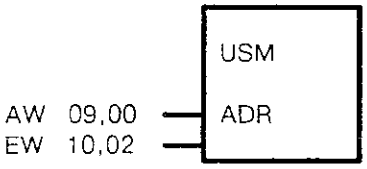
Therefore, the operand at the input ADR and its value represent an indirect address. This indirect address is generated by the ADRWA function block.

The inputs can neither be duplicated nor negated.

Note: Refer to the function block ADRWA for an explanation of the method of indirect addressing and the possibilities of using the USM function block.

Example

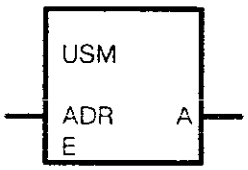
FBD/LD



IL

```
IBA   0
USM
      AW  09,00
      EW  10,02
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
ADR	E	W	N	P	Y	0	0
E	E	W	N	P	N	0	0

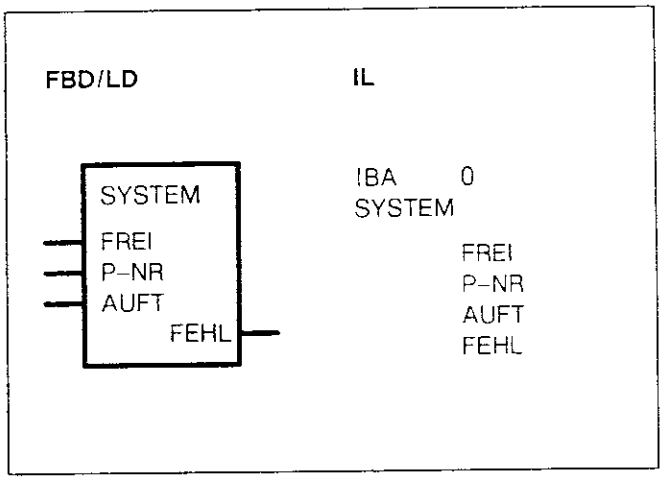
CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)
00001	USM			
00002		PP 0	ADR	Input WORD (operand address)
00003		PP 0	E	Input WORD (operand value)

This block serves to:

- Start (G)
- Abort (A)
- Stop (W)
- Continue (L)

a user program.



Parameters

FREI	BINARY	E, A, M, K, S	Block enable with 0-1 edge
P-NR	WORD	EW, AW, MW, KW	Number of the user program for which the order is intended.
AUFT	WORD	EW, AW, MW, KW	Identifier for the order to be executed.
FEHL	WORD	AW, MW	Identifier indicating whether or not the order has been executed successfully.

CE Data

Runtime:	
Basic runtime:	448 µs; 51 µs - no block enable
Additional runtime:	---
Output updating:	no
Number of historical values:	2 words
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This block serves to:

- Start (G)
- Abort (A)
- Stop (W)
- Continue (L)

a user program.

The abbreviations of these commands used in the PLC description, ABB Procontic T300, chapter entitled "Operating and test functions", where the precise functions are also explained, are given in brackets.

The block follows the ordering specification detailed in DIN 66264. After entry of the order in the control blocks in the transfer RAM, an operating system call (SCHED) is realized for the ordering task. As the result of this, the ordering task is run through once, whereby the order is executed.

Several program cycles may pass before the message appears at the output FEHL to the effect that the order has been executed. However, the order is processed in the background, i.e. processing of the user program is not affected.

The inputs and outputs can neither be duplicated nor inverted nor negated.

FREI	BINARY	Enable block processing
FREI = 0-1-edge:		Non-recurring processing of the block
FREI = no 0-1 edge:		No processing of the block

P-NR WORD

Number of the user program for which the order is intended.

P-NR = 1: User program 1
P-NR = 2: User program 2

AUFT WORD

Identifier for the order to be executed

Order	Identifier
START	1
ABORT	8
STOP	2
CONTINUE	4

FEHL WORD

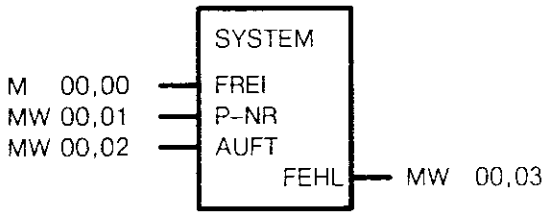
Error identifier for order handling

The output FEHL is set to -1 with a 0-1 edge at the FREI input. After handling of the order, FEHL is set to 0 or to the error identifier.

FEHL = -1: Issued order is still being processed.
FEHL = 0: Order has been executed successfully.
FEHL > 0: Error has occurred during order handling. Refer to ABB Procontic T300, Functional Description PLC chapter entitled "Self-diagnosis and reactions to errors" for an explanation of the error identifiers.

Example

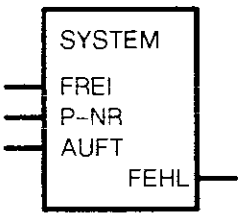
FBD/LD



IL

```
IBA      0
SYSTEM
         M    00,00
         MW   00,01
         MW   00.02
         MW   00,03
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
P-NR	E	W	N	P	Y	0	0
AUFT	E	W	N	P	Y	0	0
FEHL	A	W	N	P	Y	0	0

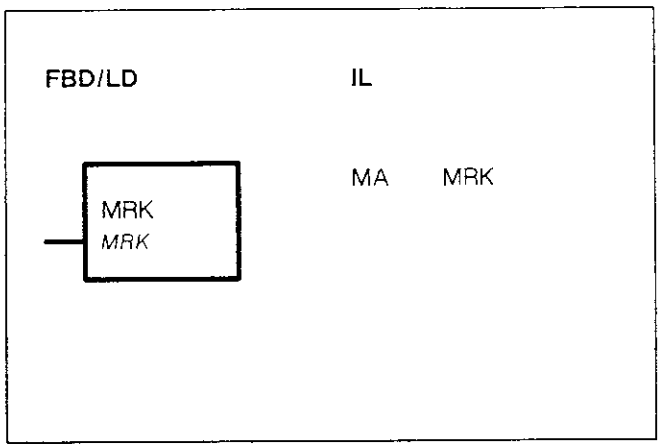
CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)
00001	SYSTEM			
00002		PP 0	FREI	Enable (BINARY)
00003		PP 0	P-NR	Program number (WORD)
00004		PP 0	AUFT	Order (WORD)
00005		PP 0	FEHL	Error (WORD)

In the PLC program, the branch target of a branch block or run number block is defined by a label.

The name and number of the label is specified at the input MRK in the following form: MRK No.

where: MRK Key word
0 ≤ No. ≤ 999 Number of the label



Parameters

MRK	SPECIAL	MRK	Target label

CE Data

Runtime:
Basic runtime: 0 µs
Additional runtime: ---
Output updating: not applicable
Number of historical values: none
Available as of: ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

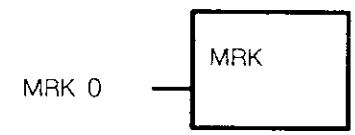
In the PLC program, the branch target of a branch block or run number block is defined by a label.

The name and number of the label is specified at the input MRK in the following form: MRK No.

Where: MRK Key word
0 ≤ No. ≤ 999 Number of the label

Example

FBD/LD



IL

```
MA      MRK 0
```

CE FBD Definition

Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
MRK	S	S	N	P	N	0	0

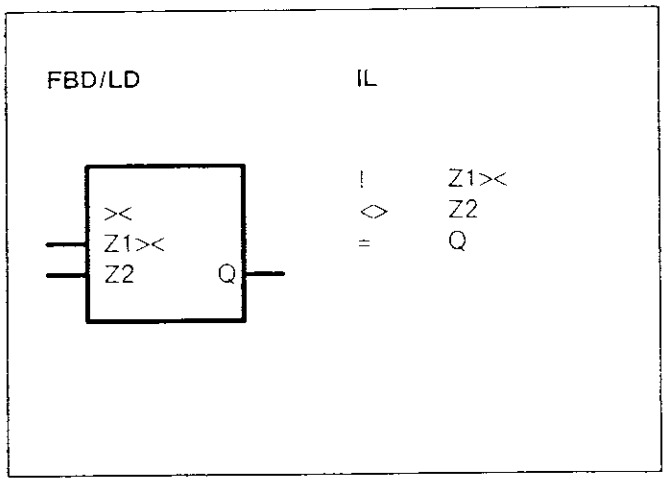
CE IL Definition

00000	MA	PP 0	MRK	Target label

UNEQUAL ><

The value of the operand at the input Z1>< is compared to the value of the operand at the input Z2.

If the value at Z1>< is greater than or less than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1>< is equal to Z2.



Parameters

Z1<>	WORD	EW, MW, AW, KW	Value to be compared
Z2	WORD	EW, MW, AW, KW	Comparison value
Q	BINARY	A, M, S	Result of the comparison

CE Data

Runtime:	
Basic runtime:	< 12 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The value of the operand at the input Z1>< is compared to the value of the operand at the input Z2.

If the value at Z1>< is greater than or less than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1>< is equal to Z2.

The inputs are capable of negation, but cannot be duplicated. The output can be inverted, but cannot be duplicated.

Note:

For reasons of compatability (graphical symbol and symbols) with ABB Procontic b, in FBD the call >< is used for "unequal" interrogation although the operand in IL is <>.

Number range

Integer word (16 bits)

The following specially applies here to the non-negated inputs:

- Low limit 8000 H -32768
- High limit 7FFF H +32767

The following generally applies:

- Low limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

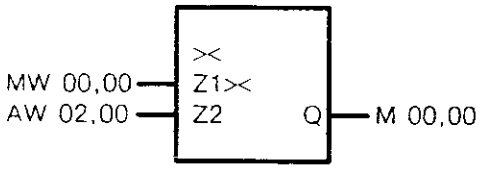
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

