Operating Manual

ABB Procontic CS31
Automation System
in Decentralized Structure

Operating and Test Functions
Monitor Functions
Memory Overview

For the description of the Operating and Test Functions, Monitor Functions and Memory Overview please refer to the System Description Advant Controller 31 (Section 7), Order No. 1SAC131699R0201. In this PDF documentation, these descriptions are added here.

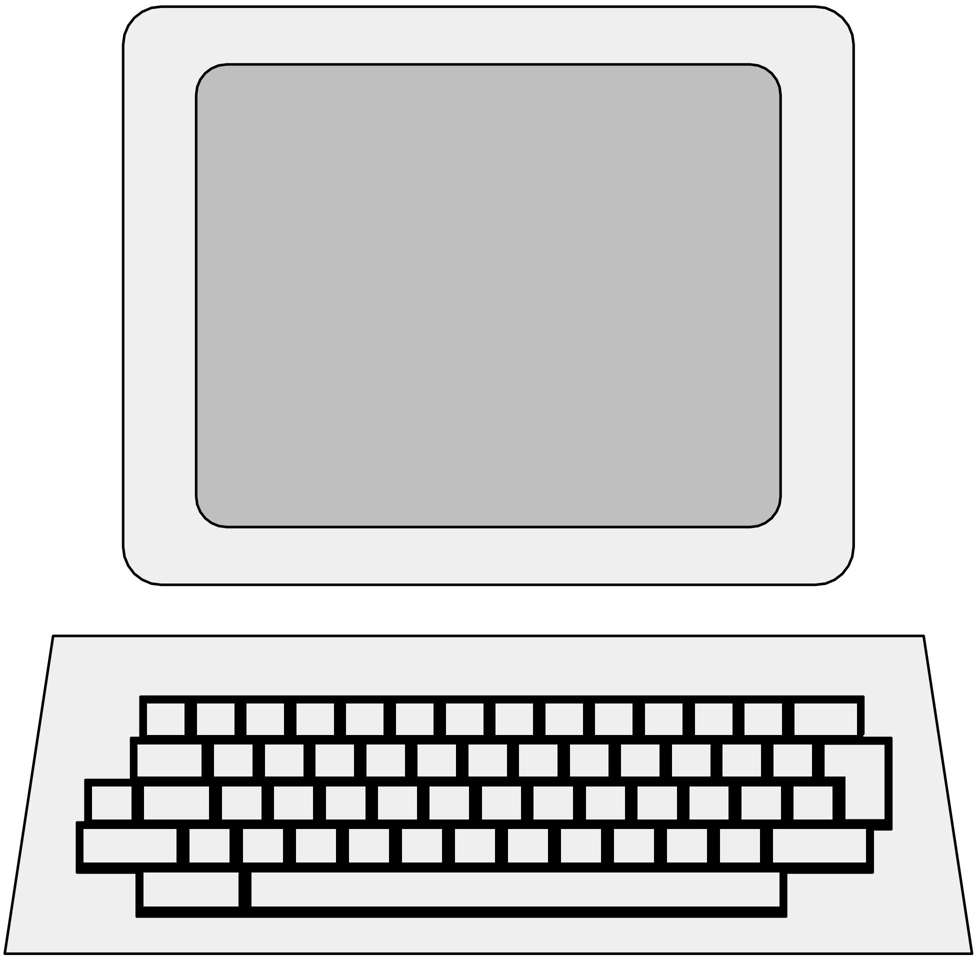


**ABB Schalt–
und Steuerungstechnik**

Advant Controller 31
Intelligent Decentralized Automation System

Programming

Table of contents, Volume 7.3

Note:

The individual chapters include a detailed table of contents when required.

1 Programming and test aids

1.1 Programming software 907 PC 33

General

The programming and test software 907 PC 33 is available as

- Programming and test software 907 PC 331 for ABB Procontic CS31 / Advant Controller 31 (basic units 07 KR 31, 07 KR 91, 07 KT 92, 07 KT 93, 07 KT 94) and ABB Procontic T200 (Communication processor 07 KP 62), order number GJP5204500R0102.
- Programming and test software 907 PC 332 for ABB Procontic T200, order number GJP5204300R0102.

The software products 907 PC 331 and 907 PC 332 are delivered respectively including system-specific documentation.

The functions which are indentical for both software products are described in the documentation "General part 907 PC 33". This documentation folder can be ordered separately with the following order number: GJP5203900R0102.

The software can be run on IBM/AT-compatible personal computers. An extensively automatic installation program installs the software package 907 PC 33 on this unit or on another IBM/AT-compatible personal computer.

The programming and test software 907 PC 33 permits a simple and economic programming of PLC programs in the following notations:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Extended instruction list (Exxt. IL)

Both symbolic and absolute program input is possible. The PLC program is supplemented by symbolic identifiers, long text, and commentary. Auxiliary and error messages which can be called at all times facilitate program input. Program creation as a FBD or as a LD takes place in a joint editor. Elements from the FBD and LD are therefore mixable and can be linked together. The library contains numerous connection elements and function blocks which considerably simplify the realization of complex functions (e.g. PID-type controller).

Features

The scope of the listed features depends on the capabilities of the individual PLCs.

General features

- All of the functions can be controlled with the mouse
- Clear display of project data and program configuration at one glance
- Scrollability in all directions in the editors
- Automatic recognition of revisions

Menu prompting

- Modern, clearly arranged menu interface employing pop-up menus
- Color display
- Quick selection of menu options with the mouse or on the keyboard
- Call-up of external programs on the DOS level directly from the menu (DOS shell)

Path information

- Input of data name with the affiliated DOS path
- Display of the project overview in a file directory

  Take-over of the file name incl. path, selection via cursor.

Password protection

- Several access privilege levels

Data safeguarding

- Data safeguarding directly from the editor
- Data safeguarding of complete projects on discs

Modularization

- Handling of large projects
- Arrangement of projects in logical structures
- Subdivision into program and variable modules
- Module change within the FBD/LD and the extended IL possible
- Modules can be called up from all levels (total project/ program module function selection)
- Simplified input of the module name and the corresponding file name

Segment plans

- The subdivision of the programs and/or program modules into segment plan yields a good program overview.
- Simple administration due to segment plan name and segment plan number

FBD/LD editor

- Uniform editor for programming with graphic symbols as function block diagram and ladder diagram
- Connection of ladder diagram networks with elements of the function block diagram

Extended IL editor

- Notation with symbols and long text in various forms
- Selection of links via a selection menu with the mouse
- Integration of the IL capabilites in the extended IL
- Translation is not conducted when the extended IL does not contain any connection elements
- The translated IL can be displayed, the segment plan structure is retained thereby. Also possible online.

Editor functions

An extensive spectrum of commands is available in the editors for program creation:

- Syntax test and plausibility test during the input of variables
- Block commands
  - for processing of program segments and variables
  - delete
  - shift
  - copy
  - store
  - load
  - print
  - delete unused variables
- Search commands
  - according to sentence number
  - according to word number
  - according to variable
  - according to symbol
  - according to command
  - according to line number
  - repeat
  - according to segment plan
  - according to connection element
  - according to unassigned terminal
- Search and replace
- Insert
- Delete

Library

- Operating interface with mouse
- Programming of a connection element in the FBD/LD
- Hierarchical arrangement possibility of the connection elements (similar to DOS directories)
- Auxiliary texts and short commentaries for connection elements
- Terminal allocation test for the timely recognition of program errors.
- For every manufacturer connection element a detailed function description can be called up directly out of the FBD as a help text.

Variable editor

- Complete list of all of the entered variables
- Sorting selectable according to absolute or symbolic variables
- One or more symbol names can be allocated to the variables
- Adoption and transfer of the variable lists to and from any word processing system
- Provision and adoption of variable lists for specific CAD/ CAE systems

Text editor

- Input of any ASCII files, up to 255 characters per line

Commentaries

- Verbal description of networks or program segments

ONLINE functions

Numerous ONLINE functions support the user during the commissioning phase, e.g.

- Status display in
  - function block diagram
  - ladder diagram
  - instruction list
  - variable list
- Program
  - transfer
  - start
  - abort
  - stop
  - continuation
  - status
- Single cycle on/off
- Single step on/off
- Breakpoint
  - setting
  - display
  - tracing during the complete program
  - delete
- Triggering
  - time
  - variable
- Overwriting
- Jogging
- Forcing
- Modification of
  - time and counter setpoints
  - variables
  - programm segments
- Online program modification

In addition selected variables can be summarized in ONLINE lists and their status can be displayed on the screen.

- "Hotkeys" for quicker operation
- Switch into ONLINE operation directly out of the FBD/LD, extended IL, variable list, ONLINE list.
- Direct PLC communication, e.g. "Send program" out of the editors
- Translate and transfer program modifications with press of a key
- Direct "overwriting" and "forcing" out of the editors
- ONLINE list with direct adoption of variables from the PLC program to "forcing", "overwriting"
- Simple setting of breakpoints with the cursor also in the FBD
- ONLINE notation of variables in various numeric forms (decimal, octal, hexadecimal, binary).

Program documentation

The automatic program documentation includes the printing of the following lists:

- Function block diagram
- Instruction list
- Connection element library
- Logic plan diagram
- Ladder diagram
- Variable list
- Cross reference list
- Commentary list
- ONLINE list
- Text page
- Data area
- Modularization list
- Total variable list
- Total cross reference list
- System configuration

Outputs can be adapted to any printer.

Printing format editor

A special printing format enables the addition of individual headers and footers to the respective list. Specific data can be included in this header and/or footer, e.g. name of the project file, date and time.

1.2 Programming via ARCNET

It is possible to program the control system of the AC31 series via ARCNET on DOS level. The following is required to do this:

- PC with installed ARCNET card or coupler connected to a parallel printer interface
- AC31 basic modules with integrated ARCNET coupler
- Special driver software
- 907 PC 331

For further information see description 907 PC 331 R0402.

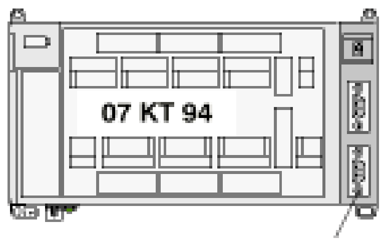
2 Access, operating and test functions

Table of contents of the chapter Access, operating and test functions

2.1 Introduction

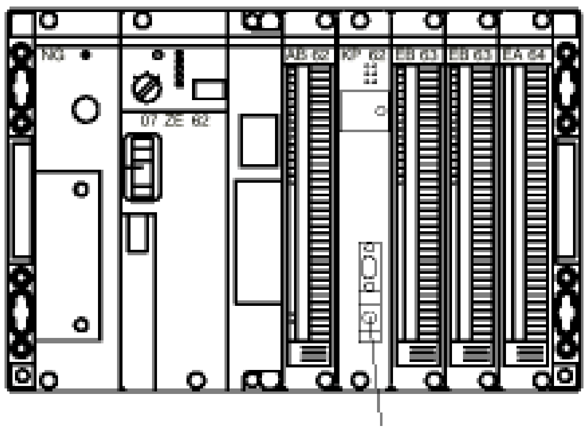
2.1.1 Access to the basic units 07 KR/KT 31, 07 KR 91, 07 KT 92, 07 KT 93, 07 KT 94 and to the communication processor 07 KP 62 of ABB Procontic T200

The access to the AC31/CS31 basic units (07 KR 31, 07 KR 91, 07 KT 92 to 07 KT 94) and to the communication processor 07 KP 62 of ABB Procontic T200 is conducted via the serial interface COM1.



Interface COM1

Fig. 1: Advant Controller 31 / ABB Procontic CS31: Access to control via the interface COM1 of 07KR31 / 07 KT 31, 07 KR 91, 07 KT 92 / 07 KT 93 / 07 KT 94



Interface COM1

Fig 2: ABB Procontic T200: Access to control via the interface COM1 of 07 KP 62

Every operating and test function of the PLC can be called via an ASCII clear text telegram. The operating mode "active mode" must be set on the serial interface.

Connectable units:
- Terminal in the VT100 mode
- Computer with VT100 emulation
- Computer with a program for the handling of the clear text telegrams of the operating and test functions

2.1.2 Interface standard

Interface standard: EIA–RS232

2.1.3 Interface operating mode

The serial interface COM 1 must be set the operating mode **"Active mode"** to use the operating and test functions.

RUN/STOP switch in position: STOP
In the switch position STOP the PLC generally sets the operating mode "active mode" on COM 1.

RUN/STOP switch in position: RUN
In the switch position RUN the operating mode "active mode" is set on COM 1, when one of the following two conditions is fulfilled:

- System constant KW 00,06 = 1

or

- System constant KW 00,06 = 0 **and** Pin 6 on COM1 has 1-signal (1-signal on Pin 6 is set by using the system cable 07 SK 90 or by not connecting Pin 6)

2.1.4 System behavior of the PLC

The following applies:

The processing of the PLC program has higher priority than the communication via the serial interfaces.

The PLC operates the receiving direction of the serial interface COM1 with interrupt-control. During a running PLC program cycle incoming characters respectively trigger an interrupt impulse, which interrupts the running PLC program until the received characters are stored in the reception buffer. To avoid a permanent interruption of the program processing, the PLC controls the data reception via the RTS line so that it takes place in the breaks between two PLC cycles.

The PLC processes the jobs received via COM1 exclusively in the breaks between the PLC program cycles. The output of characters via COM1 is also only conducted in the breaks between two program cycles. The lower the utilization rate of the PLC is, the longer the breaks are between the program cycles and the higher the possible communication rate is to COM1.

2.1.5 Synchronization of the data exchange

The synchronization of the data exchange between the control and the connected unit is conducted via the hardware handshake lines RTS and CTS.

The PLC blocks data reception via the RTS line under the following limiting conditions:

– Reception buffer has reached a certain fill.

– A PLC program cycle is just running.

The control still reacts during output of characters in addition to the XOFF/XON characters of the connected unit. The control itself does not use these SW handshakes.

2.1.6 Echo

In the breaks between two PLC program cycles the PLC processes the jobs collected in the reception buffer. To do this, the characters are read out of the interrupt-controlled reception buffer by the PLC, immediately echoed via COM 1, checked for correct syntax, and then processed. The characters are echoed in the same order as received via COM 1.

2.1.7 Abort of a character output

By sending <CTRL C> the connected unit can cause the PLC to abort the currently running character output. The respective operating or test function is also aborted thereby.

2.1.8 Ready message

After complete processing of an operating or test function the PLC is ready again for a new job. This readiness is signalized by the output of the ASCII characters

CR LF **>** ($0D_H$ $0A_H$ $3E_H$)

via COM 1.

This means that the ASCII character **>** (larger than) is set at the beginning of a new line.

2.1.9 Error message

If the PLC receives an unallowed job or a job with incorrect syntax, this is signalized via COM 1 as follows:

<# Error message as clear text>

The readiness for a new order is then signalized by the output of the ASCII character

CR LF **>** ($0D_H$ $0A_H$ $3E_H$)

via COM 1.

This means that the ASCII character **>** (larger than) is set at the beginning of a new line.

2.1.10 Notes on implementation

If the operating and test functions of the PLC should be called by a computer connected to COM 1, these functions can first be easily tried out with a terminal in the VT100 mode.
If the operating and test functions are used for the man-machine communication (MMC), then mostly the following functions are required:

- **Overwrite variable / indirect constant**
  Y command

- **Display status of variable**
  Z command
  ZO command
  ZD command
  ZZ command

If a computer is connected to COM1, the ZZ command is recommended. With the ZZ command the PLC does not send any ESC sequences to the cursor control.

- **Enter/modify values of indirect constants**
  K command

2.2 Operands

During man-machine communication the display and modification of operands play a large role. For this reason an overview of all of the operands of the PLC is given here.