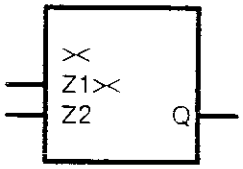
FBD/LD



IL

```
!     MW 00,00
<>    AW 02,00
=     M  00,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1><	E	W	Y	P	Y	0	0
Z2	E	W	Y	P	Y	0	0
Q	A	L	Y	P	Y	0	0

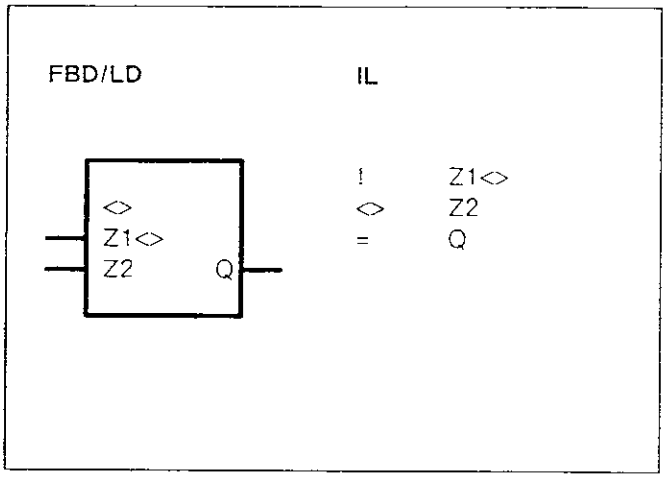
CE IL Definition

00000	I	PP 0	Z1><	Input WORD
00002	<>	PP 0	Z2	Input WORD
00004	=	PP 0	Q	Output BiNARY

UNEQUAL <>

The value of the operand at the input Z1<> is compared to the value of the operand at the input Z2.

If the value at Z1<> is greater than or less than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1<> is equal to Z2.



Parameters

Z1<>	WORD	EW, MW, AW, KW	Value to be compared
Z2	WORD	EW, MW, AW, KW	Comparison value
Q	BINARY	A, M, S	Result of the comparison

CE Data

Runtime:	
Basic runtime:	< 12 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

The value of the operand at the input Z1<> is compared to the value of the operand at the input Z2.

If the vaue at Z1<> is greater than or less than the one at Z2, the state 1 is allocated to the operand at the output Q. The state 0 is allocated to Q if Z1<> is equal to Z2.

The inputs are capable of negation, but cannot be duplicated. The output can be inverted, but cannot be duplicated.

Number range

Integer word (16 bits)

The following specially applies here to the non-negated inputs:

- Low limit 8000 H -32768
- High limit 7FFF H +32767

The following generally applies:

- Low limit: 8001 H -32767
- High limit: 7FFF H +32767
- Inadmissible value: 8000 H ---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

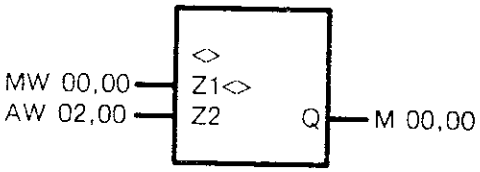
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

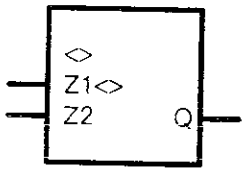
FBD/LD



IL

```
I     MW 00,00
<>    AW 02,00
=     M  00,00
```

CE FBD Definition



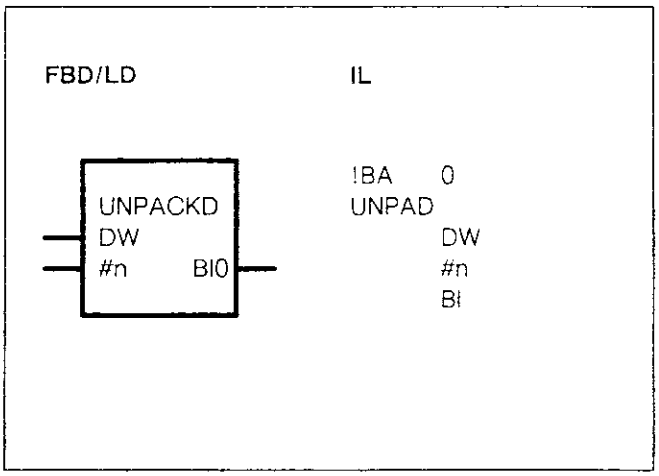
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
Z1<>	E	W	Y	P	Y	0	0
Z2	E	W	Y	P	Y	0	0
Q	A	L	Y	P	Y	0	0

CE IL Definition

00000	I	PP 0	Z1<>	Input WORD
00002	<>	PP 0	Z2	Input WORD
00004	=	PP 0	Q	Output BINARY

UNPACKING A DOUBLE WORD INTO BINARY VARIABLES UNPACKD

This function block unpacks the double word variable at the input DW. Each bit of the input variable is allocated to one binary variable each (BI0...BIn) at the output.



Parameters

DW	DOUBLE WORD	MD, KD	Double word variable to be unpacked
#n	WORD	#,#H	Number of output variables at BI0... BIn-1
BI0	BINARY	A, M	1st binary output variable; the output can be duplicated

CE Data

Runtime:	
Basic runtime:	40 µs
Additional runtime:	9 µs per output
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R101

Description

This function block unpacks the double word variable at the input DW. Each bit of the input variable is allocated to one binary variable each (BI0...BIn-1) at the output.

DW DOUBLE WORD
The variable to be unpacked is specified at the input DW. Each bit (bit 0...bit 31) of this input variable is allocated to the affiliated output variable (BI0...BIn-1).

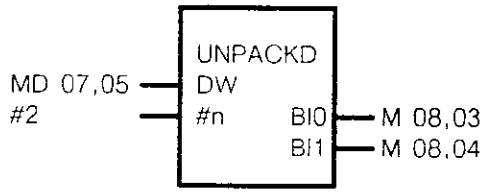
#n DIRECT CONSTANT (#,#H)
The number of planned binary outputs (BI0...BIn-1) is specified at the input #n. This is specified as a direct constant. The following applies: $1 \leq n \leq 32$

BI0 BINARY
The output BI0 is capable of duplication (BI0...BIn-1). The affiliated bits of the variable at the input DW are allocated to the binary outputs BI0...BIn.

Affiliation
Input variable Bit0 -> BI0
Input variable Bit1 -> BI1
. . . .
. . . .
Input variable Bit31 -> BI31

Example

FBD/LD



IL

```
!BA    0
UNPAD
       MD 07,05
       #2
       M   08,03
       M   08,04
```

CE FBD Definition

Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
DW	E	D	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
BI	A	L	N	P	Y	1	0

CE IL Definition

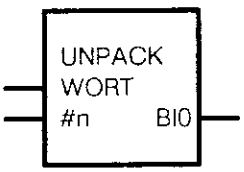
00000	!BA	0	Nr	Block No. (preset to 0)
00001	UNPAD			
00002		PP 0	DW	Input DOUBLE WORD
00003		PP 0	#n	# CONSTANT (number of BITs)
	[	1		
00004		PP 1	BI	Output BINARY
	]	1		

UNPACKING A WORD INTO BINARY VARIABLES

UNPACK

This function block unpacks the word variable at the input WORT. Each bit of the input variable is allocated to one binary variable each (BI0...BIn) at the output.

FBD/LD



IL

```
!BA     0
UNPACK
        WORT
        #n
        BI0
```

Parameters

WORT	WORD	EW, AW, MW, KW	Word variable to be unpacked
#n	DIRECT CONSTANT	#, #H	Number of output variables BI0 ... BIn-1
BI0	BINARY	A, M	1st binary output variable; the output can be duplicated

CE Data

Runtime:	
Basic runtime:	23.5 µs
Additional runtime:	9.5 µs per planned binary variable
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701 R801 / 35 ZE 93 R101

Description

This function block unpacks the word variable at the input WORT. Each bit of the input variable is allocated to one binary variable each (BI0...BIn-1) at the output.

WORT WORD

The variable to be unpacked is specified at the input WORT. Each bit (bit 0...bit 15) of this input variable is allocated to the affiliated output variable (BI0...BIn-1).

#n DIRECT CONSTANT (#,#H)

The number of planned binary outputs (BI0...BIn-1) is specified at the input #n. This is specified as a direct constant.

The following applies: $1 \leq n \leq 16$

BI0 BINARY

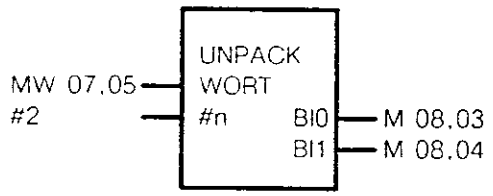
The output BI0 can be duplicated (BI0...BIn-1). The affiliated bits of the variable at the input WORT are allocated to the binary outputs BI0...BIn.

Affiliation

Input variable Bit0 -> BI0
Input variable Bit1 -> BI1
. . . .
. . . .
Input variable Bit15 -> BI15

Example

FBD/LD



IL

```
!BA     0
UNPACK
        MW 07,05
        #2
        M   08,03
        M   08,04
```

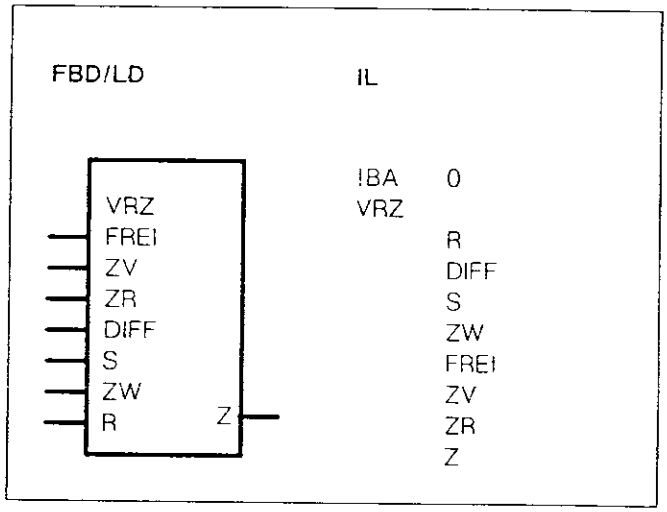
CE FBD Definition

Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
WORT	E	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
BI	A	L	N	P	Y	1	0

CE IL Definition

```
00000   !BA     0        Nr      Block No. (preset to 0)
00001   UNPACK
00002           PP  0    WORT    Input WORD
00003           PP  0    #n      # CONSTANT (number of BITs)
        [       1
00004           PP  1    BI      Output BINARY
        ]       1
```

This function block serves to count pulses. During counting, the positive edge of the pulse is evaluated in each case. The counter is capable of counting both up and down and the counting increment can be specified. It is possible to preset the counter content to an intermediate value.



Parameters

FREI	BINARY	E, M, A, K, S	Enable block processing
ZV	BINARY	E, M, A, K, S	Pulse input, up counting
ZR	BINARY	E, M, A, K, S	Pulse input, down counting
DIFF	WORD	EW, MW, AW, KW	Counter content change per positive edge (increment)
S	BINARY	E, M, A, K, S	Set counter to an intermediate value
ZW	WORD	EW, MW, AW, KW	Intermediate value
R	BINARY	E, M, A, K, S	Reset counter
Z	WORD	AW, MW	Output for counter content

CE Data

Runtime:	
Basic runtime:	92 µs
Additional runtime:	---
Output updating:	yes if FREI = 1
Number of historical values:	2 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block serves to count pulses. During counting, the positive edge of the pulse is evaluated in each case. The counter is capable of counting both up and down and the counting increment can be specified. It is possible to preset the counter content to an intermediate value.

FREI BINARY

Counting is enabled or disabled by means of the FREI input.
The following applies: FREI = 0 -> Counting disabled
FREI = 1 -> Counting enabled

ZV BINARY

Each positive edge (0->1 edge) at the input ZV increases the current counter content by the increment specified at the DIFF input.

ZR BINARY

Each positive edge (0->1 edge) at the input ZR decreases the current counter content by the increment specified at the DIFF input.

DIFF WORD

The increment for the counting operation is specified at the DIFF input. The increment is the value by which the counter is changed at the input ZV or ZR with each positive edge.

S BINARY

By means of a 1 signal at the input S, the counter content is set to the value specified at the input ZW. Counting is blocked as long as 1 signal is present at the input S. Setting is also effective when a 1 signal is present at the FREI input.

ZW WORD

The value to which the counter content is set by a 1 signal at the input S is specified at the input ZW.

R BINARY

A 1 signal at the input R sets the counter content to the value 0. The reset input R has the highest priority of all inputs.

Z WORD

The current counter content is available at the output Z.

If the counter reaches the positive or negative limit of the number range, the counter is limited to this value.

The inputs and the output can neither be duplicated nor negated/inverted.

Number range

Integer word (16 bits)

• Low limit:	8001 H	-32767
• High limit:	7FFF H	+32767
• Inadmissible value:	8000 H	---

In the two's complement arithmetic, the value 8000H (-32768) lies *outside of* the number range and is neither generated nor processed correctly by the PLC. If this *forbidden* value reaches the PLC

- by bit manipulations of the user or
- by being read from outside the PLC or
- by an indirect word constant

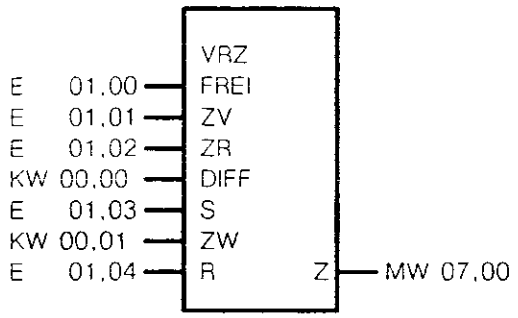
under no circumstances may negation or subtraction be carried out on this value.

An admissible value is generated again by means of an allocation (=), addition (+), multiplication (*) or division (:).

On allocation (=), the *forbidden* value 8000H (-32768) is corrected to the allowed value 8001H (-32767).

Example

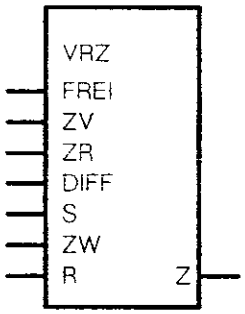
FBD/LD



IL

```
!BA   0
VRZ
      E   01.04
      KW  00.00
      E   01.03
      KW  00.01
      E   01.00
      E   01.01
      E   01.02
      MW  07.00
```

CE FBD Definition

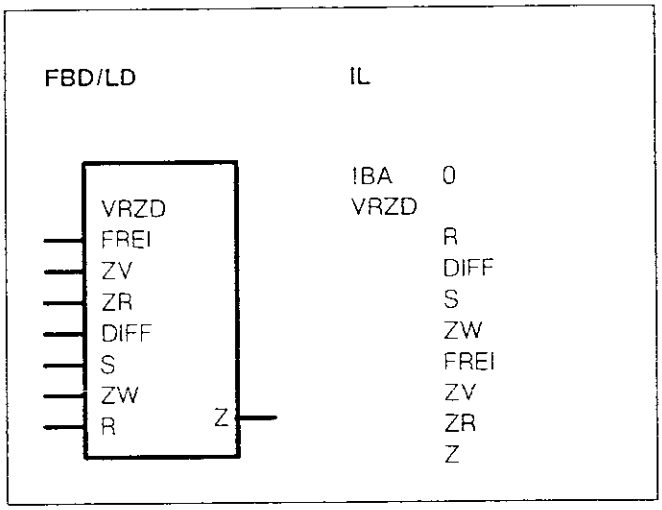


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
ZV	E	L	N	P	Y	0	0
ZR	E	L	N	P	Y	0	0
DIFF	E	W	N	P	Y	0	0
S	E	L	N	P	Y	0	0
ZW	E	W	N	P	Y	0	0
R	E	L	N	P	Y	0	0
Z	A	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	VRZ			
00002		PP 0	R	Input BINARY (Reset counter = 0)
00003		PP 0	DIFF	Input WORT (change per edge)
00004		PP 0	S	Input BINARY (set counter = ZW)
00005		PP 0	ZW	Input WORT (initial value)
00006		PP 0	FREI	Input BINARY (block enable)
00007		PP 0	ZV	Input BINARY (counter input, upward)
00008		PP 0	ZR	Input BINARY (counter input, downward)
00009		PP 0	Z	Output WORD (counter content)

This function block serves to count pulses. During counting, the positive edge of the pulse is evaluated in each case. The counter is capable of counting both up and down and the counting increment can be specified. It is possible to preset the counter to an intermediate value.



Parameters

FREI	BINARY	E, M, A, K, S	Enable block processing
ZV	BINARY	E, M, A, K, S	Pulse input, up counting
ZR	BINARY	E, M, A, K, S	Pulse input, down counting
DIFF	DOUBLE WORD	MD, KD	Counter content change per positive edge (increment)
S	BINARY	E, M, A, K, S	Set counter to an intermediate value
ZW	DOUBLE WORD	MD, KD	intermediate value
R	BINARY	E, M, A, K, S	Reset counter
Z	DOUBLE WORD	MD	Output for counter content

CE Data

Runtime:	
Basic runtime:	38 ... 113 µs (according to the mode)
Additional runtime:	---
Output updating:	yes if FREI = 1
Number of historical values:	3 words
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block serves to count pulses. During counting, the positive edge of the pulse is evaluated in each case. The counter is capable of counting both up and down and the counting increment can be specified. It is possible to preset the counter to an intermediate value.

FREI BINARY

Counting is enabled or disabled by means of the FREI input.

The following applies: FREI = 0 -> Counting disabled
FREI = 1 -> Counting enabled

ZV BINARY

Each positive edge (0->1 edge) at the input ZV increases the current counter content by the increment specified at the DIFF input.

ZR BINARY

Each positive edge (0->1 edge) at the input ZR decreases the current counter content by the increment specified at the DIFF input.

DIFF DOUBLE WORD

The increment for the counting operation is specified at the DIFF input. The increment is the value by which the counter is changed at the input ZV or ZR with each positive edge.

S BINARY

By means of a 1 signal at the input S, the counter content is set to the value specified at the input ZW. Counting is blocked as long as 1 signal is present at the input S. Setting is also effective when a 1 signal is present at the FREI input.

ZW DOUBLE WORD

The value to which the counter content is set by a 1 signal at the input S is specified at the input ZW.

R BINARY

A 1 signal at the input R sets the counter content to the value 0. The reset input R has the highest priority of all inputs.

Z DOUBLE WORD

The current counter content is available at the output Z.

If the counter reaches the positive or negative limit of the number range, the counter is limited to this value.

The inputs and the output can neither be duplicated nor negated/inverted.

Number range

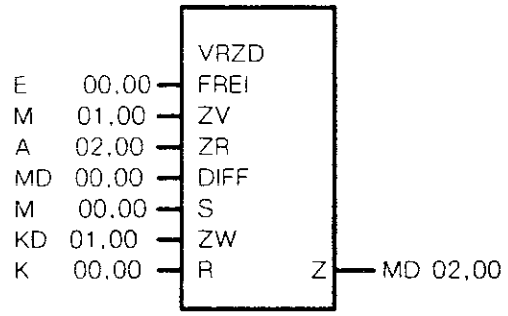
Integer double word (32 bits)

• Low limit:	8000 0001 H	-2 147 483 647
• High limit:	7FFF FFFF H	+2 147 483 647
• Inadmissible value:	8000 0000 H	---

All blocks for double word arithmetic subject values to be processed for admissibility. If the inadmissible value occurs, it is corrected to the admissible value 8000 0001 H (-2 147 483 647)

Example

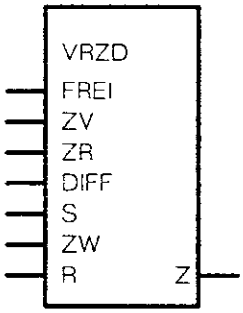
FBD/LD



IL

```
!BA        0
VRZD
      K    00.00
      MD   00.00
      M    00.00
      KD   01.00
      E    00.00
      M    01.00
      A    02.00
      MD   02.00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
ZV	E	L	N	P	Y	0	0
ZR	E	L	N	P	Y	0	0
DIFF	E	D	N	P	Y	0	0
S	E	L	N	P	Y	0	0
ZW	E	D	N	P	Y	0	0
R	E	L	N	P	Y	0	0
Z	A	D	N	P	Y	0	0

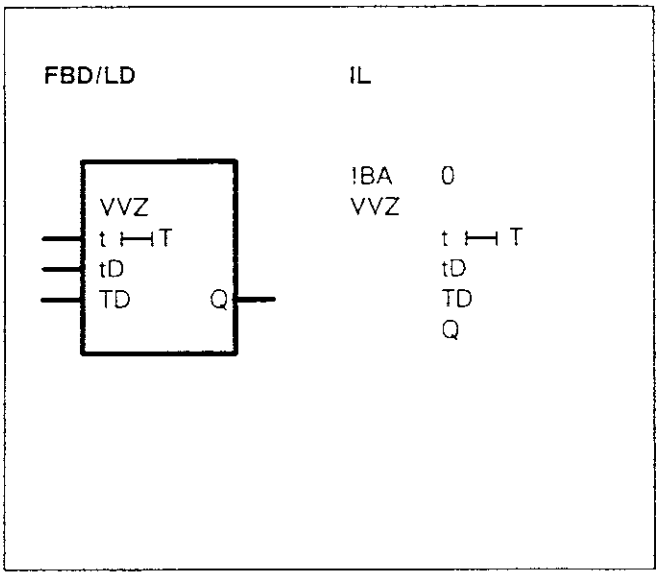
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	VRZD			
00002		PP 0	R	Input BINARY (Reset counter = 0)
00003		PP 0	DIFF	Input DOUBLE WORD (change/pulse)
00004		PP 0	S	Input BINARY (set counter = ZW)
00005		PP 0	ZW	Input DOUBLE WORD (initial value)
00006		PP 0	FREI	Input BINARY (block enable)
00007		PP 0	ZV	Input BINARY (counter input, upward)
00008		PP 0	ZR	Input BINARY (counter input, downward)
00009		PP 0	Z	Output DOUBLE WORD (counter content)

A 0-1 edge at the input t ⊢⊣ T is delayed by the time tD and output through the output Q. A subsequent 1-0 edge at the input t ⊢⊣ T is delayed by the time TD and output through the output Q.

If the input t ⊢⊣ T changes again to 0 level before the time tD has elapsed, the output Q retains 0 level.

Maximum time offset at the output: < 1 cycle time



Parameters

t ⊢⊣ T	BINARY	E, M, A, S, K	Input signal
tD	DOUBLE WORD	MD, KD	Delay time 0-1 edge
TD	DOUBLE WORD	MD, KD	Delay time 1-0 edge
Q	BINARY	M, A	Output signal

CE Data

Runtime:	
Basic runtime:	46 µs, 0-1 edge 566 µs, 1-0 edge 283 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	2 words
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

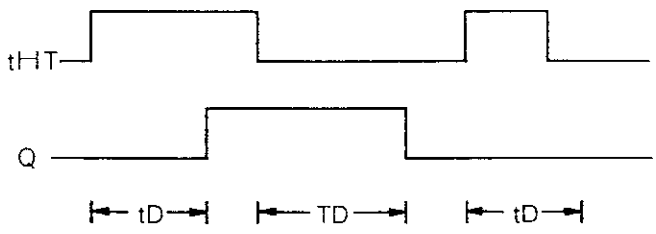
A 0-1 edge at the input t ⊢⊣ T is delayed by the time tD and output through the output Q. A subsequent 1-0 edge at the input t ⊢⊣ T is delayed by the time TD and output through the output Q.