2.2.1 Operands of 07 KT 94

2.2.1.1 Available variables and constants

Inputs

E 00,00...E 61,15	:	Digital inputs, CS31 remote module
E 62,00...E 63,15	:	Digital inputs of the basic unit 07 KT 94
E 64,00...E 64,07	:	Digital inputs of the basic unit 07 KT 94 (formed of EW 06,00...EW 6,07)
E 65,00...E 99,15	:	reserved
E 100,00...E 163,15	:	reserved
E 200,00...E 263,15	:	reserved
EW 00,00...EW 05,15	:	Analog inputs, CS31 remote module
EW 06,00...EW 06,07	:	Analog inputs of the basic unit 07 KT 94
EW 07,00...EW 07,07	:	reserved
EW 07,08...EW 07,14	:	Reading of the real time clock
EW 07,15	:	Status for CS31 system bus
EW 08,00...EW 15,15	:	Analog inputs CS31 remote module
EW 16,00...EW 34,15	:	reserved
EW 100,00...EW 107,15	:	reserved
EW 200,00...EW 207,15	:	reserved

Outputs

A 00,00...A 61,15	:	Digital outputs, CS31 remote module
A 62,00...A 63,07	:	Digital outputs of the basic unit 07 KT 94
A 62,00	:	High-speed counter, after activation direct output of the "zero crossing"
A 65,00...A 99,15	:	reserved
A 100,00...A 163,15	:	reserved
A 200,00...A 263,15	:	reserved
AW 00,00...AW 05,15	:	Analog outputs, CS31 remote module
AW 06,00...AW 06,03	:	Analog outputs of the basic unit 07 KT 94
AW 07,00...AW 07,15	:	reserved
AW 08,00...AW 15,15	:	Analog outputs CS31 remote module
AW 16,00...AW 34,15	:	reserved
AW 100,00...AW 107,15	:	reserved
AW 200,00...AW 207,15	:	reserved

Internal Operands

M 00,00...M 254,15	:	Binary flags
M 255,00	:	Oscillator approx. 2 Hz
M 255,01	:	Oscillator approx. 1 Hz
M 255,02	:	Oscillator approx. 0,5 Hz
M 255,03	:	Oscillator with period interval of approx. 1 minute
M 255,04	:	Oscillator approx. 1/8 Hz
M 255,05	:	Oscillator approx. 4 Hz
M 255,06	:	Oscillator approx. 8 Hz
M 255,10	:	Sum error message
M 255,11	:	Error message FK1
M 255,12	:	Error message FK2
M 255,13	:	Error message FK3
M 255,14	:	Error message FK4
M 255,15	:	Recognition "new start"
M 256,00...M 279,15	:	System flags / reserved
M 280,00...M 511,15	:	Binary flags

S 00,00...S 255,15	:	Steps
K 00,00...K 00,01	:	Binary constants

MW 00,00...MW 253,15	:	Word flags
MW 254,00...MW 255,15	:	Error message
MW 256,00...MW 259,15	:	System flags / reserved
MW 260,00...MW 511,15	:	User area
KW 01,00...KW 79,15	:	Word constants

MD 00,00...MD 63,15	:	Double word flags
KD 00,01...KD 23,15	:	Double word constants

Time values for time functions

KD yy,xx : Time values for time functions such as ESV, ASV etc. are configured as double word constant or as
MD yy,xx : double word flags. Only integer multiples of 1 ms are permitted.

2.2.1.2 Direct constants

Direct constants are only permitted with function blocks on certain inputs. Where this is the case it is explained in the description of the function modules.

#	-32768...+32767
#H	0000...FFFF

2.2.1.3 Labels

Labels serve as jump targets for forward jumps and consecutive number blocks.

MA 0...999

2.2.1.4 System constants

Setting the operating modes

The constants KW 00,00...KW 00,15 are reserved as system constants. Even the constants KW 00,13...KW 00,15 which are not used yet may under no circumstances be used for other purposes.

KW 00,00	:	Setting the PLC operating mode (stand-alone PLC, master PLC, slave PLC)
KW 00,01	:	Initialization: bit flag area
KW 00,02	:	Initialization: word flag area
KW 00,03	:	Initialization: double word flag area
KW 00,04	:	Initialization: step chain flag area
KW 00,05	:	Initialization: historical values
KW 00,06	:	Application modes of the serial interface COM 1
KW 00,07	:	PLC reaction to class 3 errors
KW 00,08	:	PLC reaction to an overload/short-circuit at the transistor outputs
KW 00,09	:	Minimum number of remote modules integrated in the CS31 system bus cycle
KW 00,10	:	Size of the transmitting area of the slave PLC
KW 00,11	:	Size of the receiving area of the slave PLC
KW 00,12	:	Automatic warm start after an FK2 error
KW 00,15	:	Deactivate oscillators at M 255,00...M 255,06

KW 85,00...KW 85,03	:	Configuration of the signal delay of digital inputs
KW 85,02	:	Configuration of the operating modes of the high-speed counter
KW 86,00...KW 86,07	:	Configuration of the analog inputs
KW 88,00...KW 88,03	:	Configuration of the analog outputs

Setting the cycle time

KD 00,00 : This constant serves as the specification of the cycle time for the PLC program. The cycle time is given in milliseconds. Only integer multiples of 1 ms are permitted.

2.2.1.5 System flags / Diagnosis flags

M 00,00...M 254,15	:	Binary flags
M 255,00	:	Oscillator approx. 2 Hz
M 255,01	:	Oscillator approx. 1 Hz
M 255,02	:	Oscillator approx. 0.5 Hz
M 255,03	:	Oscillator with period interval of approx. 1 minute
M 255,04	:	Oscillator approx. 1/8 Hz
M 255,05	:	Oscillator approx. 4 Hz
M 255,06	:	Oscillator approx. 8 Hz

M 255,10 : Sum error message,	signalizes that an error was detected by the PLC
M 255,11 : Error messageFK1, fatal error,	detailed information in MW 254,00...MW 254,07
M 255,12 : Error messageFK2, serious error,	detailed information in MW 254,08...MW 254,15
M 255,13 : Error messageFK3, light error,	detailed information in MW 255,00...MW 255,07
M 255,14 : Error messageFK4, warning,	detailed information in MW 255,08...MW 255,15

M 255,15 : Detection "new start"

MW 254,00...MW 255,15 : error messages

First cycle detection

M 255,15

This binary flag can be used for detection of the first program cycle after a program start. It is always set to "zero" after each program start, independent of the initialization instructions given by the system constants. If this flag is requested by the user program and then set to "1", it can be determined whether or not the user program was re-started.

2.2.1.6 CS31 status word

EW 07,15

Bit 0 = 1	:	no CS31 error of class 2 present
Bit 1 = 1	:	PLC has been adopted in CS31 cycle (only relevant when used as a slave)
Bit 2 = 1	:	Time and date are valid
Bit 3 = 1	:	Battery present
Bit 4...7	:	unused
Bit 8..15	:	currently determined maximum number of modules on CS31 system bus (only relevant when used as a master)

2.2.2 Operands of 07 KR 31 and 07 KT 31

2.2.2.1 Available variables and constants

Inputs

E 00,00...E 61,15	:	Digital inputs, CS31 remote module
E 62,00...E 62,11	:	Digital inputs of the basic unit 07 KR 31 / 07 KT 31
E 63,14	:	Digital inputs high-speed (Tv = 0.02 ms), signal identical to E 62,00
E 63,13	:	high-speed counter, request "zero crossing"
EW 00,00...EW 05,15	:	Analog inputs, CS31 remote module
EW 06,15	:	high-speed counter, request "zero crossing"
EW 07,00...EW 07,07	:	reserved (for diagnosis on the CS31 system bus)
EW 07,08...EW 07,14	:	Read the real time clock
EW 07,15	:	Status for CS31 system bus

Outputs

A 00,00...A 61,15	:	Digital outputs, CS31 remote module
A 62,00...A 62,07	:	Digital relay outputs of the basic unit 07 KR 31 / 07 KT 31
A 63,15	:	high-speed counter, adopt inital value
AW 00,00...AW 05,15	:	Analog outputs, CS31 remote module
AW 06,15	:	high-speed counter, initial value

Internal Operands

M 00,00...M 21,15	:	Binary flags
M 230,00...M 239,15		
M 255,00...M 255,15	:	Diagnosis flags
S 00,00...S 15,15	:	Steps
K 00...K 00,01	:	Binary constants
MW 00,00...MW 05,15	:	Word flags
MW 230,00...MW 239,15		
MW 254,00...MW 255,15	:	Diagnosis words
KW 00,00...KW 07,15	:	Word constants
MD 00,00...MD 01,15	:	Double word flags
KD 00,00...KD 01,15	:	Double word constants

Time values for time functions

KD yy,xx : Time values for time functions such as ESV, ASV etc. are configured as double word constant or as
MD yy,xx : double word flags. Only integer multiples of 5 ms are permitted.

2.2.2.2 Direct constants

Direct constants are only permitted with function blocks on certain inputs. Where this is the case it is explained in the description of the function modules.

#	-32768...+32767
#H	0000...FFFF

2.2.2.3 Labels

Labels serve as jump targets for forward jumps and consecutive number blocks.
MA 0...999

2.2.2.4 System constants

Identical to chapter 2.2.1.4, except of:
KW 00,08 : not used

2.2.2.5 Diagnosis flags

Identical to chapter 2.2.1.5

2.2.2.6 CS31 status

Identical to chapter 2.2.1.6, except of:
Bit 2...7 : not used

2.2.3 Operands of 07 KP 62

2.2.3.1 Available variables and constants

Inputs
The module has no process inputs.
EW 00,04...EW 00,07 : high-speed inputs from ABB Procontic T200

Outputs
The module has no process outputs.
AW 00,04...AW 00,03 : high-speed outputs to ABB Procontic T200

Internal operands

MW 00,00...MW 05,15	:	Output flags to ABB Procontic T200
MW 06,00...MW 11,15	:	Input flags from ABB Procontic T200
MW 12,00...MW 253,15	:	free word flags

Setting the cycle time

KD 00,00 : This constant serves as the specification of the cycle time for the PLC program. The cycle time is given in milliseconds. Only integer multiples of 5 ms are permitted.

Time values for time functions

KD yy,xx	:	Time values for time functions such as ESV, ASV etc. are configured as double word constant or as
MD yy,xx	:	double word flags. Only integer multiples of 5 ms are permitted.

2.2.3.2 Direct constants

Direct constants are only permitted with function blocks on certain inputs. Where this is the case it is explained in the description of the function modules.

#	-32768...+32767
#H	0000...FFFF

2.2.3.3 Labels

Labels serve as jump targets for forward jumps and consecutive number blocks.
MA 0...999

2.2.3.4 System constants

Setting the operating modes

The constants KW 00,00...KW 00,15 are reserved as system constants. Even the constants KW 00,12...KW 00,15 which are not used yet may under no circumstances be used for other purposes.

In module 07 KP 62 there are only the system constants KW 00,01...KW 00,07.

Setting the cycle time

KD 00,00 : This constant serves as the specification of the cycle time for the PLC program. The cycle time is given in milliseconds. Only integer multiples of 5 ms are permitted.

2.2.3.5 Diagnosis flags

M 255,10 : Sum error message,	signalizes that an error was detected by the PLC
M 255,11 : Error message FK1, fatal error,	detailed information in MW 254,00...MW 254,07
M 255,12 : Error message FK2, serious error,	detailed information in MW 254,08...MW 254,15
M 255,13 : Error message FK3, light error,	detailed information in MW 255,00...MW 255,07
M 255,14 : Error message FK4, warning,	detailed information in MW 255,08...MW 255,15

First cycle detection

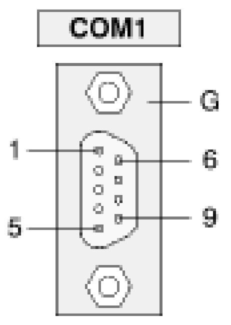
M 255,15
This binary flag can be used for detection of the first program cycle after a program start. It is always set to "zero" after each program start, independent of the initialization instructions given by the system constants. If this flag is requested by the user program and then set to "1", it can be determined whether or not the user program was restarted.

2.3 Serial interface COM1

Interface standard: EIA RS-232

Assignments of the serial interface COM1

The serial interface COM1 has the following connection assignment:



G	Housing	Protective Ground	(Shield)
1	PGND	Protective Ground	(Shield)
2	TxD	Transmit Data	(Output)
3	RxD	Receive Data	(Input)
4	RTS	Request To Send	(Output)
5	CTS	Clear To Send	(Input)
6	PROG *		(Input)
7	SGND	Signal Ground	(0V)
8	0V out		(0V)
9	+5 V out	reserved	

* 1 = Active mode (Programming/Test) Pin 6 open
0 = Passive mode (PRINT/EMAS applications), Pin 6 short-circuited with 0V out

Fig. 3: Assignment of the serial interface COM1

Operating modes of the serial interface COM1

The operating mode of the interface must be set according to the respective application:
– Programing and test or
– man-machine communication MMC

Active mode: The active mode is used for programming and testing of the basic unit, i.e. it provides access to all programming and test functions of the basic unit.

Passive mode: The passive mode is used to conduct communication configured with the DRUCK und EMAS blocks between the user program and a device connected to the serial interface.

Conditions for setting the operating modes of the serial interface COM1

RUN/ STOP- Switch	System constant KW00,06	System cable/ device	Mode set by this
STOP	x	x	Active
RUN	1	x	Active
RUN	2	x	Passive
RUN	0, <0, >2	07 SK 90	Active
RUN	0, <0, >2	07 SK 91	Passive

x: without effect

Temporary leaving the passive mode

During a running communication between the blocks DRUCK and/or EMAS and a module connected to COM1, it can become necessary to change a program. To do this COM1 must be switched from the passive mode to the active mode.

Switching: Passive mode ––> Active mode

The following three possibilities apply for switching:

- Set the RUN/STOP switch to "STOP" position
- Replace the cable 07 SK 91 with the cable 07 SK 90 (when KW 00,06 is set to <0 or >2)
- Send the following special command to the PLC: <DEL><DEL><DEL>

The third possibility also enables the switching to be performed remote-controlled, e.g. via telephone lines and suitable dialing modems. The ASCII character <DEL> has the decimal code 127 and the hexadecimal code $7F_H$. This character is created on the PC by simultaneously pressing the control key <CTRL> and the (backspace) delete key <—.

Notes:

On German keyboards the control key is not labelled with <CTRL> but with <Strg>.

If switching into the active mode is carried out with the special command <DEL><DEL><DEL>, the following applies:

During the running PLC program the system constant KW 00,06 **must not** be sent to the PLC because it will switch back to the passive mode.

The special command allots the value "1" to the image of the system constants KW 00,06 stored in the operand memory. The PLC evaluates the value of this image and sets the application mode of COM1 accordingly.

Switching back: Active mode ––> Passive mode

The three possibilities to switch back are as follows:

- Set the RUN/STOP switch back to the "RUN" position
- Replace the cable 07 SK 90 again with the cable 07 SK 91
- Cancel the special command <DEL><DEL><DEL> again as follows:
  - If the PLC program has been "interrupted":

    start the PLC program.
  - If the PLC program is "running":

    re-send the original value of the system constants KW 00,06 to the control (907 PC 33 menu option "send constants")

    or

    overwrite the system constants KW 00,06 with the original value (907 PC 33 menu option "overwrite")

Interface parameters

Active mode: The settings of the interface parameter can not be changed.

Data bits:	8
Stop bits:	1
Parity bit:	none
Baud rate:	9600
Synchronization:	RTS/CTS

Passive mode: Default–setting

Synchronization:	RTS/CTS	
Interface identifier COM1:	1	
Baud rate:	9600	
Stop bits:	1	
Data bits:	8	
Parity Bit:	none	
Echo:	off	
Send Break Character:	0	
Enable end-of-text-character for sending direction:	no	1)
Sending end-of-text character:	<CR>	1)
Receiving end-of-text character:	<CR>	2)

1) The default end-of-text character for the sending direction (CR) is not sent. Nevertheless, this default end-of-text character (CR) must not appear in the message text of the assigned DRUCK block.

2) For the receive direction, an end-of-text character is always necessary. This default end-of-text character (CR) must not appear neither in the message text nor in the user data of the assigned EMAS block.

For the passive mode of COM1, the interface parameters can be changed using the SINIT function block. If the changed values are not plausible, the COM1 interface uses the default values.

Every time the operating mode is switched the interface is re-initialized.

In the active mode the active mode parameters are set, in the passive mode the paramters defined by SINIT and/or the default values are set.

2.4 Operating and test functions

Operator control commands

The operator control commands can be subdivided into:

- Commands for creating and modifying user programs
- Commands for testing the user programs
- Commands for configuring the PLC

Notes:

- User entries require no "blanks". Any "blanks" entered are ignored.
- In order to provide greater clarity when describing the commands, the user entries
  - for keywords are shown in

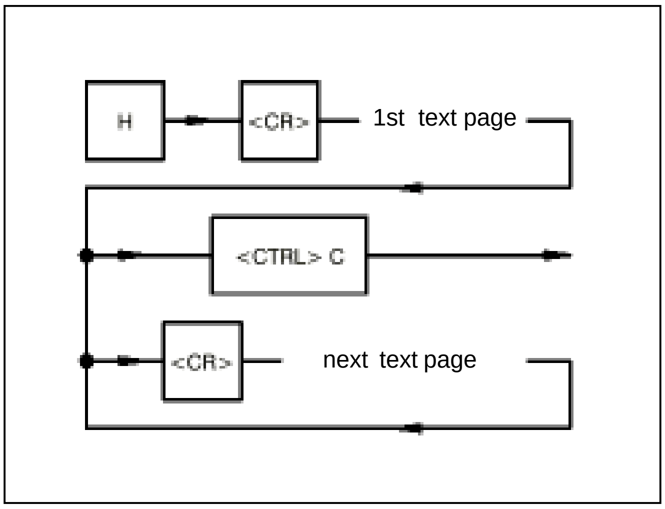
    UPPER-CASE LETTERS
  - and other entries (addresses etc.) are shown

    in lower-case letters
- Outputs generated by the PLC software on the monitor are shown in

  lower-case italics

All available commands are displayed with the HELP command on the monitor.

Help command



Function:
All available operator control and test functions are displayed on the monitor. Use <CR> to scroll the HELP text.

Commands for creating the user program (overview)

Command and function		Page
AEND	Prepare a program change on a running PLC program	18
AEND	Reject a program change which has not yet been enabled	18
ALT	Reject an enabled program change on a running PLC program and reactivate the old program status	18
AL	Display capacity utilization	18
CROSS *)	Display CROSS reference list	19
D	Display program	20
DEEP	Erase PLC program from Flash EPROM	20
F *)	Search for string in user program	20
FREI	Enable a program change on a running PLC program	20
FRD	Read data files from Flash EPROM	21
FWR	Write data files to Flash EPROM	21
FDEL	Delete data segment in Flash EPROM	21
FCINIT *)	Initialize SmartMedia Card	22
FCRD *)	Read data from SmartMedia Card	22
FCWR *)	Write data to SmartMedia Card	22
FCDEL *)	Delete data segment in SmartMedia Card	22
SIZE16	Double user program memory	23
IDA	Display program identification	23
IDR	Delete program identification	23
IDS	Enter program identification	23
K	Enter/edit values of indirect constants	23
NOP	Delete programm part, i.e. overwrite with NOPs	24
O	Optimize program	25
P	Display free program memory	25
PA *)	User programm preparation	25
S	Enter/edit user program (Substitute)	25
SO *)	Enter/edit user program without echo	25
SP	Save PLC program in Flash EPROM and in SmartMedia Card	26

*) **not** with series 30, 40, 50
) **only with series 30, 40, 50

Commands for testing the user program (overview)

Command and function		Page
V	Move user program	26
A	Abort user program	27
BA*)	Diplay break points	27
BR *)	Reset break points	27
BS *)	Set break points	27
<CTRL>W *)	Switching between operator control functions <—> monitor	28
EA *)	I/O test mode	28
EAA *)	Deactivate I/O test mode	28
ES *)	Single step mode ON	28
ESA *)	Single step mode OFF	28
EZ *)	Single cycle mode ON	28
EZA *)	Single cycle mode OFF	29
FEHLER	Display contents of the error register	29
FORC	Enter force value	30
FORC A	Display force value	30
FORC R	Delete forcing	30
G	Start user program	30
KALT	Perform cold start	31
WARM	Perform warm start	31

Command and function		Page
L *)	Continue user program	31
PS	Display program status	31
ST	Display PLC status	32
TRACE *)	TRACE-mode	32
TRACE *)	Display TRACE memory	32
TRACE E *)	Activate TRACE mode	32
TRACE A *)	Deactivate TRACE mode	32
W *)	Stop user program	33
Y	Overwrite value of a variable with a value to be defined	33
Z	Display status of variables	33
ZZ	Display only the values of the variables	34
ZD	Display and continually update status of variables	34

Commands for configuration

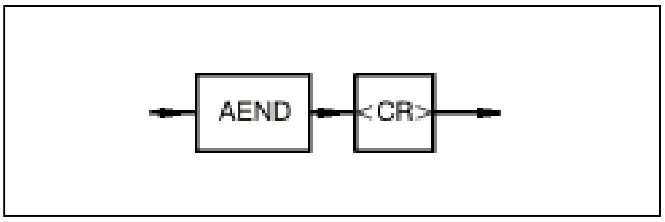
KONFS	Display/change operating modes	35
MAIL	Configuration of CS31 remote modules	35
PASS **)	Password	39
UHR	Display time and date	39
UHRS	Set time and date	40

) **not with series 30, 40, 50
) **only with series 30, 40, 50

2.4.1 Commands for creating the user program

Preparing a program change on a running PLC program

Command:



Function:

The command announces to the PLC that modifications are to be carried out on the running PLC program. After this command has been entered, the PLC is ready to accept the program and constant modifications.

When command AEND is entered, all currently active test functions are deactivated. However, force values of I/O signals remain active.

The following commands for program modifications and operation of the PLC are permitted after entering command AEND:

AL, CROSS, D, F, IDA, IDR, IDS, K, N, NOP, O, P, PA, S, SO, V, CTRL W, FEHLER, LED.

Rejecting a program change which has not been enabled yet

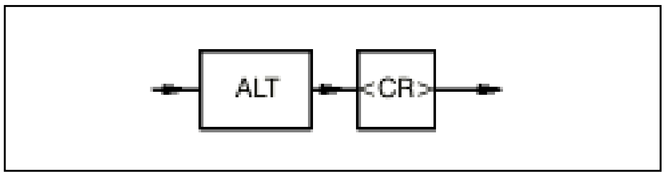
Entering the AEND command again rejects all program modifications performed to date, and the PLC is ready to accept program modifications again.

The following commands are activated during the **running** program **and** in addition reject the AEND command and thus all program modifications performed after entry 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. In order to perform new program modifications, the command AEND must be entered again.

Rejecting an enabled change on a running PLC program and reactivating the old program status

Command:



Function:

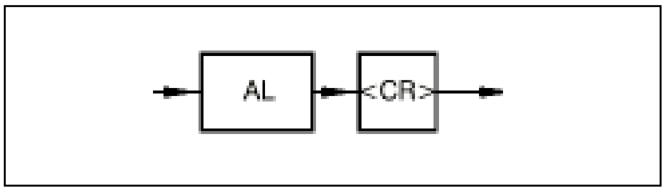
Modifications which have been performed on a running PLC program **and** which have been enabled are rejected again. In addition, the PLC restores the old program status. The old program status is the status of the program which existed before the program modification, i.e. **before** entry of command AEND to the PLC.

After command ALT is entered, the old program status is reactivated within approximately 1 ms without further intervention on the part of the user.

The command can be used if the user recognizes that the program modifications implemented do not achieve the intended result.

Display capacity utilization

Command:



Function:

The PLC's present capacity utilization is displayed in percent. This display indicates to what extent the capacity of the PLC is being utilized owing to execution of the user program.

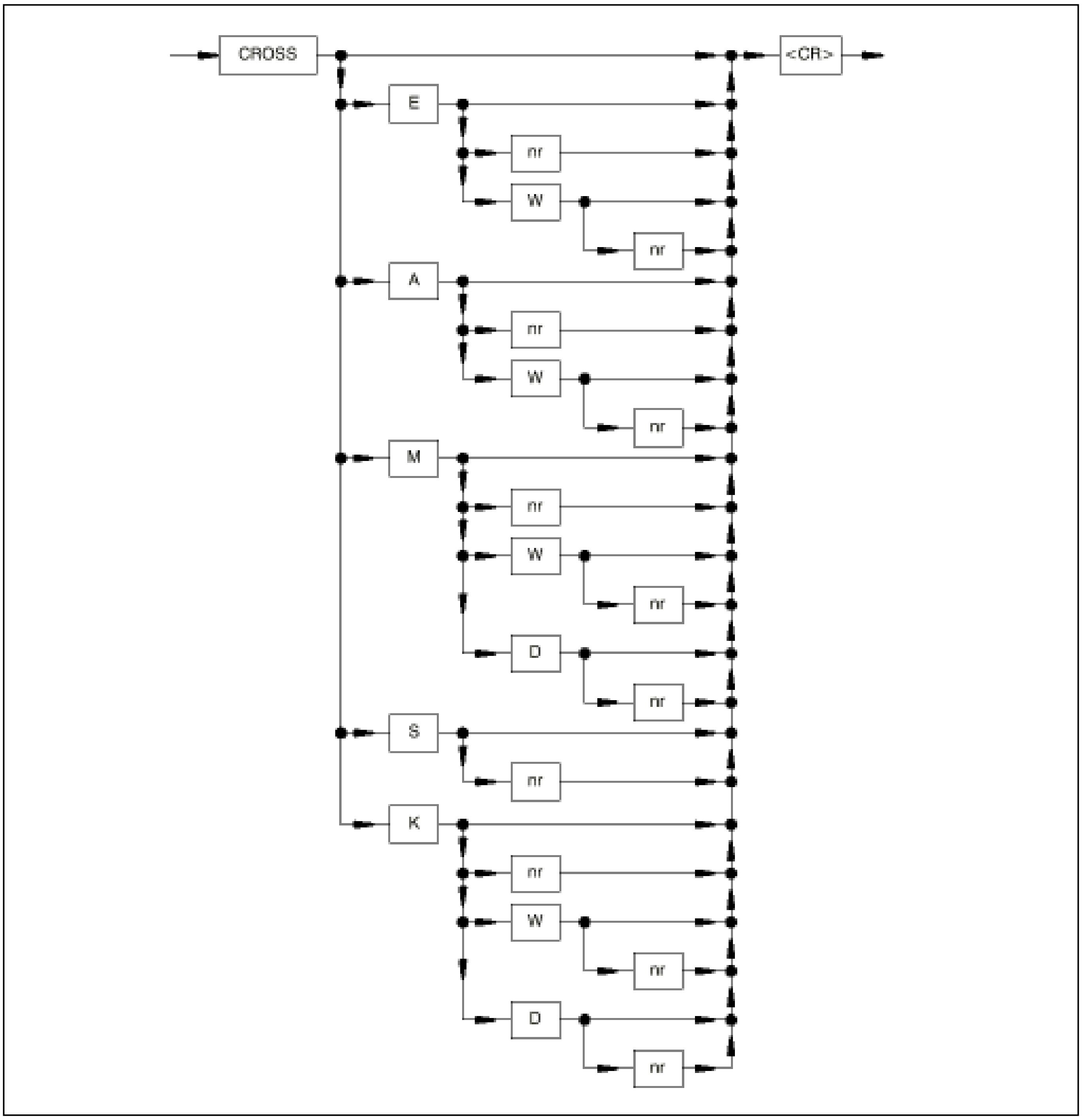
The processor capacity which corresponds to the difference between 100 % and the capacity utilization display is available for operation of the serial interfaces, i.e. for communication with the devices connected to the serial interfaces. The utilization should not be greater than 95 % for the longest program path so that communication is still possible via the serial interfaces. Note that the capacity utilization of the PLC is also determined by the current program branches (conditional jumps and consecutive number blocks).

Note:

The capacity utlization display indicates the correct utilization caused by the user program only if at the moment of display no communication is occurring via the serial interfaces.

Display cross reference list

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 variable
D:	Abbreviation for double-word variable
nr:	Number of the operand

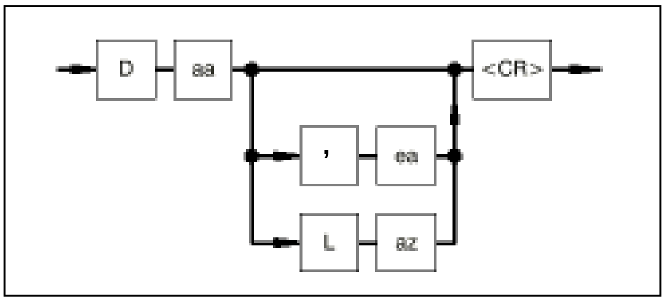
Function:

The cross reference list is the assignment of operands to the program memory addresses at which they occur. The cross reference list can be put out for

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

Display program

Command:



aa:	Start address as of which the program is to be displayed
ea:	End address of the program part to be displayed
L:	Length (keyword)
az:	Number of program memory words to be displayed

Function:
The specified program part is displayed.

Example:

- D 0,20 <CR>

The user program is displayed from address 0 through to address 20 on the monitor.

- D 10 L 20 <CR>

20 program memory words are displayed, starting from address 10.

Display format in the case of sentences:

```
start address  operator  operand
                    :
                    :
```

Display format in the case of block calls:

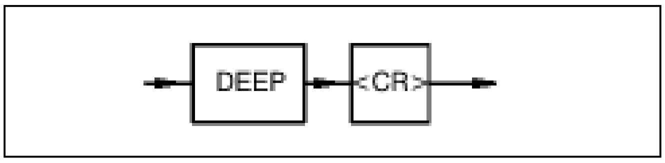
```
address n      !ba number
address n+1    type
address n+2    content of addr n+2
```

Example:

```
000000   !E 00,00
000002   &E 00,01
000004   =A 00,00
000006   !BA001
000007   AWT
000008   A 00,00
000009   KW 00,00
000010   KW 00,01
000011   AW 00,00
```

Erase PLC program on Flash EPROM

Command:

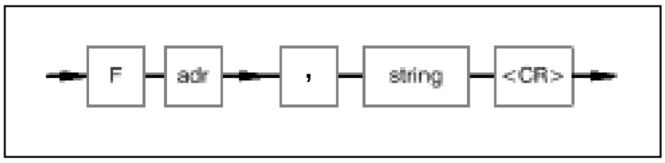


Function:

A PLC program stored on the Flash EPROM is erased (rendered invalid).

Search for string in the user program (Find)

Command:



adr:	Start address as of which searching is to be carried out. If no start address is entered, searching is performed as of address 0.
string:	Maximum 8 commands, i.e. 16 words of the intermediate code.

Function:

The user program memory is searched for the string entered by the user as of the entered start address through to the end of the user program memory. If the string is found, the address is displayed. If the string occurs several times in the program, the next program address which corresponds to the string is displayed in each case if you enter a semicolon (;).

Example:

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

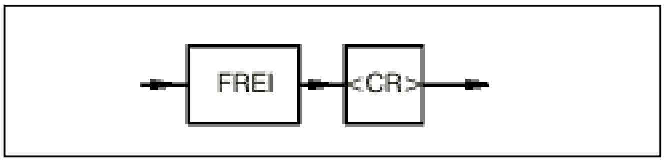
The entered string is sought as of the program memory start address 0.

F 100, !BA1 <CR>

Block call 1 is sought as of the program memory start address 100.

Enable a program change on a running PLC program

Command:



Function:

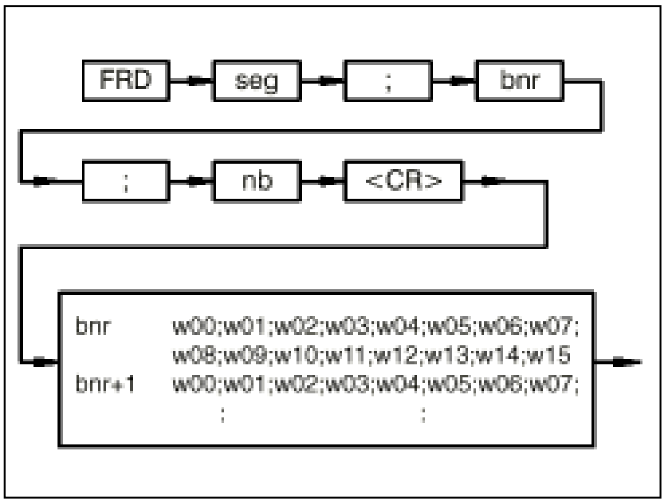
The modifications on a running PLC program performed after entry of command AEND are enabled for execution.

Before entry of command FREI, the performed program modifications have not been executed by the PLC yet.

After entry of command FREI, the performed modifications are executed by the PLC. Command ALT can be used to reactivate the old program status. The functionality of the PLC program can be further modified by a new program change.

Read data records from the Flash EPROM

Command:


FRD
seg
;
bnr
;
nb
<CR>
bnr w00;w01;w02;w03;w04;w05;w06;w07;
w08;w09;w10;w11;w12;w13;w14;w15
bnr+1 w00;w01;w02;w03;w04;w05;w06;w07;

seg:	Number of the data segment in the Flash EPROM valid values: 0...3
bnr:	Number of the block in the data segment, valid values: 0...480
nb:	Number of blocks, valid values: 1...481
;:	The individual values of the command must be separated by a semicolon.
bnr:	Number of the block in the data segment
w00:	1. word value of the block
:	: :
w15:	16. word value of the block
;:	The individual values of the answer are separated by a semicolon

Function:

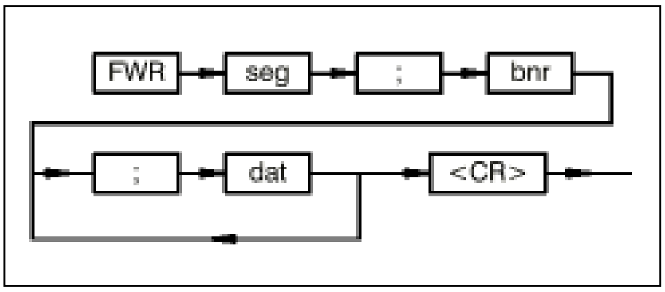
The user can read out data records from the Flash EPROM. The data are stored in blocks (16 words) in the Flash EPROM. The data of each block is safeguarded with a checksum. If a checksum error is detected when a block is read out, "ERROR" is put out instead of the number of the block (bnr). The checksum error is simultaneously entered into the respective data field as an FK3 error (Error indentification: 131 (83_H), Detailed Info: Offset, Segment).

After switching on the voltage, a checksum test of the entire Flash EPROM is performed. If a checksum error is detected, the FK3 error with the error identification 131 is displayed on the monitor and entered into the corresponding error flag.

Note: When a PLC program is started, the FK3 error flag (binary flag M 255,13) is always erased. The details (error identification, detailed information) are kept in the word flag data field (MW 255,00...MW 255,07).

Writing data records on the Flash EPROM

Command:



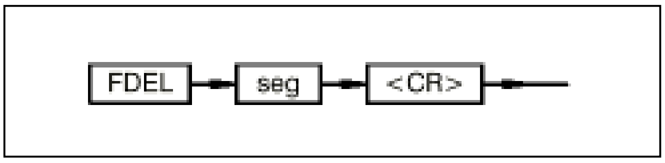
seg:	Number of the data segment in the Flash EPROM valid values: 0...3
bnr:	Number of the block in the data segment valid values: 0...480
;:	The individual values of the command must be separated by semicolons.
dat:	new value
;:	The individual values are separated by semicolons.