If the input t ⊢⊣ T changes again to 0 level before the time tD has elapsed, the output Q retains 0 level.

Maximum time offset at the output: < 1 cycle time

Meaningful range for tD and TD: > 1 cycle time

The inputs and the output can neither be duplicated nor inverted.



General response

- Started timers are processed by the PLC's operating system and are therefore completely independent of processing of the PLC program. An appropriate message of the operating system is not issued to the affiliated timer block in the PLC program until the timer has elapsed.
- Processing of a timer in the PLC's operating system is *not* influenced by the following commands:
  - Abort program
  - Start program
  - Stop program
  - Continue program

  That is to say, processing of a started timer is continued in the PLC's operating system even if the affiliated PLC program is aborted, restarted or stopped and continued again.

Initialization

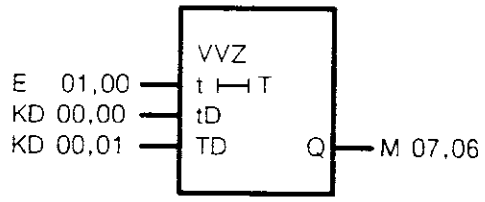
The timers are always initialized each time a cold or warm start of the PLC is executed. A running timer will always be aborted by a cold or warm start of the PLC.

Cold start:
- KALT <CR> command
- Activating the voltage for the first time

Warm start:
- WARM <CR> command
- Activating the voltage
- RESET switch

Example

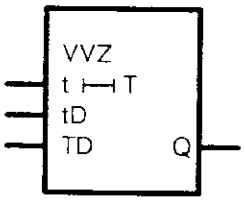
FBD/LD



IL

```
IBA   0
VVZ
      E   01,00
      KD  00,00
      KD  00,01
      M   07,06
```

CE FBD Definition

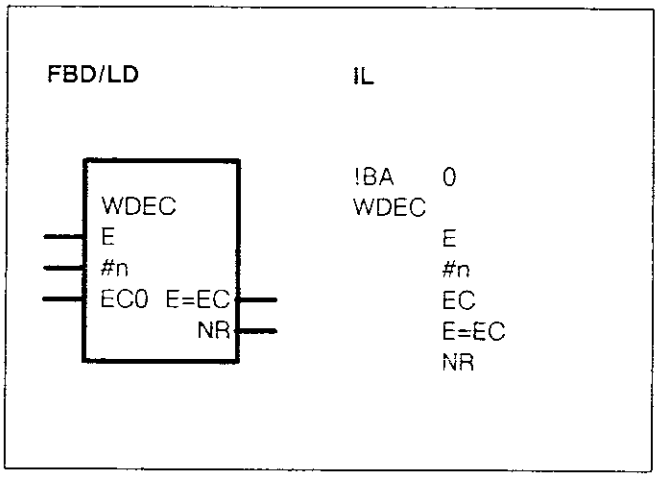


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
t ⊢⊣ T	E	L	N	P	Y	0	0
tD	E	D	N	P	Y	0	0
TD	E	D	N	P	Y	0	0
Q	A	L	N	P	Y	0	0

CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)
00001	VVZ			
00002		PP 0	t ⊢⊣ T	Input BINARY (start time)
00003		PP 0	tD	Constant DOUBLE WORD (time value)
00004		PP 0	TD	Constant DOUBLE WORD (time value)
00005		PP 0	Q	Output BINARY (time)

This function block compares the value of the operand at the input E to the reference values of the operands at the inputs EC0...ECn-1. The result of the comparison is signalled at the outputs.



Parameters

E	WORD	EW, AW, MW, KW	Input
#n	DIRECT CONSTANT	#, #H	Number of reference values
EC0	WORD	EW, AW, MW, KW	Reference value; input can be duplicated
E=EC	BINARY	A, M	Coincidence indication
NR	WORD	AW, MW	Number of the reference value in the event of coincidence

CE Data

Runtime:	
Basic runtime:	22 µs
Additional runtime:	4 µs (max.) per entered reference value
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block compares the value of the operand at the input E to the reference values of the operands at the inputs EC0...ECn-1. The result of the comparison is signalled at the outputs.

If the input E agrees with at least 1 of the n reference values EC, the output E=EC is set to 1. The number of the 1st reference value EC agreeing with the input E is allocated to the operand at the output NR. Therefore, the number may assume a value from 1 to n.

The outputs E=EC and NR are set to 0 if no agreement between the input value E and the reference value EC is determined.

The input EC (input code) is capable of duplication. The number of inputs EC must be specified as a direct constant at the input #n.

E WORD

The operand whose value is to be compared to the reference values at the inputs EC0...ECn-1 is specified at the input E.

#n DIRECT CONSTANT (#, #H)

The number n of planned reference values is specified at the input #n. This is specified as a direct constant.

EC0 ... ECn-1 WORD

The input EC0 must be duplicated according to the required number of reference values. The operands for the reference values are specified at the inputs EC0...ECn-1. The value of the operand at the input E is compared to these reference values.

E=EC BINARY

Coincidence of the operand value at the input E with one of the reference values is signalled at the output E=EC.

The following applies: E=EC = 0 -> No coincidence
E=EC = 1 -> Coincidence

NR WORD

The number of the reference value that agrees with the value of the operand at the input E is signalled at the output NR.

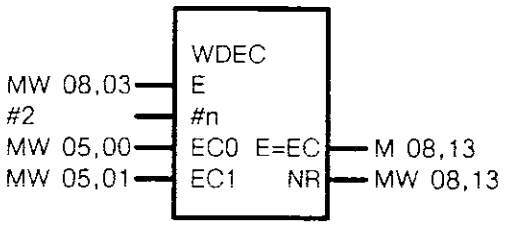
The following applies:

No coincidence -> NR = 0
Coincidence-> NR = n where $n \geq 1$

E = EC0 -> NR = 1
E = EC1 -> NR = 2
. . .
. . .
E = ECn-1 -> NR = n

Example

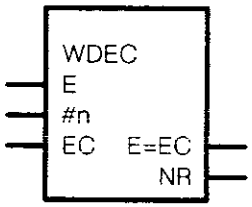
FBD/LD



IL

```
!BA    0
WDEC
       MW 08,03
       #2
       MW 05,00
       MW 05,01
       M  08,13
       MW 08,13
```

CE FBD Definition

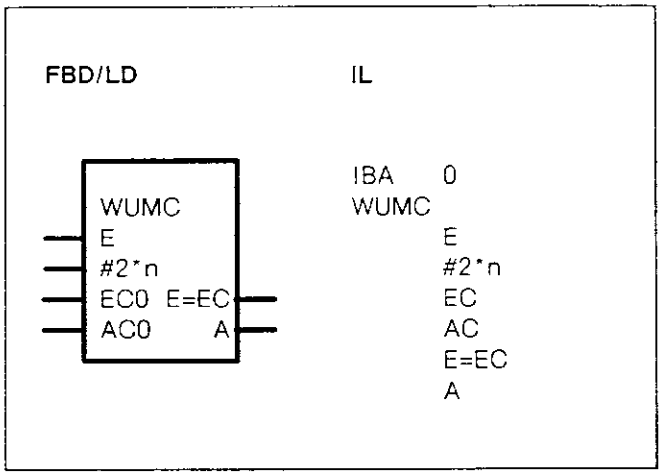


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	W	N	P	Y	0	0
#n	K	W	N	P	Y	0	0
EC	E	W	N	P	Y	1	0
E=EC	A	L	N	P	Y	0	0
NR	A	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WDEC			
00002		PP 0	E	input WORD (input value)
00003		PP 0	#n	# CONSTANT (number of ECs)
	[	1		
00004		PP 1	EC	Input WORD (input code)
	]	1		
00005		PP 0	E=EC	Output BINARY (input value=Ecode)
00006		PP 0	NR	Output WORD (number)

This function block compares the value of the operand at the input E to the reference values of the operands at the inputs EC0...ECn-1. If the input E agrees with at least one of the reference values EC, the output E=EC is set to 1. The output A receives the value of the output code AC, which is allocated to the reference value EC found.



Parameters

E	WORD	EW, AW, MW, KW	Input
#2*n	DIRECT CONSTANT	#, #H	Quantity n of reference values (multiplied by 2)
EC0	WORD	EW, AW, MW, KW	Reference value; input can be duplicated
AC0	WORD	EW, AW, MW, KW	Output code; input can be duplicated
E=EC	BINARY	A, M	Coincidence indication
A	WORD	AW, MW	Output of the output code's value

CE Data

Runtime:	
Basic runtime:	40.0 µs
Additional runtime:	7.5 µs per entered comparison value
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

This function block compares the value of the operand at the input E to the reference values of the operands at the inputs EC0...ECn-1. If the input E agrees with at least one of the reference values EC, the output E=EC is set to 1. The output A receives the value of the output code AC, which is allocated to the reference value EC found.

Each reference value at the inputs ECi is assigned an operand for the output code ACi. The affiliation of EC to AC is recognizable by the index i. The index begins with 0 and is generated automatically in the event of duplication.

The number of inputs EC and AC must be specified as a direct constant at the input #2*n.

The inputs and outputs cannot be negated/inverted.

E WORD

The operand whose value is to be compared to the values of the n reference values (EC0...ECn-1) is specified at the input E.

#2*n DIRECT CONSTANT (#, #H)

The total number (2*n) of the reference values (EC0...ECn-1) and output codes (AC0...ACn-1) is specified at the input #2*n. This is specified as an indirect constant.

EC0 ... ECn-1 WORD

The input EC0 must be duplicated according to the required number of reference values. The operands for the reference values are specified at the inputs EC0...ECn-1. The value of the operand at the input E1 is compared to the reference values.

AC0 ... ACn-1 WORD

By duplication of the input EC0, the input AC0 is automatically also duplicated.

The output codes are specified at the inputs AC0...ACn-1. The output code ACi is output through the output A if the input E agrees with one of the reference values ECi.

Affiliations between the reference values and output codes:

EC0 <-> AC0
EC1 <-> AC1
. .
. .
EC_{n-1} <-> ACn-1

E=EC BINARY

Agreement between the operand value of the input E and one of the reference values is signalled at the output E=EC.

The following applies: E=EC = 0 -> No coincidence
E=EC = 1 -> Coincidence

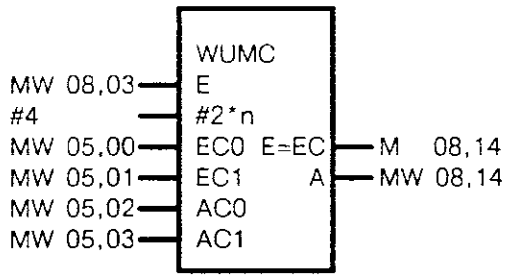
A WORD

The output code ACi is output through the output A if the input E agrees with one of the reference values ECi.