Function:

The user can write data records into the Flash EPROM. The data are entered as decimals (–32768...+32767). The data are always stored in blocks in the Flash EPROM and safeguarded with a checksum. Each block can store 16 words. If less than 16 word values are entered, the rest of the words are filled with the value zero. After 8 word values are entered, a <CR><LF> and 2 blanks are displayed on the monitor.

Erase data segment on the Flash EPROM

Command:



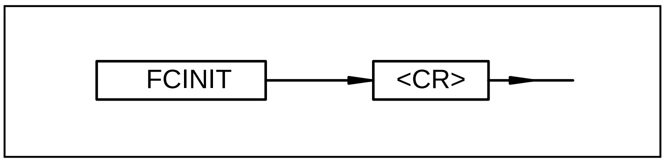
seg:	Number of the data segment in the Flash EPROM valid values: 0...3

Function:

The user can erase a data segment in the Flash EPROM. During erasing all data in this data segment are lost.

Initialize SmartMedia Card

Command:



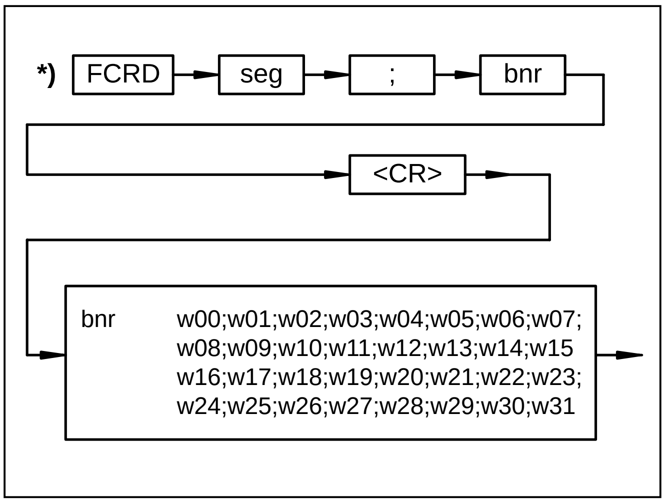
Function:

The SmartMedia Card is initialized as a user data card. Only on **initalized** cards data can be written.

When the SmartMedia Card is initialized, all previous data is deleted. A card, initialized for user **data**, can no more be used for storing user **programs**.

Read data records from the SmartMedia Card

Command:


*)
FCRD
seg
;
bnr
<CR>
bnr
w00;w01;w02;w03;w04;w05;w06;w07;
w08;w09;w10;w11;w12;w13;w14;w15
w16;w17;w18;w19;w20;w21;w22;w23;
w24;w25;w26;w27;w28;w29;w30;w31

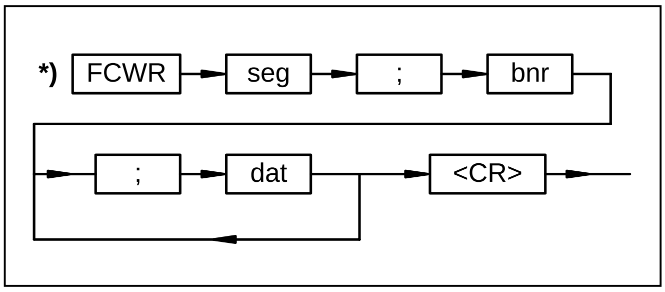
seg:	Number of the data segment in the SmartMedia Card valid values: 1...250
bnr:	Number of the block in the data segment, valid values: 0...127
;:	The individual values of the command must be separated by a semicolon.
bnr:	Number of the block in the data segment
w00:	1st word value of the block
:	: :
w31:	32nd word value of the block
;:	The individual values of the answer are separated by a semicolon

Function:

The user can read out data records from the SmartMedia Card. The data are stored in blocks (32 words) in the SmartMedia Card. The data of each block is safeguarded with a checksum.

Writing data records to the SmartMedia Card

Command:



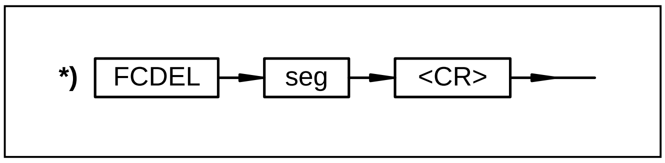
seg:	Number of the data segment in the SmartMedia Card valid values: 1...250
bnr:	Number of the block in the data segment valid values: 0...127
;:	The individual values of the command must be separated by semicolons.
dat:	new value
;:	The individual values are separated by semicolons.

Function:

The user can write data records into the Smart Media Card. The data are entered as decimals (–32768...+32767). The data are always stored in blocks in the SmartMedia Card and safeguarded with a checksum. Each block can store 32 words. If less than 32 word values are entered, the rest of the words are filled with the value zero. After 8 word values are entered, a <CR><LF> and 2 blanks are displayed on the monitor. A block only can be written once to the SmartMedia Card. Before rewriting the block, the segment has to be deleted.

Delete data segment on the SmartMedia Card

Command:



seg:	Number of the data segment in the SmartMedia Card valid values: 1...250

Function:

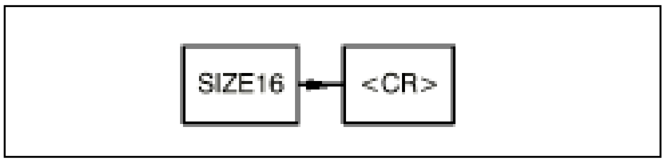
The user can delete a data segment in the Smart Media Card. During deleting all data in this data segment is lost.

---

*) A block only can be written once to the SmartMedia Card. Before rewriting the block, the segment has to be deleted.

Double user program memory

Command:



SIZE16 is available only for the basic units 07 KR 91, 07 KT 92 R202/R262 and 07 KT 93 R101/R171. SIZE 16 is no longer required for the other basic units because of their larger memories.

Function:

The user program memory is doubled (to 15296 instructions). After this command is entered it is no longer possible to change a program during a running PLC program.

The command can be entered only under the following conditions:

– no error of error class 2 present **and**

– PLC in the status "ABORTED"

and

– invalid user programm (DEEP command) on the Flash EPROM

After this command is entered, the SP command must be executed (save the user program on the Flash EPROM). This way the double program is stored and cannot be lost during a voltage shutdown.

If the SP command is not executed, the doubling of the program memory is cancelled when the voltage is switched OFF/ON and/or WARM command or COLD command are given.

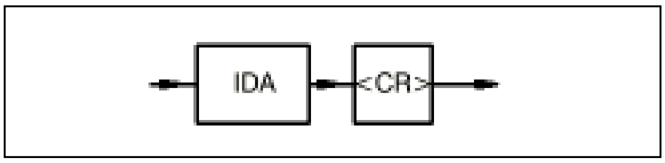
The doubling of the program memory is reversed as follows:

– Execute DEEP command **and**

– Voltage OFF/ON, WARM command or COLD command

Display program identification

Command:

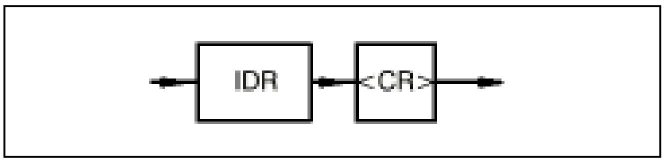


Function:

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

Delete program identification

Command:

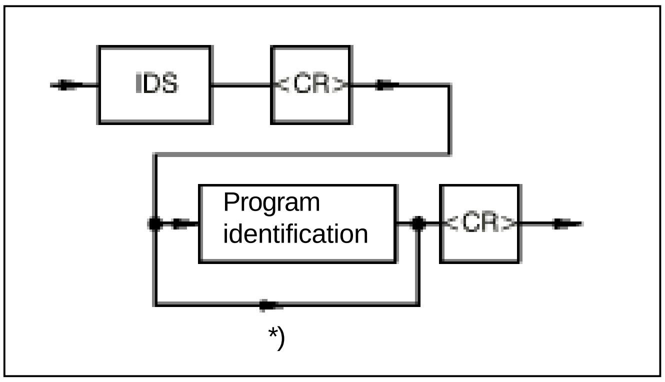


Function:

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

Enter program identification

Command:



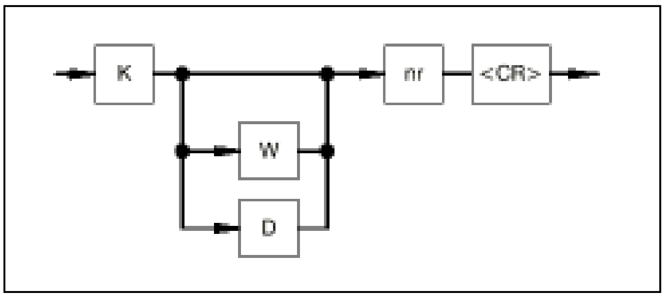
Program identification: These characters are assigned as the identification to the user program.

Function:

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

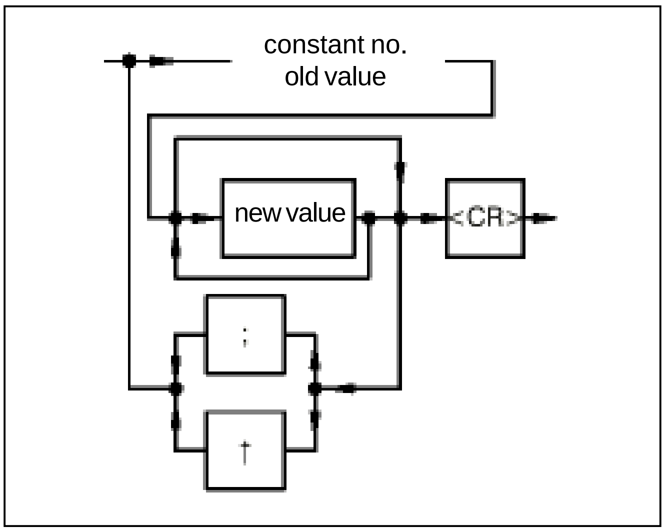
Enter/Edit values of indirect constants

Command:



W:	Abbreviation for word constants
D:	Abbreviation for double-word constants
nr:	Entered number of the constant

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



constant no. old value:
Displayed number and value of the constant.

new value:
The user can overwrite the value of the displayed constant by a new value. In the case of the word and double-word constants, a hexadecimal value may also be entered in place of a decimal value. An H is prefixed to the numerical value for this purpose.

Caution: Values H8000 and H8000 0000 are forbidden in two's-complement arithmetic (practical only in the case of masks for instance).

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

↑: Entering character "↑" results in the display of number and value of the constant with the next number down. If the character "↑" is entered without entering a new value, the old value of the displayed constant is retained. (Use character " ^ "on the PC keyboard.)

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

Function:

The required numerical values are assigned to the indirect constants.

This value assignment can also be performed with the user program running. This means that time values of timers can be modified when the system is running for instance.

Cycle time:

The cycle time is set with the double word constant KD 00,00. The set cycle time must be an integer multiple of the basic time of 1 ms, i.e. 1 ms, 7 ms, 23 ms etc.

Example:

K 0,0 <CR>

Output of the number and value of the binary constant K 00,00. This value can be overwritten if required. If a semicolon is entered, the number and value of the next binary constant (K 00,01) is output.

KW 0,4 <CR>

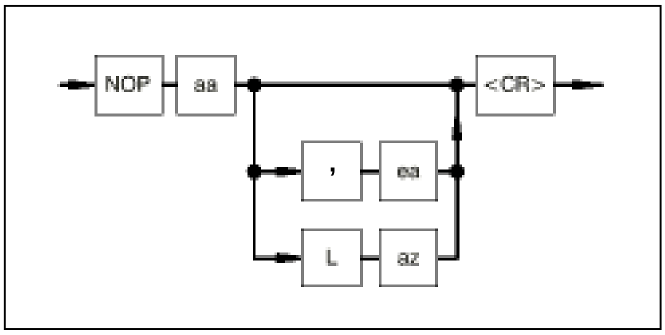
Output of the number and value of the word constant KW 00,04.

KD 0,0 <CR>

Output of the number and value of the double-word constant KD 00,00. The cycle time is preset with this constant.

Delete program part, i.e. overwrite with NOPs

Command:



aa:	Start address of the program part to be deleted
ea:	End address of the program part to be deleted
L:	Length (keyword)
az:	Number of program memory words to be deleted

Function:

The specified program part is deleted. Before deletion, a prompt is displayed in order to establish whether you really do want to delete this program part. The user must once again either confirm deletion with "Y" or cancel deletion with "N".

Example:

NOP 0,20 <CR>

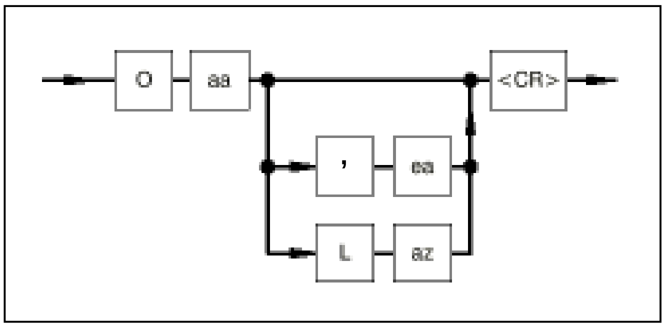
The user program is deleted from address 0 through to address 20.

NOP 10 L 20 <CR>

20 program memory words are deleted, as of address 10.

Optimize program

Command:



aa:	Start address of the area as of which the program memory is to be optimized
ea:	End address of the area
L:	Length (keyword)
az:	Number of program memory words

Function:

All NOPs are removed from the given program part and thus the program is compressed.

Example:

O 0 <CR>

The entire program memory is optimized.

O 0,10 <CR>

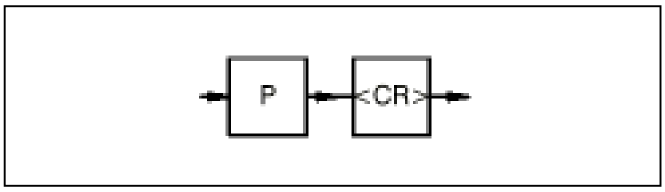
The program memory is optimized as of address 0 through to address 10.

O 10 L 10 < CR>

The NOPs within the next 10 program memory words as of address 10 are removed and the program is compressed accordingly.

Display free program memory

Command:

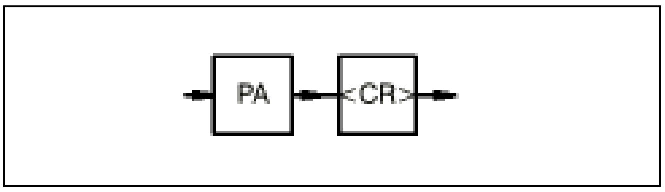


Function:

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

Prepare user program

Command:

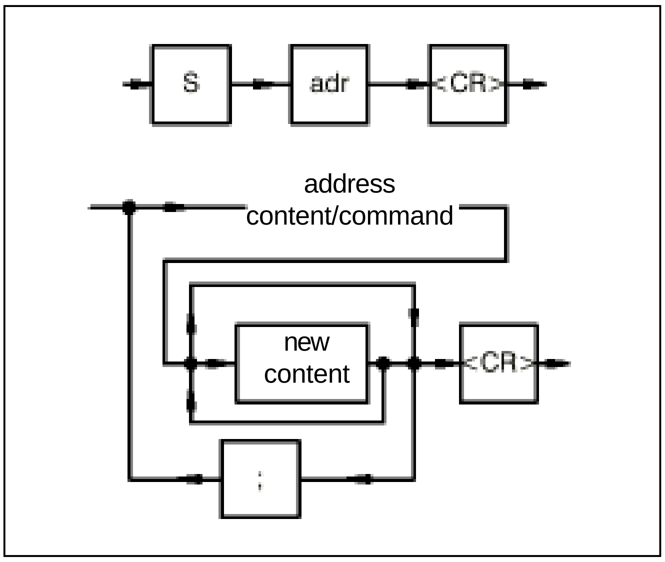


Function:

The I/O signals planned in the user program are enabled in the I/O configuration list of the PLC. In addition, a syntax check is carried out for the user program. In the case of sentences with relational operators using bracketed expressions, the RIGHT BRACKET in front of the binary assignment is stored by the translator as a binary RIGHT BRACKET in the intermediate code. This binary RIGHT BRACKET is corrected to form a word bracket by program preparation. PA computes the target addresses and the historical values to be skipped for the branch blocks and consecutive number blocks. The PA command is called automatically each time the program is started (G command).

Enter/Edit user program (substitute)

Command:



adr:	Program memory address as of which the program is to be entered or modified in instruction list.
address:	The program memory address whose content is to be modified is displayed by the PLC.
content:	Applies to block calls only. The content of the program memory address, translated back, is displayed.

command: Applies to sentences and the block header (number and type). The command or block header, translated back, is displayed, always as an entire command, i.e. operand and operator or block call and block type. If an address which does not point to the start of a command or to a block call is entered, this is corrected to the start of the command by the PLC.

new content: New content of the user program.