The following applies: A = 0 -> No coincidence
A = ACi -> Coincidence

Example

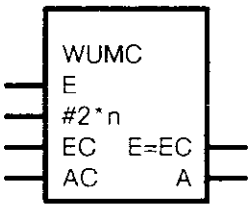
FBD/LD



IL

```
!BA    0
WUMC
       MW 08,03
       #4
       MW 05,00
       MW 05,01
       MW 05,02
       MW 05,03
       M  08,14
       MW 08,14
```

CE FBD Definition

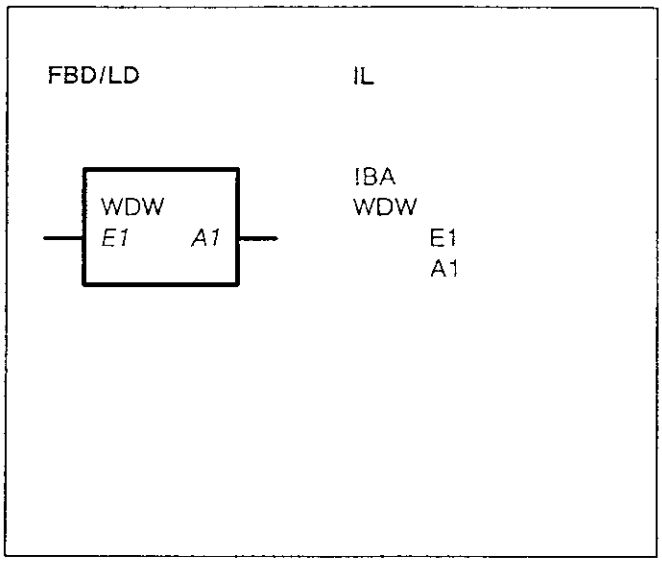


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E	E	W	N	P	Y	0	0
#2*n	K	W	N	P	Y	0	0
EC	E	W	N	P	Y	1	0
AC	E	W	N	P	Y	1	0
E=EC	A	L	N	P	Y	0	0
A	A	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WUMC			
00002		PP 0	E	Input WORD (input value)
00003		PP 0	#2*n	# CONSTANT (number of EC and AC)
	[	1		
00004		PP 1	EC	Input WORD (reference value)
	]	1		
	[	1		
00005		PP 1	AC	Input WORD (output code)
	]	1		
00006		PP 0	E=EC	Output BINARY (input value=Ecode)
00007		PP 0	A	Output WORD (output value)

The value of the word operand at the input E1 is converted to a double word quantity and the result is allocated to the double word operand at the output A1.



Parameters

E1	WORD	EW, AW, MW, KW	Word quantity to be converted
A1	DOUBLE WORD	MD	Result of conversion, double word quantity

CE Data

Runtime:
- Basic runtime: 26 µs
- Additional runtime: ---

Output updating: yes
Number of historical values: none
Available as of: ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R101

Description

The value of the word operand at the input E1 is converted to a double word quantity and the result is allocated to the double word operand at the output A1.

Value range for E1:
$8000H \leq E1 \leq 7FFFH$
$-32768 \leq E1 \leq 32768$

The input and the output can neither be duplicated nor negated.

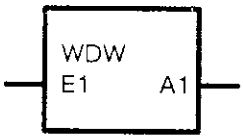
Example

FBD/LD

IL

```
!BA   0
WDW
      EW 00,00
      MD 02,00
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
A1	A	D	N	P	N	0	0

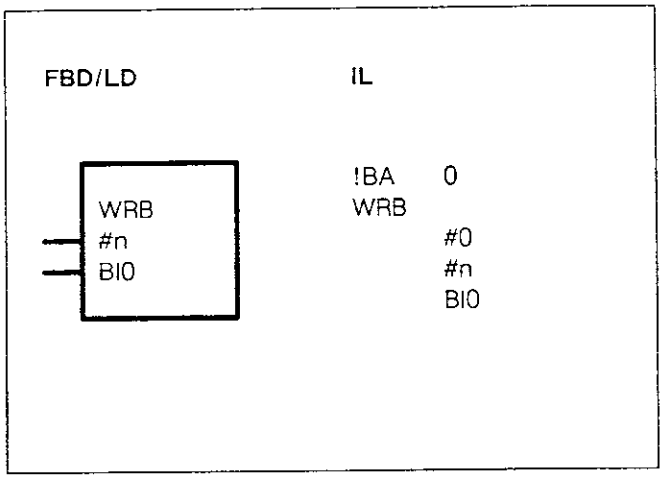
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WDW			
00002		PP 0	E1	Input WORD
00003		PP 0	A1	Output DOUBLE WORD

WRITE BINARY VALUES INTO HISTORICAL VALUES MEMORY

WRB

This function block stores the values of the operands at the input BI0...BIn-1 in the historical values memory. The affiliated function block RDB reads these values out of the historical values memory again. The function blocks WRB and RDB always occur in pairs.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of inputs BI0 ... BIn-1
BI0	BINARY	A, M	Input for the binary values to be written, capable of duplication (BI0 ... BIn-1)

CE Data

Runtime:	
Basic runtime:	see RDB block
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block stores the values of the operands at the input BI0...BIn-1 in the historical values memory. The affiliated function block RDB reads these values out of the historical values memory again. The function blocks WRB and RDB always occur in pairs.

To be able to use ready-made program parts several times in one user program (e. g. 907 PC 31/32 connection elements), it may be necessary to work with local variables within this part of the program. These local variables lose their validity outside of this program part. At the end of the program part, the values of these local variables are stored in the historical values memory of the affiliated RDB block by the WRB block and are allocated to the local variables at the start of the program part by the RDB block. For this purpose, the WRB block stores the values of the binary inputs in the historical values memory of the affiliated RDB block. The number of binary inputs is specified as a direct constant at the input #n. This quantity must agree with the quantity of outputs belonging to the affiliated RDB block. At the intended input, the editing program "PA" enters the pointer to the historical values memory of the affiliated RDB block in the instruction list of the WRB block.

#0 DIRECT CONSTANT

This input does *not* exist in FBD/LD. In the instruction list, the value 0 is specified as a direct constant at this point. The PLC then enters the pointer to the historical values of the affiliated RDB block at this point.

#n DIRECT CONSTANT

The number of inputs BI0...BIn–1 is specified at the input #n. This is specified as a direct constant.

Note:

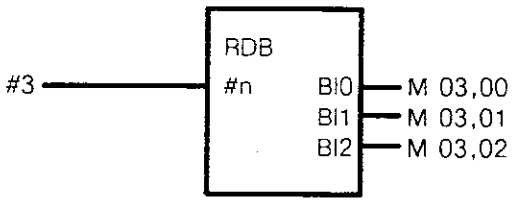
The value specified at the input #n must also agree with the number of outputs belonging to the affiliated RDB block.

BI0...BIn–1 BINARY

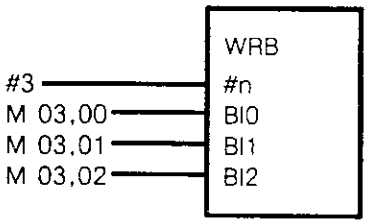
The input BI0 is capable of duplication (BI0...BIn–1). The values of the operands specified at the inputs BI0...BIn–1 are written into the historical values memory of the affiliated RDB block.

Example

FBD/LD



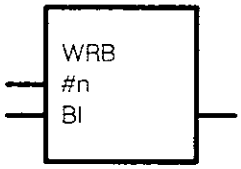
The program part, which is used multiply and in which the variables M 03.00 ... M 03.02 are needed in the next cycle, is located here. For this purpose, the values are written into the historical values memory by means of the WRB block and are read out again in the next cycle by means of the RDB block.



IL

```
!BA     0
RDB
        #3
        M 03,00
        M 03,01
        M 03,02

!BA     0
WRB
        # 0
        # 3
        M 03,00
        M 03,01
        M 03,02
```

CE FBD Definition

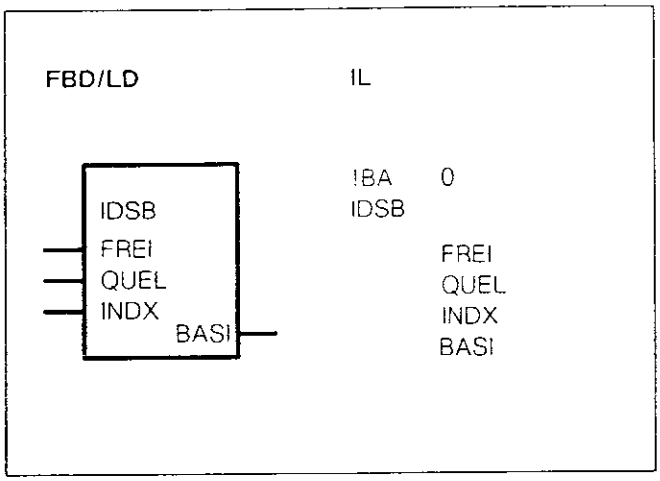


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
BI	E	L	N	P	Y	1	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WRB			
00002		# 0		
00003		PP 0	#n	DIRECT CONSTANT (number of bits)
	[	1		
00004		PP 1	BI	Input BINARY
	]	1		

This function block serves the purpose of indexed writing of binary variables.



Parameters

FREI	BINARY	E, A, M, K, S	Enable block FREI = 0: Block is not processed FREI = 1: The value of the source variable is read and allocated to the target variable
QUEL	BINARY	E, A, M, K	Source variable
INDX	WORD	EW, AW, MW, KW	The current target variable results from the index and the basic variable
BASI	BINARY	A, M	Basic variable

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T300 V8.5

Description

This function block serves the purpose of indexed writing of binary variables.

When the block is enabled, the value of the source variable is read and allocated to the target variable. The target variable is defined by indexing the basic variable.

The inputs and outputs can neither be duplicated nor negated nor inverted.

Note
EPROM programming in case of this block is not possible via the programming system. EPROM programming must be carried out in terminal emulation as operating function using command PU. At the same time the EPROM programming unit must be connected to serial interface 1.

The group and channel number of the target flag (target variable) are determined on the basis of the basic flag and the index INDX.

The target flag is called: M (G_Basis+A).(K_Basis+B) where:

G_Basis: Group number of the basic flag
K_Basis: Channel number of the basic flag

Formula:

INDX
---- = A Remainder B
16

Group No. of the target flag:
Group No. of the basic flag + A

Channel No. of the target flag:
Channel No. of the basic flag + B

Example:

Basic variable: M 00,00
-> INDX = 10 -> 10 : 16 = A = 0, Remainder B = 10
-> Target variable:
M(00+A), (00+B) = M (00+0),(00+10) =
M 00,10
-> Target variable: M 00,10

Further examples:

Basic variable		INDX	Target variable
M	00,00	0	M 00,00
M	00,00	2	M 00,02
M	00,00	16	M 01,00
M	00,02	18	M 01,04
A	00,00	3	A 00,03
A	00,15	1	A 01,00
A	05,05	6	A 05,11
A	05,05	12	K 06,01

FREI BINARY
Enable block

FREI=0: -> Block is not processed
FREI=1: -> The value of the source variable is read and allocated to the target variable ZIEL.

QUEL BINARY
The source variable is specified at the input QUEL. The value of this variable is read and allocated to the target variable.

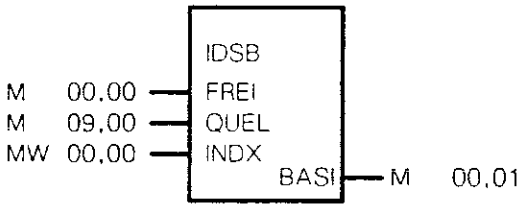
INDX WORD
The index value is specified at the input INDX. The target variable (see above for a calculation) results from the index INDX and the basic variable.

Value range: $-16383 \leq INDX \leq +16383$

BASI BINARY
The basic variable is specified at the input BASI. The target variable (see above for a calculation) results from the index INDX and the basic variable.

Example

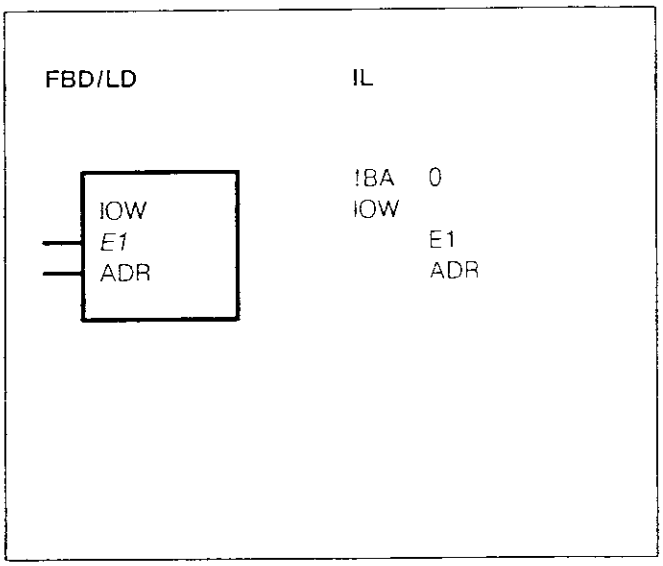
FBD/LD



IL