;: Entering a semicolon displays the subsequent program memory address and its content, and this can be modified if required. If no new "content" is entered before the semicolon, the old content of the displayed program memory address remains unchanged.

Function:

Entering or modifying the PLC program in instruction list. A program memory word is selected and displayed on the monitor as an instruction or operand. The displayed content can then be overwritten.

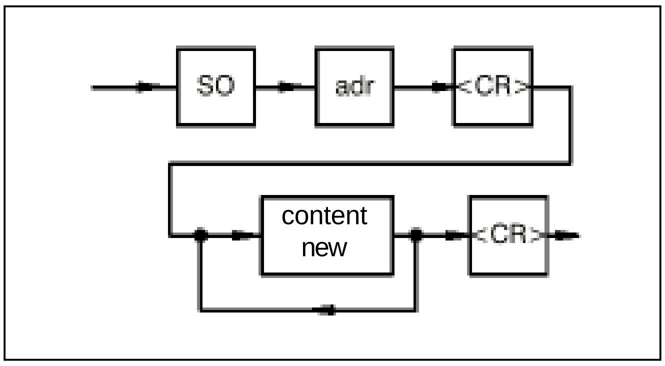
Note:

You will also find the following information for entering/modifying the instruction list with this command at the end of this Appendix:

- Syntactic structure of the instruction list.
- Instructions on how texts for function blocks DRUCK/EMAS are entered and displayed.

Enter/Edit user program without echo

Command:



adr:	Program memory address as of which the program is to be entered or modified

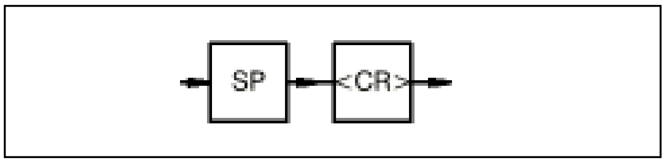
content new: New content of the user program

Function:

The program memory address as of which the program is to be entered is preset. The program can then be entered consecutively. The PLC returns no echo of the entered program. However, in the event of an error, the PLC returns an error message (e.g. Incorrect Entry).

Save PLC program in Flash EPROM and in the SmartMedia Card

Command:

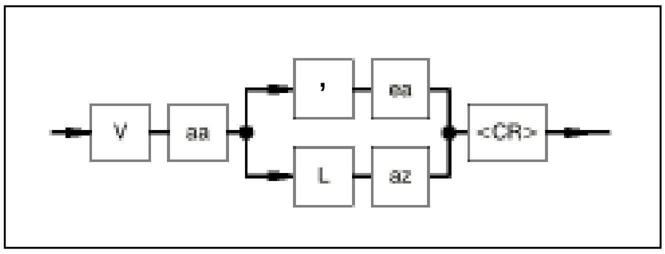


Function:

The PLC program is transferred from the RAM to the Flash EPROM and, if existing, also to the SmartMedia Card. Character <*> is displayed on the monitor at intervals of approximately 1 second during programming.

Move user program

Command:



aa:	Start address of program part to be moved
ea:	End address
L:	Length (keyword)
az:	Number of program memory words by which the program part is to be moved

Function:

The program is moved from address aa to address ea or from address aa by the specified number of program memory words. The gap which results is filled with NOPs. New program parts can be inserted in this gap. Moving is possible only if the required space is still available at the end of the user program. However, this is checked automatically.

Example:

V 0,10 <CR>

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

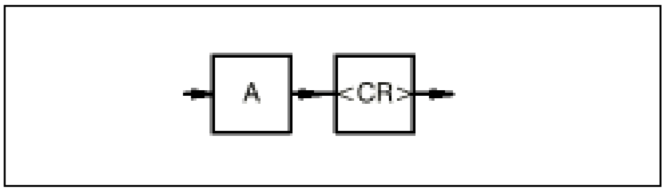
V 10 L 20 <CR>

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

2.4.1 Commands to test the user program

Abort user program

Command:



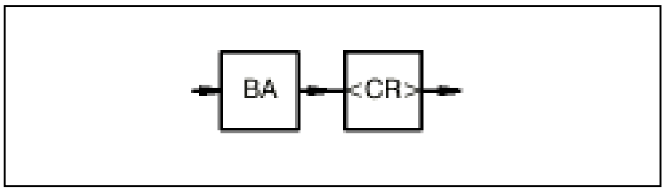
Function:

Execution of the user program is aborted. All outputs (binary and word) are set to zero. The user program can be restarted by entering "G".

Timers which have been started continue to run independently of the program status in the operating system. They are aborted only by a cold-start or power OFF/ON.

Display break points

Command:

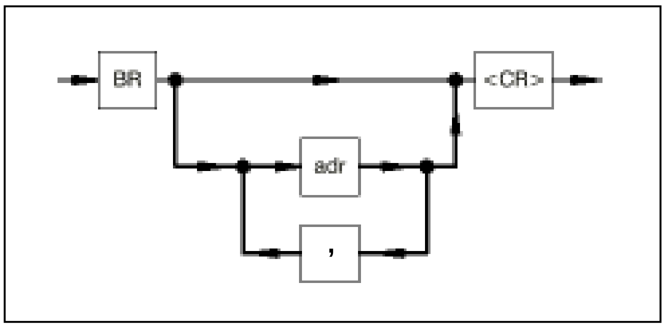


Function:

All breakpoints of the program are displayed. The address of the start of the command and its content are displayed and not the breakpoint address when the command is issued.

Reset break points

Command:



adr: Address of the break point to be deleted

,: If only certain break points are deleted, the individual addresses must be separated by a comma when entering.

Function:

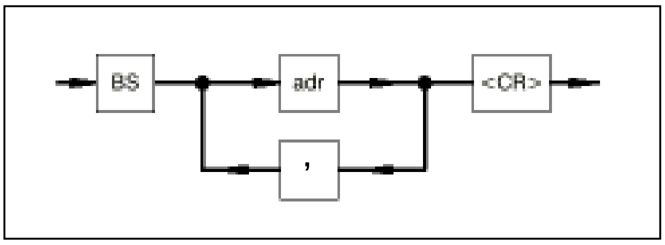
The breakpoints can be individually deleted. The command

BR <CR>

deletes all of the breakpoints of the program.

Set breakpoints

Command:



adr: Address of the breakpoint

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

Breakpoints can be set:

- to the address of the operand after an assignment character
- to the address of a RIGHT BRACKET
- to the address of the last parameter of a block
- to the address of the end of the program

Function:

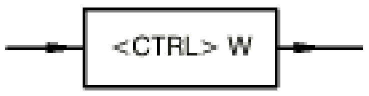
After the program start, the program stops at the first breakpoint. Breakpoints may also be entered with the program running. A maximum of 15 breakpoints may be set.

Advancing to the next breakpoint: If a semicolon is entered, the program runs to the next breakpoint after expiry of the cycle time and displays the program address and the command at this address. If the next breakpoint is not reached after a specific period, owing to a long cycle time, the display operation can be aborted by entering <CTRL>C if required.

If a breakpoint is set to a program point which is not executed, e.g. owing to a jump, the program continues its cycles but with four times the cycle time, which may have a disadvantageous effect on the functionality.

Change over between operator control functions <–––> Monitor

Command:

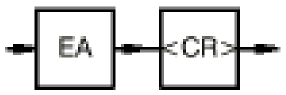


Function:

Pressing key <CTRL> and key W simultaneously takes you to the monitor program of the PLC. This makes available certain basic functions at the monitor level to the user. If you are in the monitor program, you can switch back to the operating program of the PLC by entering <CTRL> and W again.

I/O-Test

Command:



Function:

This mode permits the user to check the wiring of his I/O signals from the PLC through to the process in order to ensure that the wiring is correct.

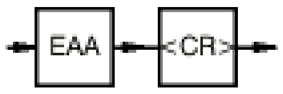
After starting the user program, it is not executed. Only the I/O signals planned in the program are operated, i.e. the input signals are read in and the output signals are brought out.

By actuating limit switches etc., it is possible to check whether the signals arrive under the declared designation in the PLC. By setting outputs in targeted manner, it is possible to check whether the signals arrive at the correct point in the process. Command Z or ZD can be used to display the required I/O variables in the PLC.

Command "EA" can also be entered with the program running. In this case, the mode does not take effect until the start of the next program cycle.

Deactivate I/O test mode

Command:

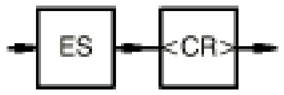


Function:

Mode I/O test is deactivated with this command, i.e. the user program continues to run normally as of this point. It is advisable to abort the program before deactivating the I/O test.

Activate single step mode

Command:



Function:

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

Command Z can be used to display variable values.

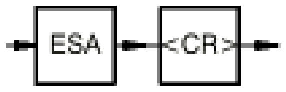
Command "ES" can also be entered with the program running. In this case, the mode does not take effect until the start of the next program cycle.

Advancing by one step:

If you enter a semicolon, the program runs to the next breakpoint after expiry of the cycle time and displays the program address and the command at this address. If the next breakpoint is not reached after a specific period, owing to a long cycle time, the display operation can be aborted by entering <CTRL>C if required.

Deactivate single step mode

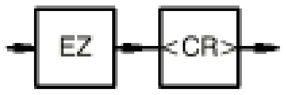
Command:



Single-step mode is deactivated, i.e. the user program continues to run normally as of the current breakpoint.

Activate single step mode

Command:



Function:

When the program is started, the program stops at the end of the program. Command "EZ" can also be entered with the program running.

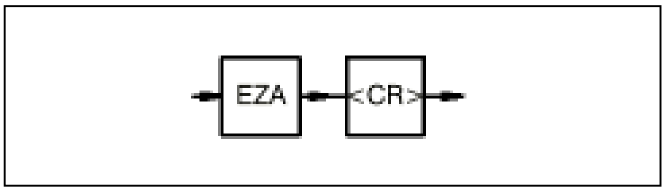
The mode does not come into effect until the start of the next program cycle.

Advancing by one program cycle:

If a semicolon is entered, the program is run through once after expiry of the cycle time and displays the program address and the command at this address (!PE). If the next breakpoint is not reached after a specific period, owing to a long cycle time, the display operation can be aborted by entering <CTRL>C if required.

Deactivate single cycle mode:

Command:

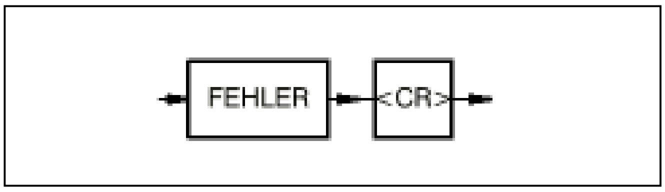


Function:

Single-cycle mode is deactivated, i.e. the program is executed normally again.

Display the contents of the error registers

Command:



Function:

The error information stored in the PLC is output.

Forcing I/O signals

On the PLC, the user can "force" input signals and output signals. This means that values are preset for I/O signals by the user. The PLC then operates with the force values instead of the real input signals. In turn, the PLC issues the force values to the output devices and not the output signals computed in the PLC program. The force values apply until forcing is cancelled for individual I/O signals or for all I/O signals. Both the values supplied by the input devices and the values assigned to outputs in the PLC program thus have no effect during forcing. Forcing can be applied both to binary I/O signals and to word I/O signals.

Maximum number of I/O signals to be forced:

• Digital inputs:	64
• Word inputs:	16
• Digital outputs:	64
• Word outputs:	16

Forcing is performed in the following way:

Forcing inputs

The PLC generates an image of the input signals planned in the PLC program at the start of each program cycle. If inputs are to be forced, their real values are replaced by the force values preset by the user after read-in. The PLC operates only with the modified input image during the program cycle, and, thus, signal changes on the input device during the program cycle are unimportant.

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 devices. If outputs are to be forced, their real values are replaced by the force values before they are output in the output image.

Behavior after power failure, RESET or warm start

After a power failure, the PLC has "forgotten" the force job. The list of I/O signals to be forced, entered before the power failure, is, however, still present in the PLC and can also be displayed with command FORC A <CR>. The overall force list is reactivated and forcing is placed back into effect by entering a single signal to be forced.

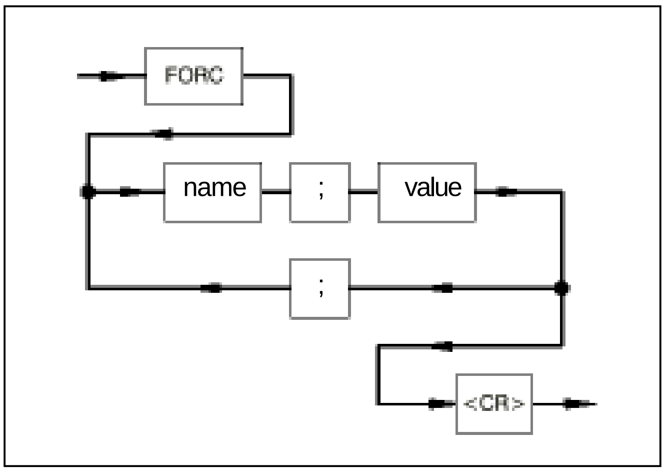
The following commands are available for forcing I/O signals:

• 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 with the command FORC.

Command:



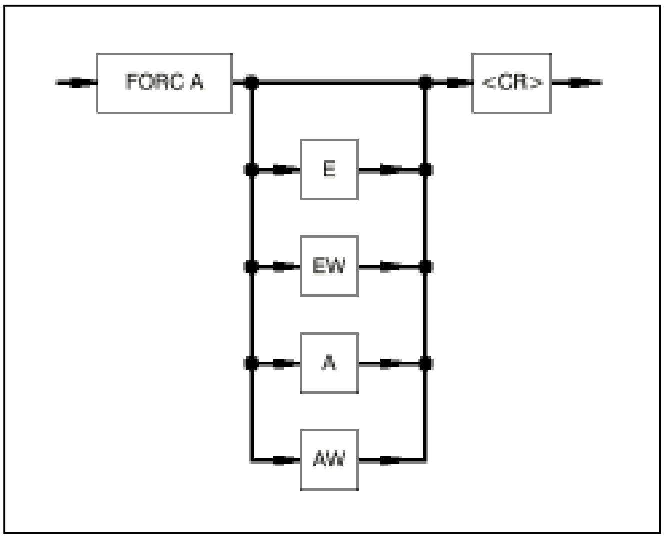
name: Name of the input or output signal to be forced

value: Force value for the input or output

;: A semicolon is used as the separator between the name and the force value. If several inputs/outputs are to be forced, they must also be separated by a semicolon.

Display force value

Command:

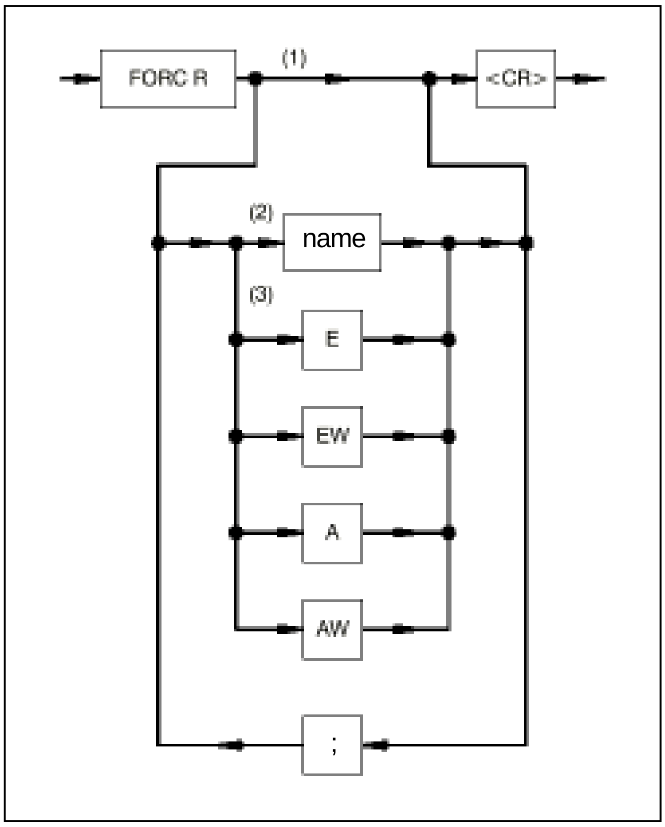


Function:

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

Delete forcing

Command:



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

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

Function:

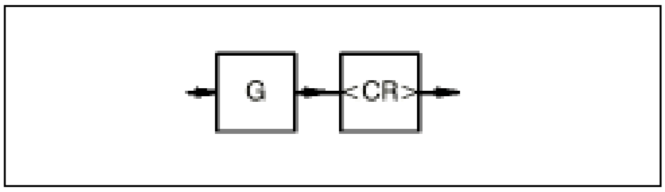
(1) Terminating forcing for all I/O signals

(2) Terminating forcing for single I/O signals

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

Start user program

Command:



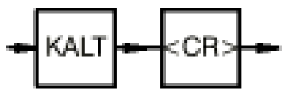
Function:

The user program is started and the operands are initialized.

The operand areas are initialized according to the corresponding system constant.

Perform cold start

Command:



Function:

The cold start command is only allowed when the PLC program is "aborted".