```
!BA   0
IDSB
      M   00.00
      M   09.00
      MW  00.00
      M   00.01
```

This function block writes the byte specified at the input E1 into the I/O area.



Parameters

E1	WORD	EW, AW, MW, KW	Operand whose LOW byte is read and written into the I/O area
ADR	WORD	EW, AW, MW, KW	I/O address to which the value of the operand is written.

CE Data

Runtime:	
Basic runtime:	24 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

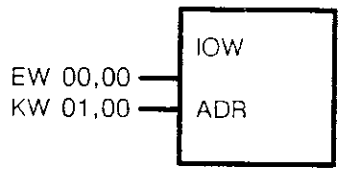
This function block writes the byte specified at the input E1 into the I/O area.

E1 WORD
The LOW BYTE of the operand at the input E1 is written into the I/O area.

ADR WORD
The value of the operand at the input ADR represents the I/O address to which the value is written.

Example

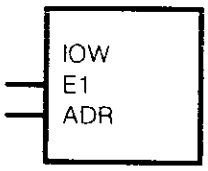
FBD/LD



IL

```
!BA    0
IOR
       EW 00,00
       KW 01,00
```

CE FBD Definition



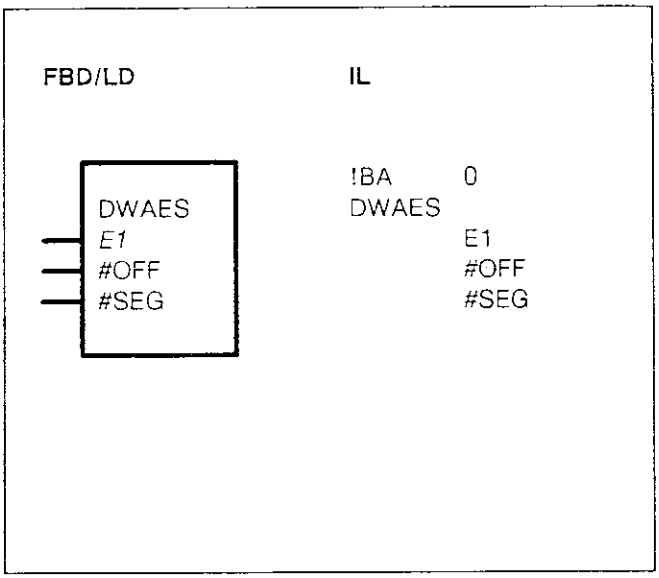
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
ADR	E	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	IOW			
00002		PP 0	E1	Input WORD (value)
00003		PP 0	ADR	Input WORD (I/O address)

WRITE DOUBLE WORD IN THE EVENT OF VALUE CHANGE — DWAES

If the value of the operand at the input E1 changes in comparison with the value during previous processing of the block, the value of the operand at the input E1 is written to the specified physical address.



Parameters

E1	DOUBLE WORD	MD, KD	Input for the operand to be read
#OFF	DIRECT CONSTANT	#, #H	Offset address of the memory location to which the value of E1 must be written in the event of a change.
#SEG	DIRECT CONSTANT	#, #H	Segment address of the memory location to which the value of E1 must be written in the event of a change.

CE Data

Runtime:	
Basic runtime:	27 µs
Additional runtime:	---
Output updating:	yes
Number of historical values:	2 words
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

If the value of the operand at the input E1 changes in comparison with the value during previous processing of the block, the value of the operand at the input E1 is written to the specified physical address.

The physical address consists of a segement and an offset.

The inputs can neither be duplicated nor negated.

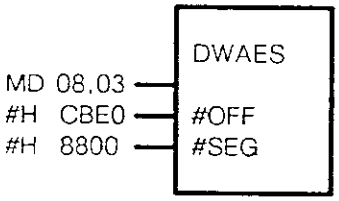
E1 DOUBLE WORD
If the operand at the input E1 changes, its value is written to the address specified at the inputs #OFF and #SEG.

#OFF DIRECT CONSTANT (#,#H)
The offset component of the address to be written is specified at the #OFF input. This is specified as a direct constant.

#SEG DIRECT CONSTANT (#,#H)
The segment component of the address to be written is specified at the input #SEG. This is specified as a direct constant.

Example

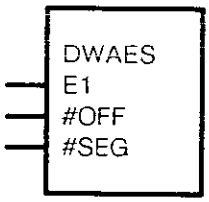
FBD/LD



IL

```
!BA     0
DWAES
        MD 08.03
        #H  CBE0
        #H  8800
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	D	N	P	N	0	0
#OFF	K	W	N	P	Y	0	0
#SEG	K	W	N	P	Y	0	0

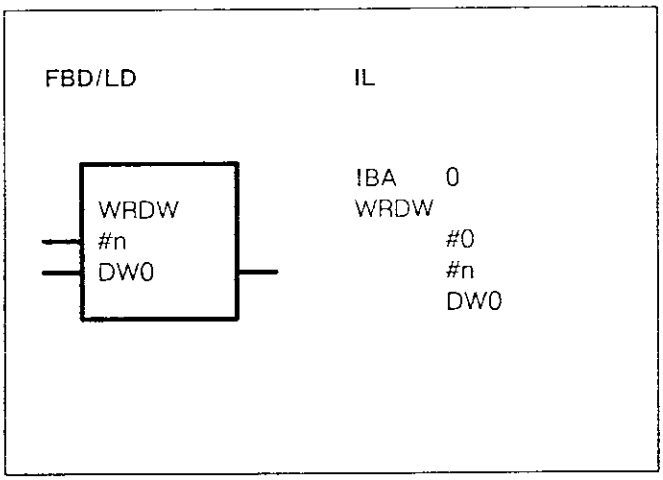
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	DWAES			
00002		PP 0	E1	Input DOUBLE WORD (Value)
00003		PP 0	#OFF	# CONSTANT (offset address)
00004		PP 0	#SEG	# CONSTANT (segment address)

WRITE DOUBLE WORD VALUES INTO HISTORICAL VALUES MEMORY

WRDW

This function block stores the values of the operands at the inputs DW0...DWn-1 in the historical values memory. The affiliated function block RDDW reads these values out of the historical values memory again. The WRDW and RDDW function blocks always occur in pairs.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of inputs DW0 ... DWn-1
DW0	DOUBLE WORD	MD	Input for the double word values to be written capable of duplication (DW0 ... DWn-1)

CE Data

Runtime:	
Basic runtime:	see RDDW block
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block stores the values of the operands at the inputs DW0...DWn-1 in the historical values memory. The affiliated function block RDDW reads these values out of the historical values memory again. The WRDW and RDDW function blocks always occur in pairs.

To be able to use ready-made program parts multiply in a user program (e.g. 907 PC 31/32 connection elements), it may be necessary to work with local variables within this part of the program. These local variables lose their validity outside of this program part. At the end of the program part, the WRDW block stores the values of these local variables in the historical values memory of the affiliated RDDW block and the RDDW block allocates them to the local variables at the start of the program part. To do this, the WRDW block stores the values of the double word inputs in the historical values memory of the affiliated RDDW block. The number of double word inputs is specified as a direct constant at the input #n. This quantity must agree with the number of outputs belonging to the affiliated RDDW block. The editing program "PA" enters the pointer to the historical values memory of the affiliated RDDW block at the input provided for this purpose in the instruction list of the WRDW block.

#0 DIRECT CONSTANT

This input does *not* exist in FBD/LD. In the instruction list, the value 0 is specified here as a direct constant. At this point, the PLC then enters the pointer to the historical values of the affiliated RDDW block.

#n DIRECT CONSTANT

The number of inputs DW0...DWn–1 is specified at the input #n. This is specified as a direct constant.

Note:

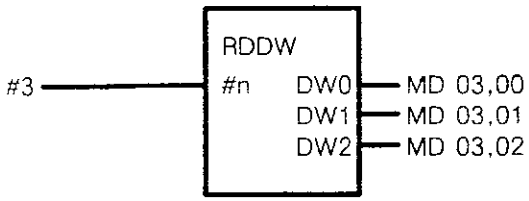
The value specified at the input #n must also agree with the number of outputs of the affiliated RDDW block.

DW0...DWn–1 DOUBLE WORD

The input DW0 is capable of duplication (DW0...DWn–1). The values of the operands specified at the inputs DW0...DWn–1 are written into the historical values memory of the affiliated RDDW block.

Example

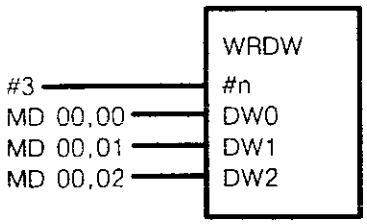
FBD/LD



IL

```
!BA    0
RDDW
       #   3
       MD 03,00
       MD 03,01
       MD 03,02
```

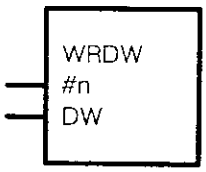
The program part, which is used multiply and in which the varlables MD 03.00 ... MD 03.02 are needed in the next cycle again, is located here. For this purpose, the WRDW block writes the values into the historical values memory and the RDDW block reads them out again in the next cycle.