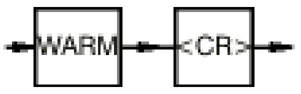
- All RAM memories are tested and deleted.
- If there is no user program in the Flash EPROM, the default values are set for all of the system constants (same as factory setting).
- If there is a user program in the Flash EPROM, it will be stored in the RAM together with the system constants.
- The operating modes defined by the system constants are set.
- The CS31 system bus is re-initialized (only in case of CS31 system bus master).

Performing a cold start

- Command KALT <CR> in terminal mode or
- voltage OFF/ON, when there is no battery or
- menu option "Cold start" in the programming system.

Perform warm start

Command:



Function:

The warm start command is only allowed when the PLC program is "aborted".

Warm start

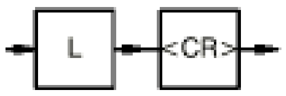
- All RAM memories are tested and deleted with the exception of the program memory and the operand memory (flags).
- If there is a user program in the Flash EPROM, it will be stored in the RAM together with the system constants.
- The operating modes defined by the system constants are set.
- The CS31 system bus is re-initialized (only in case of CS31 system bus master).

Performing a warm start

- Command WARM <CR> in terminal mode or
- voltage OFF/ON, when there is a battery or
- menu option "Enable PLC mode" in the programming system.

Continue user program

Command:



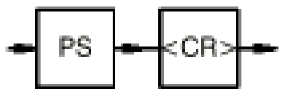
Function:

The user program is continued after a preceding stop ("W"). When continuing, the flags and internal statuses have the same value as with program stop.

Timers which have started continue to run independently of the program status in the operating system. They are aborted only by a cold-start or power OFF/ON.

Display program status

Command:

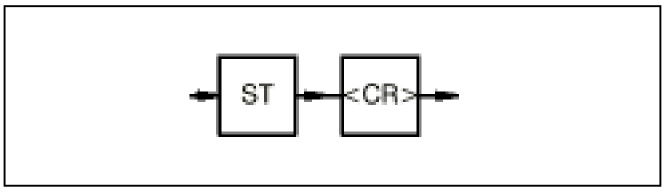


Function:

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

Display PLC status

Command:



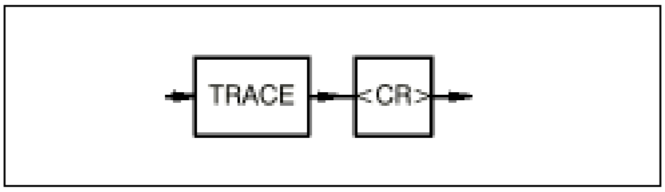
Function:

The entire PLC status is displayed as follows:

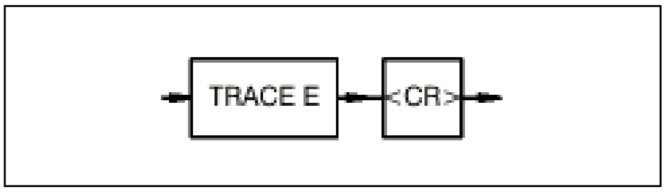
- Program identification
- Cycle time
- Program status
- Active test functions
- TRACE registers
- Error messages
- Capacity utilization

TRACE mode

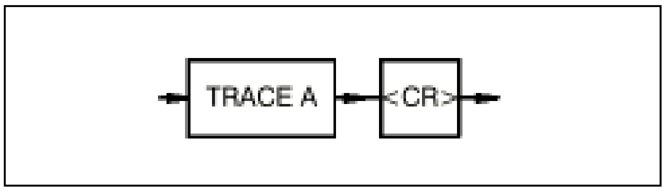
Command: Display TRACE memory



Command: Activate TRACE mode



Command: Deactivate TRACE mode

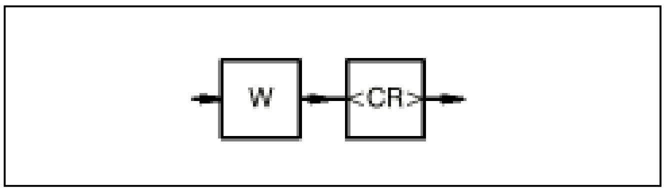


Function:

In TRACE mode, the PLC notes the address of the block last executed or the address of the instruction last executed. After a system crash, the operator is thus provided with information as to how far the user program has been executed. The contents of the TRACE memory are retained in the event of a RESET.

Stop user program

Command:



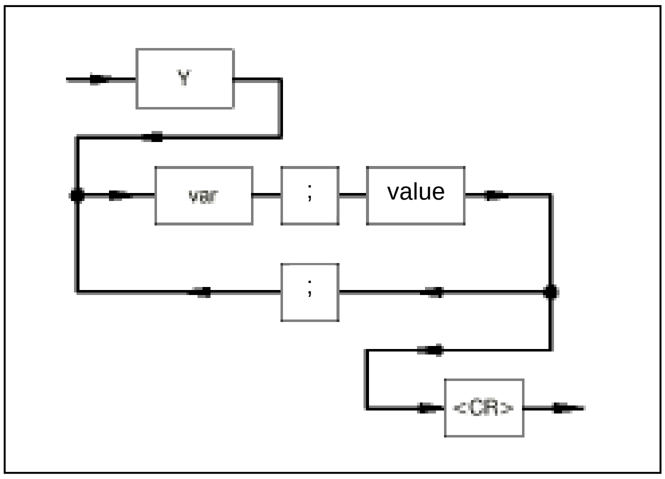
Function:

The user program is stopped.

The values of the outputs and of the flags are retained. Timers which have been started continue to run independently of the program status in the operating system. They are aborted only by a cold-start or power OFF/ON.

Overwrite value of a variable with value to be entered

Command:



var: Name of the variable or indirect constant

value: New value which is to be assigned to the variable

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

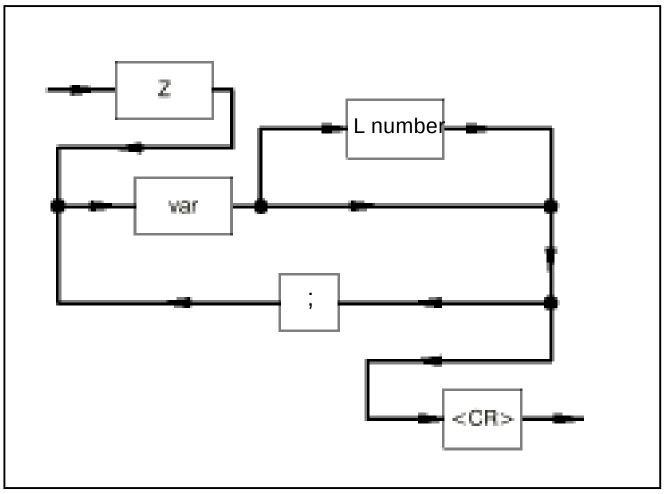
Note:

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

If an indirect constant is modified with this command, this modification is performed only in the operand memory and not in the program memory, i.e. this value is overwritten again by the value from the program memory with the next program start.

Display status of variables

Command:

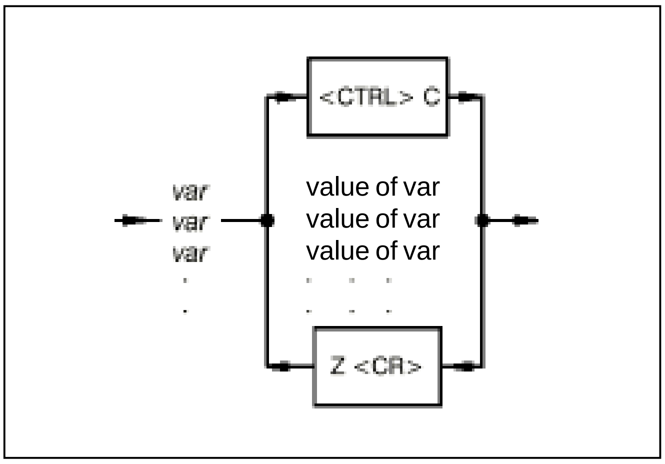


var: Variable (flag, step, input, output, indirect constant) to be displayed

;: The individual variables must be separated by semicolons.

L number: Number of consecutively numbered variables as of the variable var which are to be displayed.

Example: M 0,0 L 3
The following are displayed:
M 0,0 M 0,1 M 0,2



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

Function:

The variable names preset by the user are displayed on the monitor. The value of this variable is updated each time the character Z <CR> is entered. The displayed variable values always originate from the same program cycle and represent a "snapshot" at the end of the cycle.

The number of variables to be displayed is restricted to 22 with this command since no more screen lines than this are available.

Computer connection instead of terminal

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

ZO: Number of possible variables maximum 120, otherwise as for command Z.

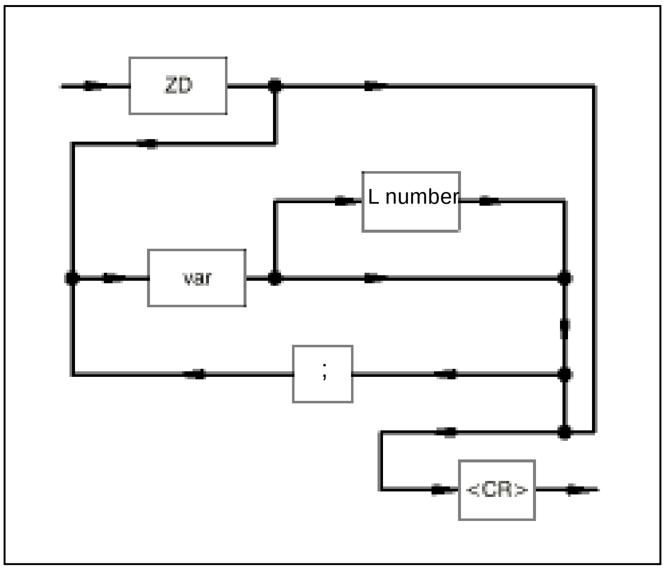
Screen control: In the case of commands Z, ZO and ZD, the following control characters are used by the PLC for screen control:

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

ZZ: Number of possible variables maximum 120. The PLC sends no ESC sequences to the screen controller, but only the variable values, each followed by a <CR>. The variable values have the same order as the preset variable list, otherwise as with command Z.

Display and continually update status of variables

Command:

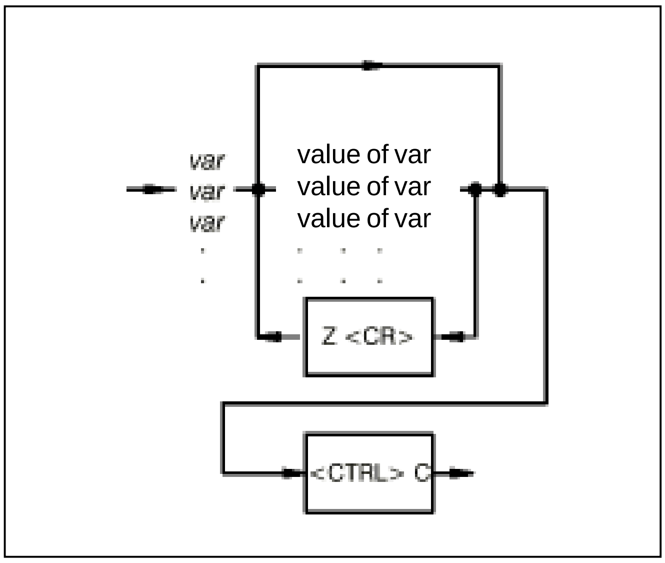


var: Variable (flag, input, output, indirect constant) to be displayed

;: The individual variables must be separated by semicolons.

L number: Number of consecutively numbered variables as of the variable var which are to be displayed.

Example: M 0,0 L 3
The following are displayed:
M 0,0 M 0,1 M 0,2



Function:

The variable names preset by the user are displayed on the monitor. The related variable values are updated automatically. The displayed variable values always originate from the same program cycle and represent a "snapshot" at the end of the cycle.

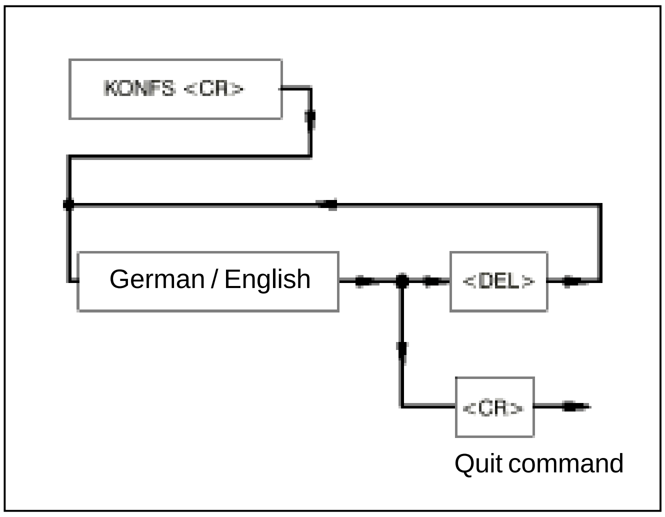
The maximum number is 22. The command is terminated by a <CTRL>C.

If character Z <CR> or ZD <CR> is then entered, the status display is reactivated for the previously entered variables.

2.4.3 Commands for configurating

Display/change operating modes

Command:



Function:

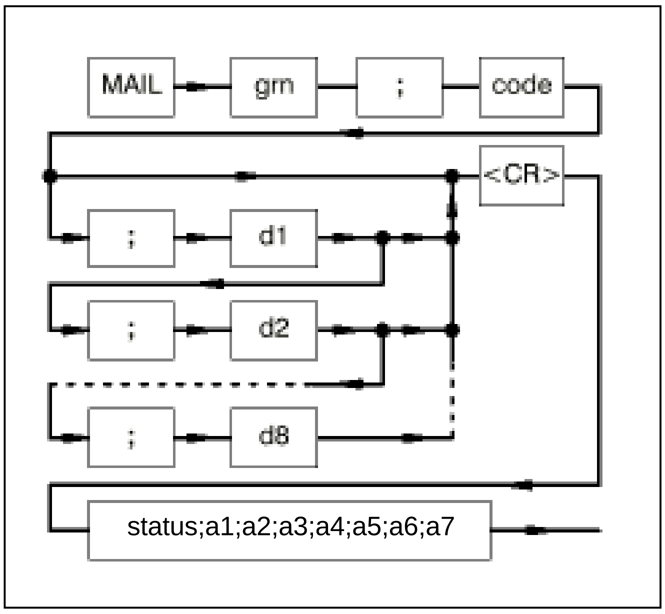
After command KONFS <CR> is entered, the set language is displayed on the monitor. If you press key <DELETE> (<CTRL> and the backspace key on PCs), the language is switched over. The command is terminated by entering a <CR>.

Note:

The DELETE key is frequently not available on personal computers. The key code ($7F_H$) of the DELETE key can be generated on such keyboards by pressing two keys. In general, these keys are keys <CTRL> and the backspace key.

Configuration/interrogation of the configuration of CS31 remote module (07 KR 31, 07 KR 91, 07 KT 92, 07 KT 93, 07 KT 94)

Command:



grn:	Group number with which the remote module is addressed by the PLC program
code:	Job code
d1:	1st data byte of the job
:	: :
d8:	8th data byte of the job
;:	The individual values of the job must be separated by semicolons.
status:	Status of the response: 51 (OK response) 170 (Not OK response)
a1:	1st data byte of the response
:	: :
a7:	7th data byte of the response
;:	The individual values of the job are separated by semicolons.

Function:

The user has the option of configuring CS31 remote modules and interrogating the set configuration. The jobs are handled internally via a transmit mailbox (job) and receive mailbox (response).

List of jobs:

The OK responses are described for the respective jobs. The not OK responses always appear as follows:

- **Not OK response**
  status: 170

a1:	1	=	Unknown job code
	2	=	Invalid parameter, e.g. group number
	3	=	Remote module does not respond
	10	=	Mail Box is not free within 3 sec.
	11	=	Job aborted by activation of the RUN/STOP switch
	12	=	Job is not fetched within 6 sec.
	13	=	No reply within 6 sec.

a2...a7: 0

- **Updating the maximum number of remote modules detected**

The contents of the input word EW 07,15 include the maximum number of remote modules detected in the past. The current actual number of existing remote modules may be less than this.

This command updates this value. The existing modules are counted and the value is stored.

The user can interrogate this value in the PLC program (EW 07,15, bit 8...15).

– **Job**

grn:	255 (master PLC with bus)
code:	132
d1...d8:	not used

– **OK response**

status:	51
a1...a7:	0

- **Interrogation whether open-circuit monitoring is activated or deactivated for an input**

– **Job**

grn:	Group number 0...63
code:	32
d1:	Channel number 0...15
d2...d8:	not used

– **OK response**

status:	51
a1:	47 = Open-circuit monitoring ON
	32 = Open-circuit monitoring OFF
a2...a7:	0

- **Interrogation whether open-circuit monitoring is activated or deactivated for an output**

– **Job**

grn:	Group number 0...63
code:	33
d1:	Channel number 0...15
d2...d8:	not used

– **OK response**

status:	51
a1:	47 = Open-circuit monitoring ON
	32 = Open-circuit monitoring OFF
a2...a7:	0

- **Activating or deactiving open-circuit monitoring of an input**

– **Job**

grn:	Group number 0...63
code:	224 = Open-circuit monitoring ON
	160 = Open-circuit monitoring OFF
d1:	Channel number 0...15
d2...d8:	not used

– **OK response**

status:	51
a1...a7:	0

- **Activating or deactivating open-circuit monitoring of an output**

– **Job**

grn:	Group number 0...63
code:	225 = Open-circuit monitoring ON
	161 = Open-circuit monitoring OFF
d1:	Channel number 0...15
d2...d8:	not used

– **OK response**

status:	51
a1...a7:	0

- **Interrogation whether a channel is configured as an input or as an input/output**

– **Job**

grn:	Group number 0...63
code:	34
d1:	Channel number 0...15
d2...d8:	not used

– **OK response**

status:	51
a1:	34 = Input
	35 = Input/Output
a2...a7:	0

- **Configuration of a channel as an input or input/output**

– **Job**

grn:	Group number 0...63
code:	162 = Input
	163 = Input/Output
d1:	Channel number 0..15
d2...d8:	not used

– **OK response**

status:	51
a1...a7:	0

- **Interrogation of the input delay of a channel**

– **Job**

grn:	Group number 0...63
code:	38
d1:	Channel number 0...15
d2...d8:	not used

– **OK response**

status:	51
a1:	Input delay:
	2 = 2 ms
	4 = 4 ms
	.
	.
	.
	30 = 30 ms
	32 = 32 ms
a2...a7:	0

- **Setting the input delay of a channel**

  – **Job**

grn:	Group number 0...63
code:	166
d1:	Channel number 0...15
d2:	Input delay 2 = 2 ms 4 = 4 ms . . . 30 = 30 ms 32 = 32 ms
d3...d8:	not used

  – **OK response**

status:	51
a1...a7:	0

- **Acknowledge error on remote module**

  This command resets the error messages on the selected remote module. The error messages can only be reset when the cause of error has been remedied.

  – **Job**

grn:	Group number 0...63
code:	232
d1:	smallest chan. number on the module: 0 = smallest channel number on the module is 0 (≤7) 8 = smallest channel number on the module is 8 (>7)
d2:	Module type: 0 = Digital input 1 = Analog input 2 = Digital output 3 = Analog output 4 = Digital input/output 5 = Analog input/output Note: Bit: even number (0, 2, 4) Word: odd number (1, 3, 5)
d3...d8:	not used

  – **OK response**

status:	51
a1...a7:	0

- **Acknowledge error on the remote module and reset configuration values to default setting**

  In addition to "Acknowledging error on remote module" all of the configurable settings are reset to default setting.

  – **Job**

grn:	Group number 0...63
code:	233
d1:	first channel number on the module: 0 = first channel number on the module is 0 (≤7) 8 = first channel number on the module is 8 (>7)
d2:	Module type: 0 = Digital input 1 = Analog input 2 = Digital output 3 = Analog output 4 = Digital input/output 5 = Analog input/output Note: Bit: even number (0, 2, 4) Wort: odd number (1, 3, 5)
d3...d8:	not used

  – **OK response**

status:	51
a1...a7:	0

- **Interrogation of the configuration of an analog input**

  – **Job**

grn:	Group number 0...63
code:	42
d1:	Channel number 0...15
d2...d8:	not used

  – **OK response**

status:	51
a1:	51 = Input ±10 V 50 = Input 0...20 mA 49 = Input 4...20 mA
a2...a7:	0

- **Interrogation of the configuration of an analog output**

  – **Job**

grn:	Group number 0...63
code:	43
d1:	Channel number 0...15
d2...d8:	not used

  – **OK response**

status:	51
a1:	50 = Output 0...20 mA 49 = Output 4...20 mA 51 = Output ±10V
a2...a7:	0

- **Configuration of an analog input**

  – **Job**

grn:	Group number 0...63
code:	170
d1:	Chanel number 0...15
d2:	51 = Input 10 V 50 = Input 0...20 mA 49 = Input 4...20 mA
d3...d8:	not used

  – **OK response**

status:	51
a1...a7:	0

- **Configuration of an analog output**

  – **Job**

grn:	Group number 0...63
code:	171
d1:	Channel number 0...15
d2:	50 = Output 0...20 mA 49 = Output 4...20 mA 51 = Output ±10V
d3...d8:	not used

  – **OK response**

status:	51
a1...a7:	0

- **Interrogation of the bus configuration**

The bus interface of the master PLC has a list which stores specific data of the remote modules. In this list, the remote modules are numbered in the order in which they are encountered on the CS31 bus. This command involves specifying the internal number of the modules. The response received is the group number stored under this number and the status information on the corresponding module.

  – **Job**

grn:	0 (is not evaluated)
code:	80
d1:	Number from the module list (1...31)
d2...d8:	not used

  – **OK response**

status:	51
a1:	Status of the remote module: Bit 0...3: Number of process data bytes (binary module) and/or words (word module) sent to the master by the module Bit 4...7: Number of process data bytes (binary module) and/or words (word module) sent to the module by the master
a2:	Group number (0...63)
a3:	Bit 0: 0 = smallest channel number ≤7 1 = smallest channel number >7 Bit 1: 0 = Binary module 1 = Word module
a4...a7:	0

- **Reading 1...6 bytes (07 KR 91, 07 KT 92, 07 KT 93 and 07 KT 94)**

  – **Job**

grn:	Group number 0...63
code:	49 = Read 1 byte 50 = Read 2 bytes 51 = Read 3 bytes 52 = Read 4 bytes 53 = Read 5 bytes 54 = Read 6 bytes
d1:	first channel number on the module: 0 = first channel number on the module is 0 (≤7) 8 = first channel number on the module is 8 (>7)
d2:	Module type: 0 = Digital input 1 = Analog input 2 = Digital output 3 = Analog output 4 = Digital input/output 5 = Analog input/output Note: Bit: even number (0, 2, 4) Word: odd number (1, 3, 5)
d3:	Byte start address (low byte)
d4:	Byte start address (high byte)
d5...d8:	not used

  – **OK response**

status:	51
a1:	Value of the 1st byte
a2:	Value of the 2nd byte or 0
a3:	Value of the 3rd byte or 0
a4:	Value of the 4th byte or 0
a5:	Value of the 5th byte or 0
a6:	Value of the 6th byte or 0
a7:	0

- **Reading 1 bit of a byte**

  – **Job**

grn:	Group number 0...63
code:	63
d1:	first channel number on the module: 0 = first channel number on the module is 0 (≤7) 8 = first channel number on the module is 8 (>7)
d2:	Module type: 0 = Digital input 1 = Analog input 2 = Digital output 3 = Analog output 4 = Digital input/output 5 = Analog input/output Note: Bit: even number (0, 2, 4) Word: odd number (1, 3, 5)

d3:	Byte start address (low byte)
d4:	Byte start address (high byte)
d5:	Bit position within the bytes 0...7
d6...d8:	not used

– **OK response**

status:	51
a1:	Bit value (0 or 1)
a2...a7:	0

- **Writing 1...4 bytes (07 KR 91, 07 KT 92, 07 KT 93 and 07 KT 94)**

  – **Job**

grn:	Group number 0...63
code:	65 = Write 1 byte 66 = Write 2 bytes 67 = Write 3 bytes 68 = Write 4 bytes
d1:	first channel number on the module: 0 = first channel number on the module is 0 (≤7) 8 = first channel number on the module is 8 (>7)
d2:	Module type: 0 = Digital input 1 = Analog input 2 = Digital output 3 = Analog output 4 = Digital input/output 5 = Analog input/output Note: Bit: even number (0, 2, 4) Word: odd number (1, 3, 5)
d3:	Byte start address (low byte)
d4:	Byte start address (high byte)
d5:	Value of the 1st byte
d6:	Value of the 2nd byte or not used
d7:	Value of the 3rd byte or not used
d8:	Value of the 4th byte or not used

  – **OK response**

status:	51
a1...a7:	0

- **Writing 1 bit of a byte**

  – **Job**

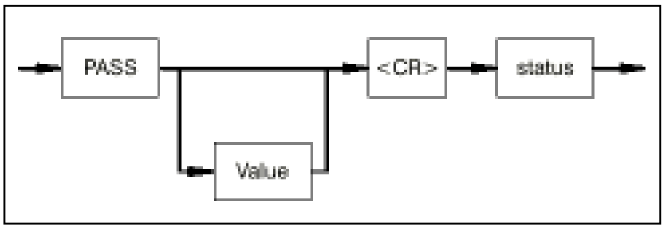
grn:	Group number 0...63
code:	79
d1:	first channel number on the module: 0 = first channel number on the module is 0 (≤7) 8 = first channel number on the module is 8 (>7)
d2:	Module type: 0 = Digital input 1 = Analog input 2 = Digital output 3 = Analog output 4 = Digital input/output 5 = Analog input/output Note: Bit: even number (0, 2, 4) Word: odd number (1, 3, 5)
d3:	Byte start address (low byte)
d4:	Byte start address (high byte)
d5:	Bit position within the byte 0...7
d6:	Bit value (0 or 1)
d7...d8:	not used

  – **OK response**

status:	51
a1...a7:	0

Password (only for series 30, 40, 50)

Command:



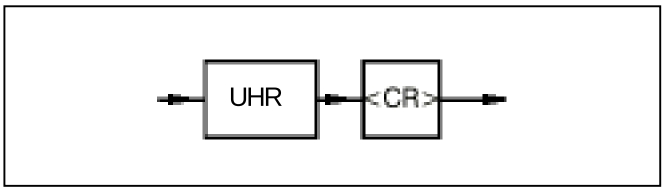
Function:

The command PASS activates and/or deactivates the password function. Any 4 digit hexadecimal number (except 0000) can be entered as a password. If the password word is activated, the following functions are disabled: AEND, D, DEEP, FREI, N, NOP, O, S, V.

Value:	Any 4 digit hexadecimal number. Attention: The value 0000 has not effect.
status:	The activation/deactivation of the password function is displayed.

Display of time and date (07 KR 91, 07 KT 92, 07 KT 93 and 07 KT 94)

Command:



Function:

The time and the date are displayed on the monitor in the following form:

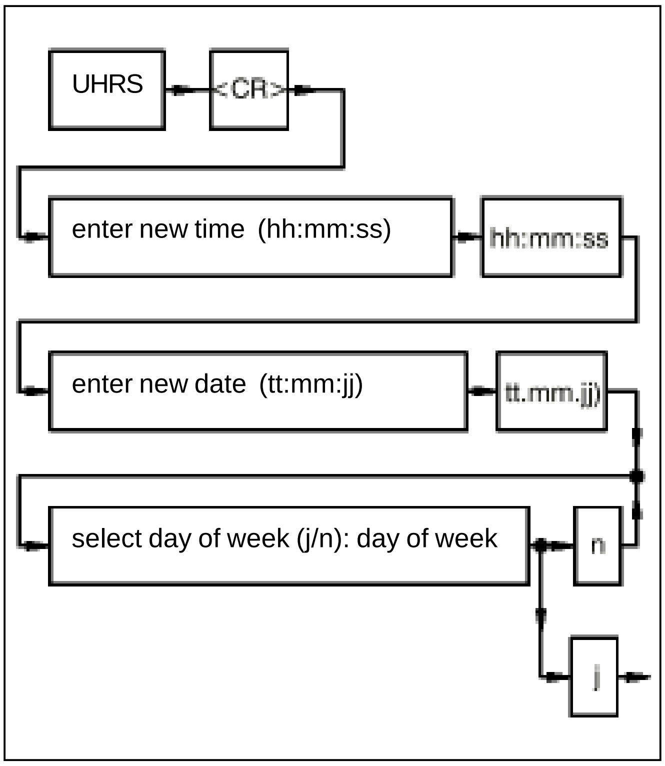
SYSTEM TIME : HH:MM:SS
SYSTEM DATE: DAY OF WEEK TT.MM.JJ

Where:

HH:	Hours
MM:	Minutes
SS:	Seconds
DAY OF WEEK:	Name of the day of the week
TT:	Day
MM:	Month
JJ:	Year

Setting time and date

Command:


UHRS
<CR>
enter new time (hh:mm:ss)
hh:mm:ss
enter new date (tt:mm:jj)
tt.mm.jj)
select day of week (j/n): day of week
n
j

Function:

Setting the time and date. For the day of the week, the clock possesses a number between 1 and 7 internally. When converting the number to the name, it assumes that Monday is the first day of the week (number 1 --> Monday). If the clock is set with block UHR (see also block catalog), a different number may then be assigned to Monday. In this case, the display of the day of the week no longer corresponds to the command UHR <CR> since the display function always assumes that Monday is assigned the number 1.

hh and/or hh:	Hours
mm and/or mm:	Minutes
ss and/or ss:	Seconds
tt and/or tt:	Day
mm and/or mm:	Month
jj and/or jj:	Year
day of week:	Name of the day of the week
n:	Input for ’no’
j:	Input for 'yes'

3 Monitor functions for 07 KR 91 and 07 KT 92 to 07 KT 94

The monitor program offers the specialist access at hexadecimal level to the entire address range of the PLC. Memory areas can be displayed and modified, and hardware tests can be conducted.

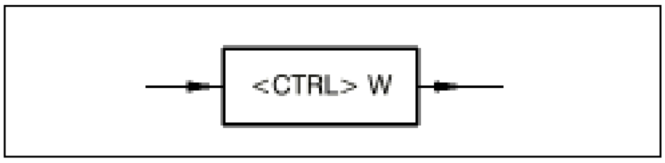
Monitor commands which change memory storage areas may endanger the functionality of the PLC. For this reason the monitor functions should be handled with care.

- The modules work with monitor commands with 32 bit addresses. The high word of the 32 bit address is set with the Y command.

  Example:
  ```
  Y = 20    <CR> }   The value of the address
  DW0       <CR> }   0020 0000H is displayed.
  ```

Switchover
operator control functions <--> Monitor functions

Command:



Command not available in 07 KR 31.

Function:

By pressing key <CTRL> and key W simultaneously the monitor program of the PLC is accessed. If the you are in the monitor program, you can change back to the operator-control program of the PLC by re-entering <CTRL> and W.

Explanation of the syntax:

- The monitor pogram responds with the character * and waits for an entry.
- All of the numbers are hexadecimal numbers (leading zeors may be omitted).
- If more digits than necessary are entered, only the last digits are valid (with byte commands the last two digits and with word commands the last 4 digits).
- The blank character (space) is ignored and can be used for more clearly structured entries.
- The character CTRL-C aborts the currently running operation.
- Every display on the screen can be stopped with <CTRL>S (XOFF) and be continued with <CTRL>Q (XON).
- If no segment is specified when entering an address, the working segment is used (see Y instruction).

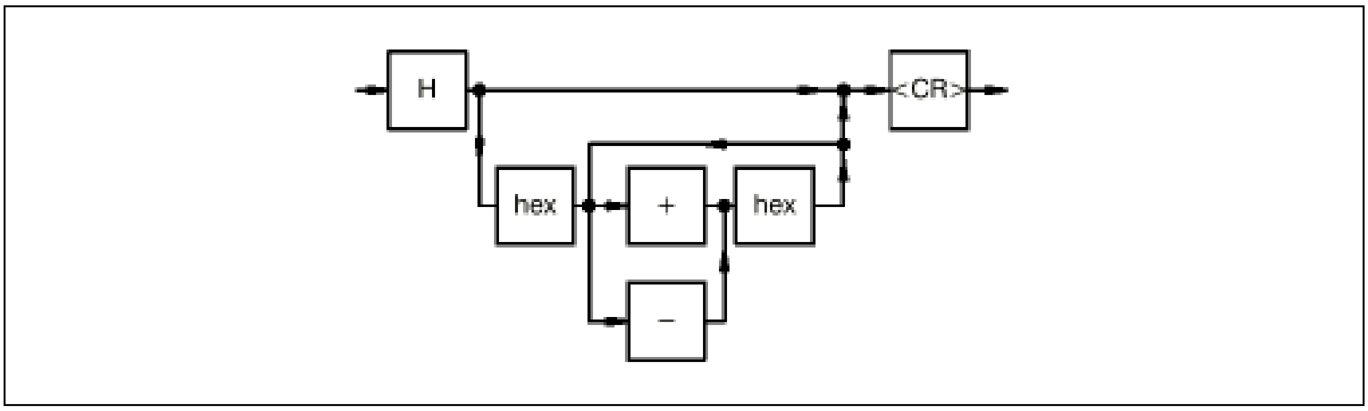
Overview of the monitor functions

Function and explanation		**Page**
H	Display help text / calculate hexadecimal	42
D	Display memory contents	42
I	Fill memory area with a value	43
M	Transfer memory areas	43
P	Read/write port	44
S	Display/edit memory contents	45
Y	Display/edit working segment	45
ZG	Simultaneous output of 3 values	46
ZR	RAM test	46
ZZA	Output of 3 values after entering a semicolon (;)	47
ZZF	Search for string	47
ZZV	Compare memory commands word-serially	48
R	Read Intel HEX file	49
W	Write Intel HEX file	49

Display help text / calculate hexadecimal

After the command H <CR> is entered, all of the available functions of the monitor are displayed on the screen. In addition, this command enables the calculation of simple hexadecimal arithmetic expressions.