IL

```
!BA    0
WRDW
       #   0
       #   3
       MD 00,00
       MD 00,01
       MD 00,02
```

CE FBD Definition

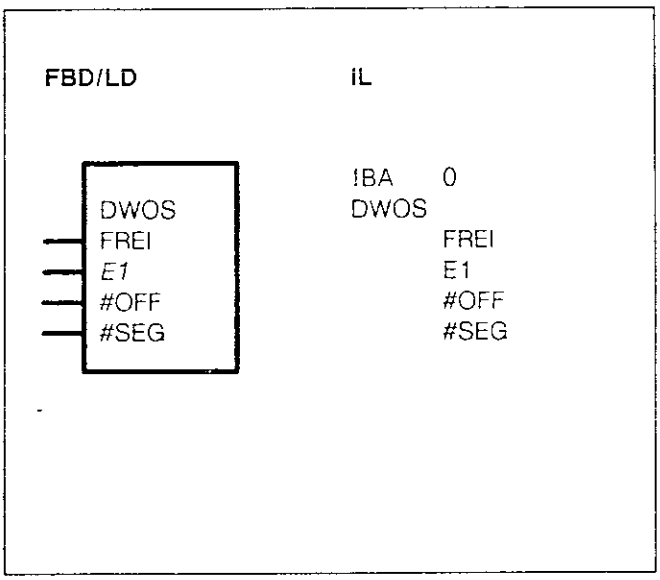


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
DW	E	D	N	P	Y	1	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WRDW			
00002		#0		
00003		PP 0	#n	# CONSTANT (number of double words)
	[	1		
00004		PP 1	DW	Input DOUBLE WORD
	]	1		

If there is a 1 signal at the input FREI, the value of the operand at the input E1 is read and then written to the specified physical address.



Parameters

FREI	BINARY	E, M, A, S, K	Block enable
E1	DOUBLE WORD	MD, KD	Input for the operand to be written
#OFF	DIRECT CONSTANT	#, #H	Offset address of the memory location to which the value of E1 must be written.
#SEG	DIRECT CONSTANT	#, #H	Segment address of the memory location to which the value of E1 must be written.

CE Data

Runtime:	
Basic runtime:	38 µs in case of enable; 19 µs in case of no enable.
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

If there is a 1 signal at the input FREI, the value of the operand at the input E1 is read and then written to the specified physical address.

No double word is written if there is a 0 signal at the FREI input.

The physical address consists of a segment and an offset.

The inputs can neither be duplicated nor negated.

FREI BINARY
Processing of the block is enabled or disabled with the operand at the input FREI. The following applies:
FREI = 0 -> Processing disabled
FREI = 1 -> Processing enabled

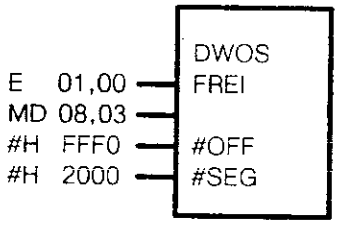
E1 DOUBLE WORD
The operand at the input E1 is read and its value is written to the address defined by the inputs #OFF and #SEG.

#OFF DIRECT CONSTANT (#,#H)
The offset component of the address to be written is specified at the input #OFF. This is specified as a direct constant.

#SEG DIRECT CONSTANT (#,#H)
The segment component of the address to be written is specified at the input #SEG. This is specified as a direct constant.

Example

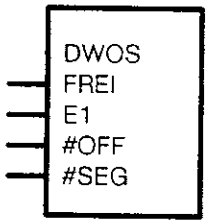
FBD/LD



IL

```
!BA    0
DWOS
       E   01,00
       MD  08,03
       #H  FFF0
       #H  2000
```

CE FBD Definition



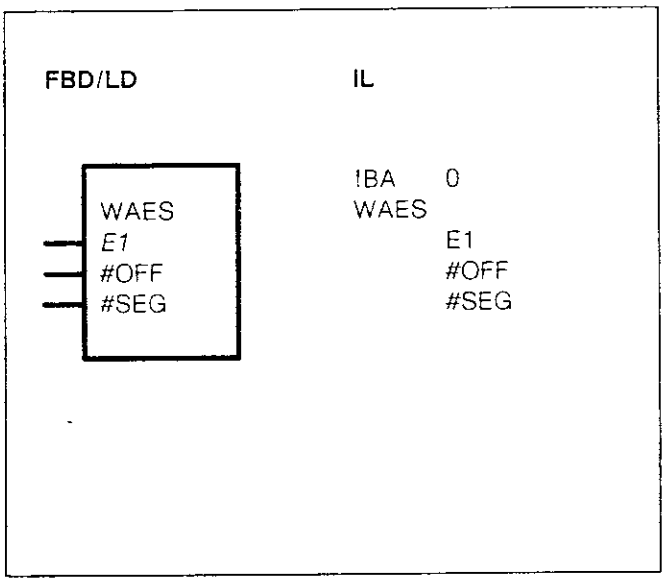
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
E1	E	D	N	P	N	0	0
#OFF	K	W	N	P	Y	0	0
#SEG	K	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (vorbelegt 0)
00001	DWOS			
00002		PP 0	FREI	Input BINARY (block enable)
00003		PP 0	E1	Input DOUBLE WORD (value)
00004		PP 0	#OFF	# CONSTANT (offset address)
00005		PP 0	#SEG	# CONSTANT (segment address)

WRITE WORD IN THE EVENT OF VALUE CHANGE — WAES

If the value of the operand at the input E1 compared to the value during previous processing of the block changes, the value of the operand at the input E1 is written to the specified physical address.



Parameters

E1	WORD	EW, MW, AW, KW	Input for the operand to be read
#OFF	DIRECT CONSTANT	#, #H	Offset address of the memory location to which the value of E1 must be written in the event of a change.
#SEG	DIRECT CONSTANT	#, #H	Segment address of the memory location to which the value of E1 must be written in the event of a change.

CE Data

Runtime:	
Basic runtime:	23 µs, no reading; 33 µs, reading
Additional runtime:	---
Output updating:	yes
Number of historical values:	1 word
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

If the value of the operand at the input E1 compared to the value during previous processing of the block changes, the value of the operand at the input E1 is written to the specified physical address.

The physical address consists of a segment and offset.

The inputs can neither be duplicated nor negated.

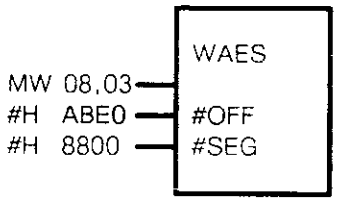
E1 WORD
If the operand at the input E1 changes, its value is written to the address specified at the inputs #OFF and #SEG.

#OFF DIRECT CONSTANT (#, #H)
The offset of the address to be written is specified at the input #OFF. This is specified as a direct constant.

#SEG DIRECT CONSTANT (#, #H)
The segment of the address to be written is specified at the input #SEG. This is specified as a direct constant.

Example

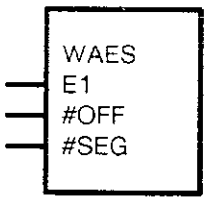
FBD/LD



IL

```
!BA    0
WAES
       MW 08,03
       #H ABE0
       #H 8800
```

CE FBD Definition



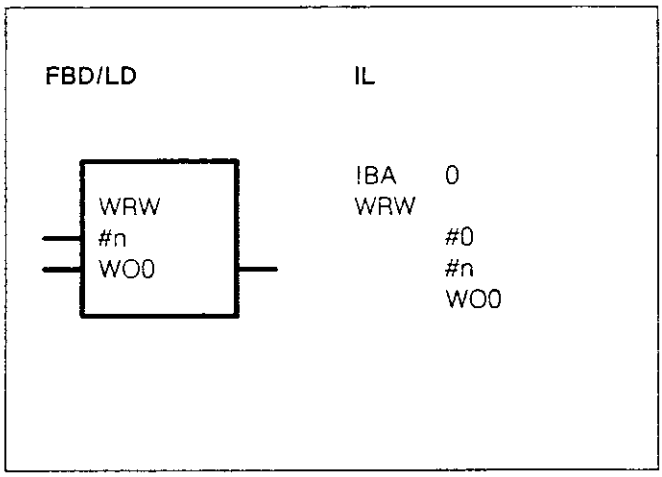
Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
E1	E	W	N	P	N	0	0
#OFF	K	W	N	P	Y	0	0
#SEG	K	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WAES			
00002		PP 0	E1	Input WORD (value)
00003		PP 0	#OFF	# CONSTANT (offset address)
00004		PP 0	#SEG	# CONSTANT (segment address)

WRITE WORD VALUES TO HISTORICAL VALUES MEMORY — WRW

This function block stores the values of the operands at the inputs WO0...WOn-1 in the historical values memory. The affiliated function block RDW reads these values out of the historical values memory again. The function blocks WRW and RDW always occur in pairs.



Parameters

#n	DIRECT CONSTANT	#, #H	Number of inputs WO0 ... WOn-1
WO0	WORD	AW, MW	Input for the word values to be written, capable of duplication (WO0 ... WOn-1)

CE Data

Runtime:	
Basic runtime:	see RDW block
Additional runtime:	---
Output updating:	not applicable
Number of historical values:	none
Available as of:	ABB Procontic T320 V6 / 935 PC 83 R301 / 35 ZE 93 R201

Description

This function block stores the values of the operands at the inputs WO0...WOn-1 in the historical values memory. The affiliated function block RDW reads these values out of the historical values memory again. The function blocks WRW and RDW always occur in pairs.

To be able to use ready-made program parts multiply in a user program (e.g. 907 PC 31/32 connection elements), it may be necessary to work with local variables within the program part. These local variables lose their validity outside of the program part concerned. The WRW block stores the values of these local variables at the end of the program part in the historical values memory of the affiliated RDW block and the RDW block allocates them to the local variables at the start of the program part. To do this, the WRW block stores the values of the word inputs WO0...WOn-1 in the historical values memory of the affiliated RDW block. The number of word inputs is specified as a direct constant at the input #n. This number must agree with the number of outputs belonging to the affiliated RDW block. At the intended input in the instruction list of the WRW block, the editing program "PA" enters the pointer to the historical values memory of the affiliated RDW block.

#0 DIRECT CONSTANT

This input *does not* exist in FBD/LD. In the instruction list, the value 0 must be specified here as a direct constant. The PLC then enters the pointer to the historical values of the affiliated RDW block at this point.

#n DIRECT CONSTANT

The number of the inputs WO0...WOn-1 is specified at the input #n. This is specified as a direct constant.

Note:

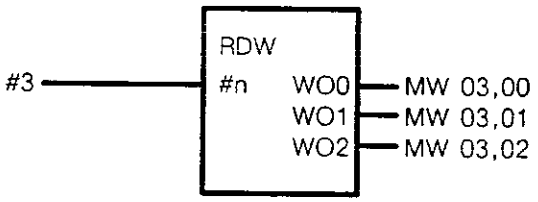
The value specified at the input #n must also agree with the number of the outputs belonging to the affiliated block RDW.

WO0 ... WOn-1 BINARY

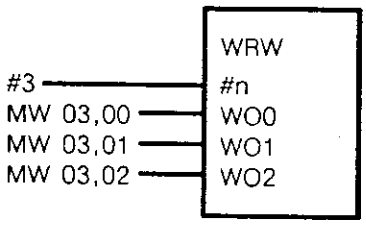
The input WO0 can be duplicated (WO0...WOn-1). The values of the operands specified at the inputs WO0...WOn-1 are written into the historical values memory of the affiliated RDW block.

Example

FBD/LD



The program part which is used multiply and in which the variables MW 03.00 ... MW 03.02 are used in the next cycle is located here. For that the values are written into the historical values memory by the WRW block and are being read out by the RDW block in the next cycle.



IL

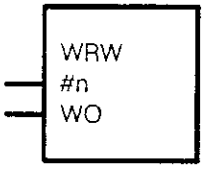
```
!BA    0
RDW
       #3
       MW 03,00
       MW 03,01
       MW 03,02
```

```
!BA    0
WRW
       #    0
       #    3
       MW 03,00
       MW 03,01
       MW 03,02
```

WRITE WORD VALUES TO HISTORICAL VALUES MEMORY

WRW

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
#n	K	W	N	P	Y	0	0
WO	E	W	N	P	Y	1	0

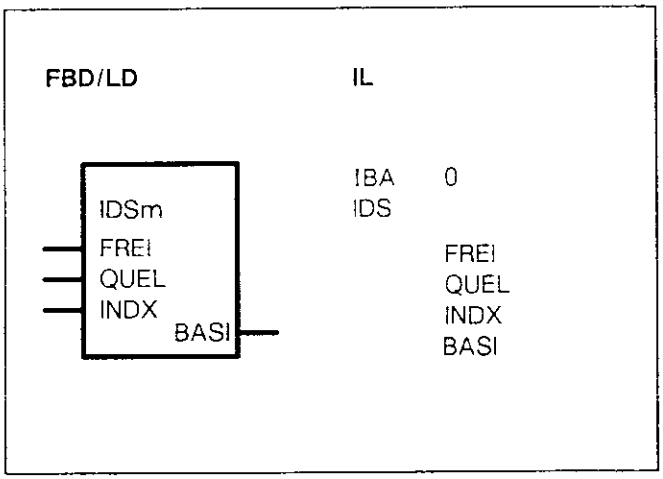
CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WRW			
00002		#0		
00003		PP 0	#n	#CONSTANT (number of words)
	[	1		
00004		PP 1	WO	Input WORD
	]	1		

WRITE WORD VARIABLE, INDEXED

IDSm

This function block serves the purpose of indexed writing of word variables.



Parameters

FREI	BINARY	E, A, M, K, S	Enable block FREI = 0: Block is not processed FREI = 1: The value of the source variable is read and allocated to the target variable
QUEL	WORD	EW, AW, MW, KW	Source variable
INDX	WORD	EW, AW, MW, KW	The current target variable results from the index and the basic variable
BASI	WORD	AW, MW	Basic variable

CE Data

Runtime:	
Basic runtime:	35 µs
Additional runtime:	none
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V7 / 935 PC 83 R401 / 35 ZE 93 R301

Description

This function block serves the purpose of indexed writing of word variables.

When the block is enabled, the value of the source variable is read and allocated to the target variable. The target variable is defined by indexing the basic variable.

The inputs and outputs can neither be duplicated nor negated nor inverted.

The group and channel number of the target flag (target variable) are determined on the basis of the basic flag and the index INDX.

The target flag is called: MW (G_Basis+A),(K_Basis+B) where:

G_Basis: Group number of the basic flag
K_Basis: Channel number of the basic flag

Formula:

$$\frac{\text{INDX}}{16} = \text{A} \quad \text{Remainder B}$$

Group No. of the target flag:
Group No. of the basic flag + A

Channel No. of the target flag:
Channel No. of the basic flag + B

Example:

Basic variable: MW 00,00
-> INDX = 10 -> 10 : 16 = A = 0, Remainder B = 10
-> Target variable:
MW(00+A), (00+B) = MW (00+0),(00+10) = MW 00,10
-> Target variable: MW 00,10

Further examples:

Basic variable	INDX	Target variable
MW 00,00	0	MW 00,00
MW 00,00	2	MW 00,02
MW 00,00	16	MW 01,00
MW 00,02	18	MW 01,04
AW 00,00	3	AW 00,03
AW 00,15	1	AW 01,00
AW 05,05	6	AW 05,11
AW 05,05	12	KW 06,01

FREI BINARY
Enable block

FREI=0: -> Block is not processed
FREI=1: -> The value of the source variable is read and allocated to the target variable ZIEL.

QUEL WORD
The source variable is specified at the input QUEL. The value of this variable is read and allocated to the target variable.

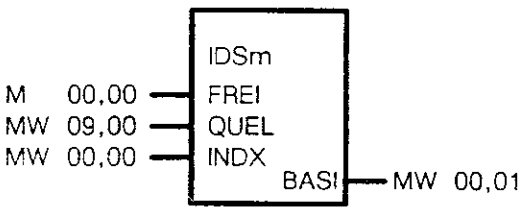
INDX WORD
The index value is specified at the input INDX. The target variable (see above for a calculation) results from the index INDX and the basic variable.

Value range: $-16383 \leq INDX \leq +16383$

BASI WORD
The basic variable is specified at the input BASI. The target variable (see above for a calculation) results from the index INDX and the basic variables.

Example

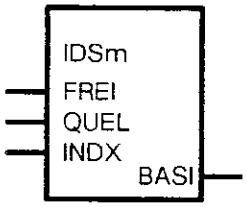
FBD/LD



IL

```
IBA   0
IDS
      M    00,00
      MW   09,00
      MW   00,00
      MW   00,01
```

CE FBD Definition

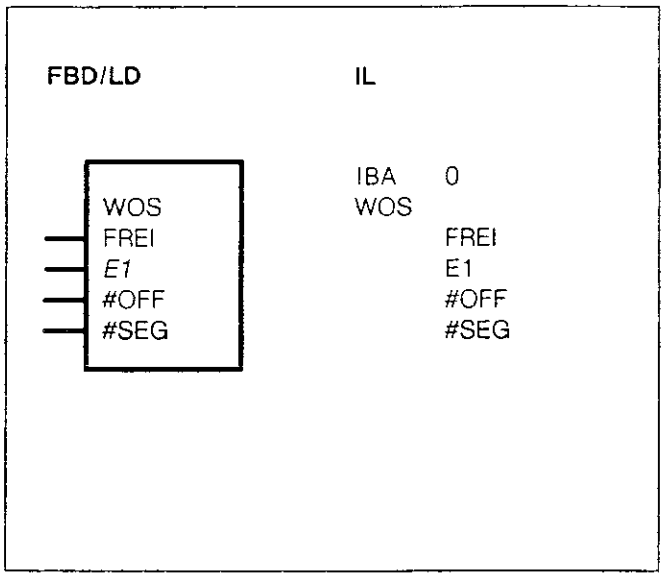


Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
QUEL	E	W	N	P	Y	0	0
INDX	E	W	N	P	Y	0	0
BASI	A	W	N	P	Y	0	0

CE IL Definition

00000	IBA	0	Nr	Block No. (preset to 0)
00001	IDS			
00002		PP 0	FREI	Enable (BINARY)
00003		PP 0	QUEL	Source variable (WORD)
00004		PP 0	INDX	Index relatet to basis (WORD)
00005		PP 0	BASI	Basic variable for target (WORD)

When the input FREI has a 1 signal, the value of the operand at the input E1 is read and is then written to the specified physical address.



Parameters

FREI	BINARY	E, M, A, S, K	Enable block
E1	WORD	EW, MW, AW, KW	Input for the operand to be written
#OFF	DIRECT CONSTANT	#, #H	Offset address of the memory location to which the value of E1 must be written
#SEG	DIRECT CONSTANT	#, #H	Segment address of the memory location to which the value of E1 must be written

CE Data

Runtime:	
Basic runtime:	25 µs, no writing; 39 µs, writing
Additional runtime:	---
Output updating:	yes
Number of historical values:	none
Available as of:	ABB Procontic T320 V3 / 935 PC 81 R701, R801 / 35 ZE 93 R101

Description

When the input FREI has a 1 signal, the value of the operand at the input E1 is read and is then written to the specified physical address.

The block is not processed if there is a 0 signal at the FREI input.

The physical address consists of a segment and offset. Thus, the attainable address area is 1 MByte.

The inputs can neither be duplicated nor negated.

FREI BINARY
Processing of the block is enabled or disabled with the operand at the input FREI.

The following applies:
FREI = 0 -> Processing disabled
FREI = 1 -> Processing enabled

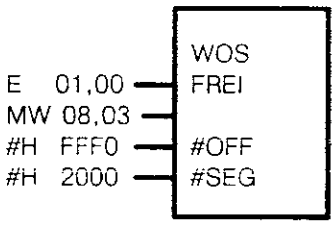
E1 WORD
The operand at the input E1 is read and its value is written to the address defined by the inputs #OFF and #SEG.

#OFF DIRECT CONSTANT (#,#H)
The offset of the address to be written is specified at the input #OFF. This is specified as a direct constant.

#SEG DIRECT CONSTANT (#,#H)
The segment of the address to be written is specified at the input #SEG. This is specified as a direct constant.

Example

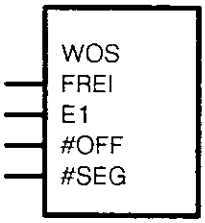
FBD/LD



IL

```
!BA   0
WOS
      E   01,00
      MW  08,03
      #H  FFF0
      #H  2000
```

CE FBD Definition



Param.	Param. Group	Param. Type	Inv.	Occupation	Displ. Screen	Param. Block	Dupli. Type
FREI	E	L	N	P	Y	0	0
E1	E	W	N	P	N	0	0
#OFF	K	W	N	P	Y	0	0
#SEG	K	W	N	P	Y	0	0

CE IL Definition

00000	!BA	0	Nr	Block No. (preset to 0)
00001	WOS			
00002		PP 0	FREI	Input BINARY (block enable)
00003		PP 0	E1	Input WORD (value)
00004		PP 0	#OFF	# CONSTANT (offset address)
00005		PP 0	#SEG	# CONSTANT (segment address)

ASEA BROWN BOVERI

ABB Schalt- und Steuerungstechnik GmbH
Eppelheimer Straße 82
W-6900 Heidelberg 1

Telephone (06221) 777-0
Telefax (06221) 777-111

Printed in the Federal Republic of Germany

Functional Description PLC

ABB Procontic T300

GATS 1315 11 R2001

Functional Description PLC

ABB Procontic T300

GATS 1315 11 R2001

Functional Description PLC

ABB Procontic T300

GATS 1315 11 R2001

Functional Description PLC

ABB Procontic T300

GATS 1315 11 R2001

Functional Description PLC

ABB Procontic T300

GATS 1315 11 R2001

Functional Description PLC

ABB Procontic T300

GATS 1315 11 R2001

Functional Description PLC

ABB Procontic T300

GATS 1315 11 R2001

Functional Description PLC

ABB Procontic T300

GATS 1315 11 R2001

Functional Description PLC

ABB Procontic T300

GATS 1315 11 R2001