Command:

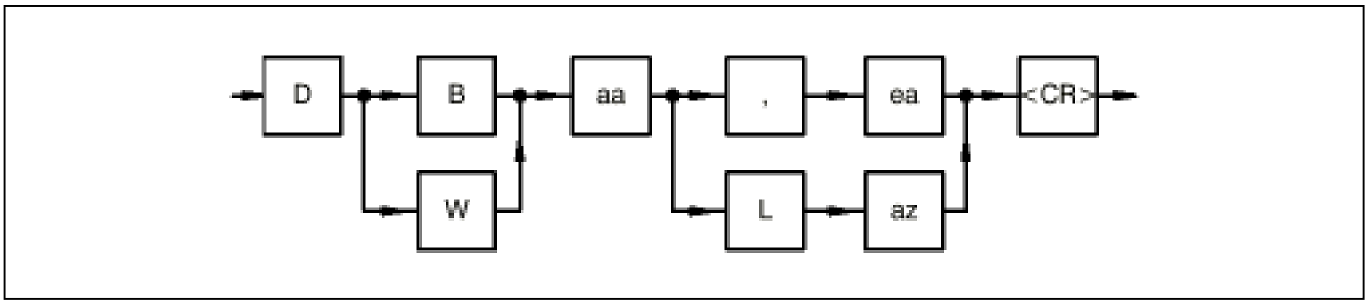


hex:	Hexadecimal numbers

Display memory contents

The memory contents can be displayed byte-serially or word-serially.

Command:



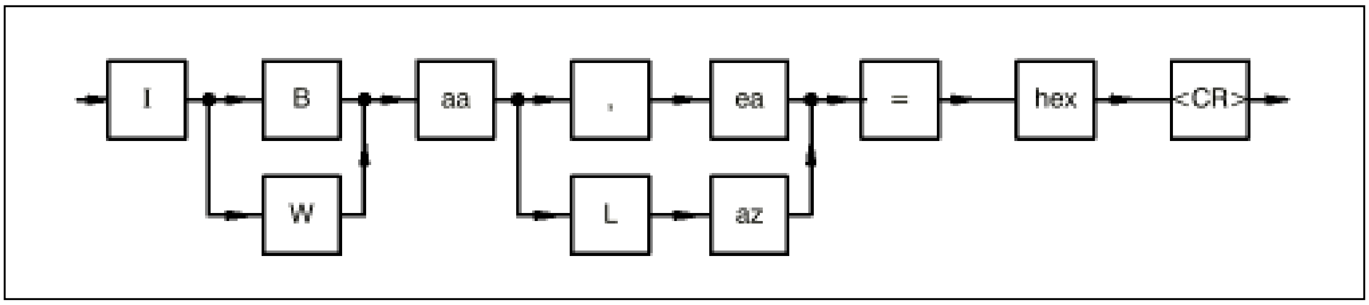
B:	Byte-serially (keyword)
W:	Word-serially (keyword)
aa:	Start address as of which the memory contents are to be displayed
,:	Keyword (separator)
ea:	End address of the memory contents to be output
L:	Length (keyword)
az:	Number of bytes/words to be output

Example:

DB 0:0L2<CR>	Display memory contents byte-serially
0000:0000 02 00	Monitor display
DW 0,2<CR>	Display memory contents word-serially
0000:0000 0002 0000	Monitor display

Fill memory area with a value

Command:



B:	Byte-serially (keyword)
W:	Word-serially (keyword)
aa:	Start address as of which the memory contents are to be filled with the specified value
,:	keyword (separator)
ea:	End address of the memory area
L:	Length (keyword)
az	Number of bytes/words to be filled
hex:	Hexadecimal value with which the memory area is to be filled

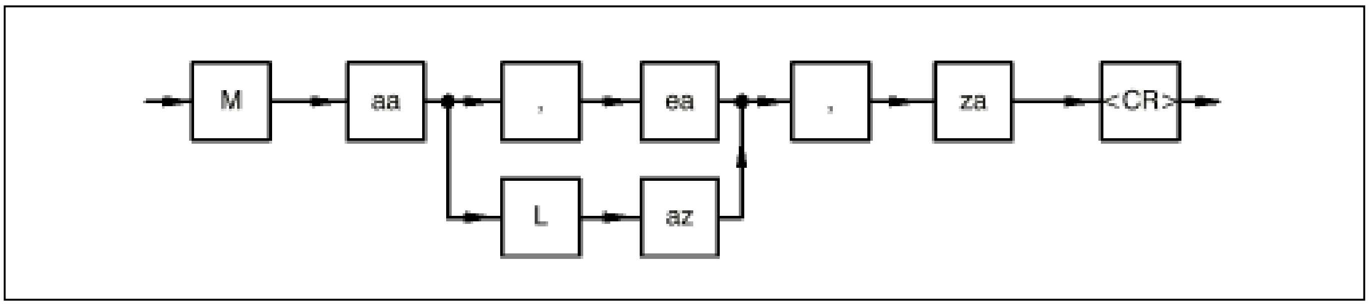
Example:

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 data are transferred word-serially, but the number is specified in bytes when entering (i.e. when az = 3 one word is transferred).

Command:



aa:	Start address as of which the memory contents are to be copied
,:	Keyword (separator)
ea:	End address of the memory area
L:	Length (keyword)
az:	Number of bytes to be copied
za:	Target address of the memory area

Example:

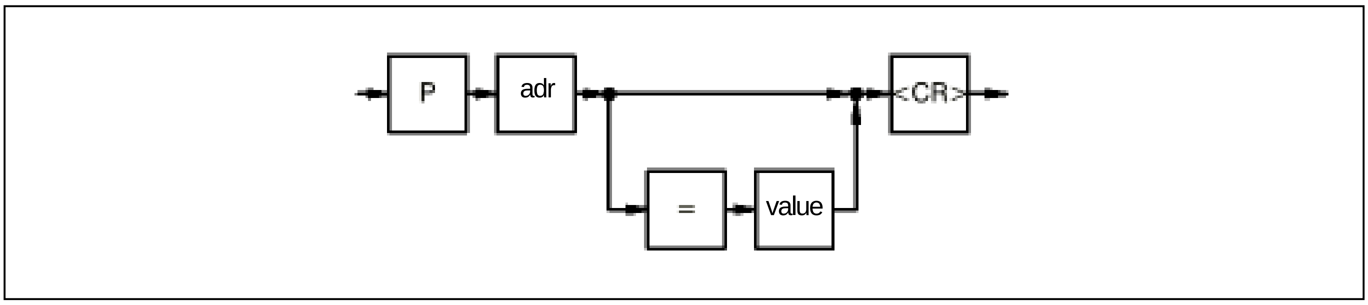
M 8000:80L4,8000:90<CR> or M 8000:80,84,8000:90<CR>

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

Command:

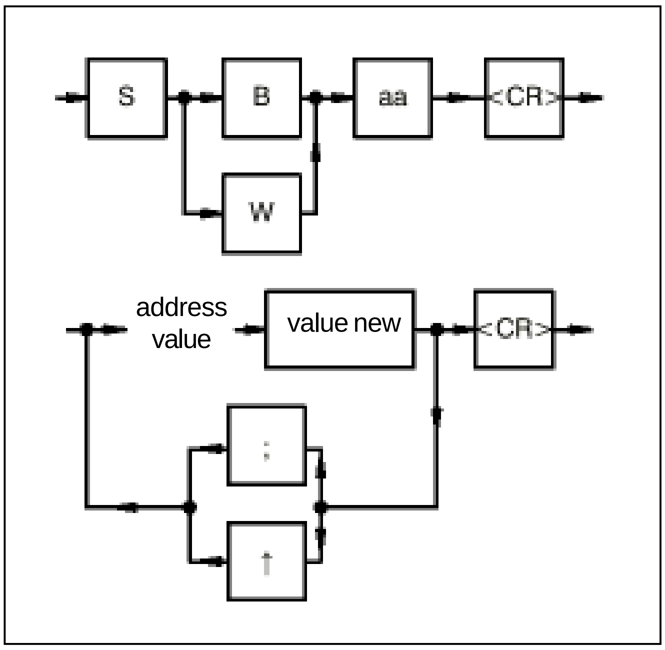


adr:	I/O-Address
value:	Byte value to which the I/O address is written
=:	Keyword

Display/edit memory contents

The memory contents can be displayed and edited byte-serially or word-serially.

Command:



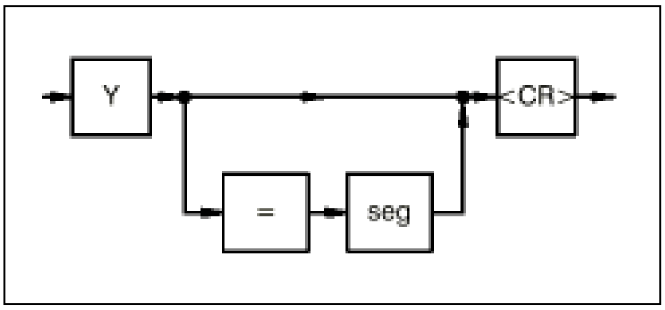
B:	Byte-serially (keyword)
W:	Word-serially (keyword)
aa:	Start address as of the which the memory contents are to be displayed/edited
address:	Address of the memory content
value:	Value of the memory contents
value new:	New value of the memory contents (user entry)
;:	Entering a semicolon increments the address by 1 (command SB) or by 2 (command SW).
↑:	Entering an "arrow up" (^ on the PC) decrements the adress by 1 (command SB) or by 2 (command SW)

Display/edit working segment

If only an offset and no segment is specified when entering an address, the working segment Y is used as the segment. The default value of the working segment is zero.

The modules work with monitor commands with 32 bit addresses. The high word of the 32 bit address is set with the Y command.

Command:



seg:	new segment address of the working segment
=:	keyword

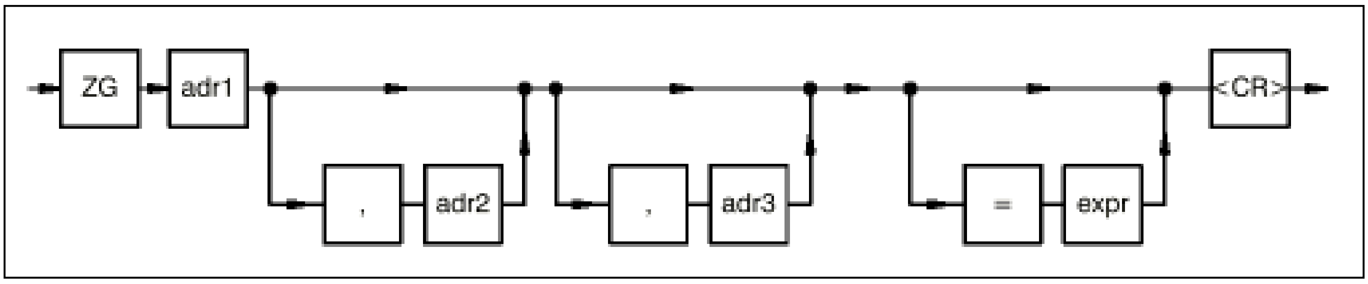
Example:

Y = 20 <CR>	User command: Edit high word of the 32 bit address
DW0 <CR>	Display memory contents word-serially
0020:0000 0200	Monitor display

Simultaneous display of 3 values

The ZG command permits the display of the values of maximum 3 addresses. Whenever the value of the first address changes, the values are updated on the monitor. The expression "expr" states how frequently updating of the values is to be suppressed.

Command:



adr1:	1st address whose value is displayed on the monitor. If the value of adr1 changes, the values are updated on the monitor.
adr2:	2nd address whose value is displayed on the monitor.
adr3:	3rd address whose value is displayed on the monitor.
expr:	Number expressing how frequently updating of the values on the monitor is to be suppressed when the value of adr1 changes.
,:	Keyword (separator)
=:	Keyword

Example:

ZG 1000:0, 1000:100 <CR>

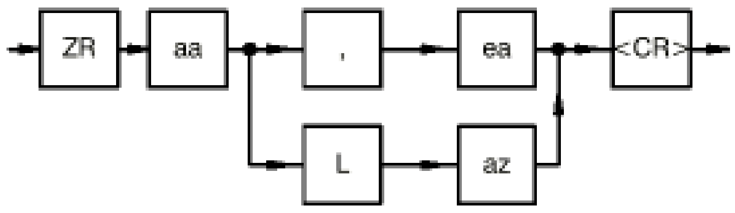
The values of addresses 1000:0$_H$ and 1000:100$_H$ are displayed on the monitor. If the value of address 1000:0$_H$ changes, the values of the two addresses are updated on the monitor.

RAM test

The specified area is written with a test pattern (FFFF, 5555, AAAA), and a check is then conducted in order to establish whether the test values have been stored correctly in the specified area. If an error is established, the address, actual value and required value are output. The test can be continued by pressing any key (apart from <SPACE>). CTRL C aborts the test.

3 test cycles are performed with test values whose order is reversed. The 4th test cycle consists of storing a counter at the start address, checking for correct storage and repeating the test with the decremented counter until it reaches value zero. The RAM test is then terminated with monitor message (*).

Command:



aa:	Start address of the RAM area
ea:	End address of the RAM area
L:	Length (keyword)
az:	Number of bytes of the RAM area
,:	Keyword (separator)

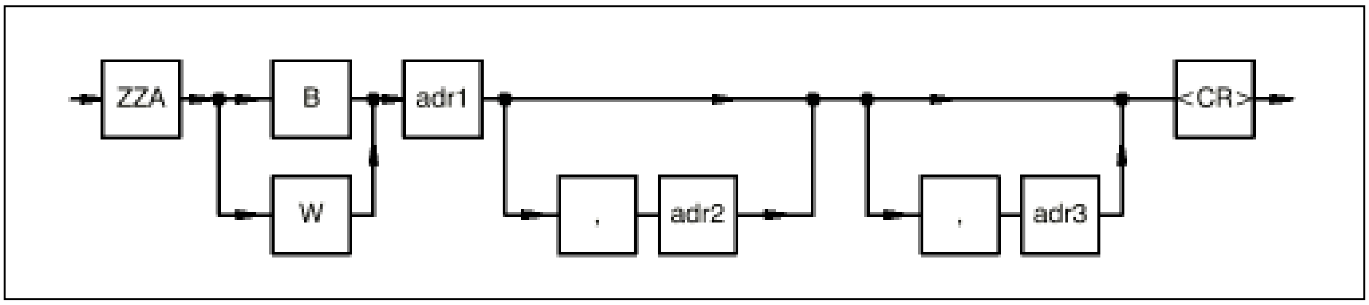
Example:

ZR 1000:0L100<CR>   RAM test over the specified memory area.

Output of 3 values after entering a semicolon (;)

Command ZZA permits the display of the values (byte or word) of maximum 3 addresses every time a semicolon is entered. The command can be aborted with <CR>.

Command:



B:	Byte-serially (keyword)
W:	Word-serially (keyword)
adr1:	1st address whose value is displayed on the monitor
adr2:	2nd address whose value is displayed on the monitor
adr3:	3rd address whose value is displayed on the monitor
,:	Keyword (separator)

Example:

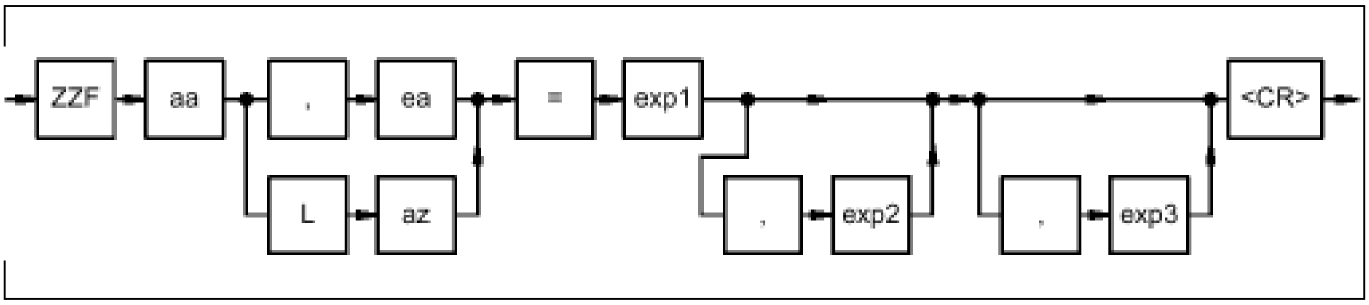
ZZA 1000:0, 1000:100 <CR>

After entry of a semicolon (;), the values of addresses 1000:0_H and 1000:100_H are displayed on the monitor.

Search for String

Command ZZF can be used to search for a string with maximum 3 words in the specified memory area. If the string is found, the address is displayed on the monitor. The search is continued by entering a semicolon (;). If the string is not found the monitor message <#07> is displayed.

Command:



aa:	Start address of the memory area
ea:	End address of the memory area
L:	Length (keyword)
az:	Number of words of the memory area
exp1:	1st word of the string
exp2:	2nd word of the string
exp3:	3rd word of the string
,:	Keyword (separator)

Example:

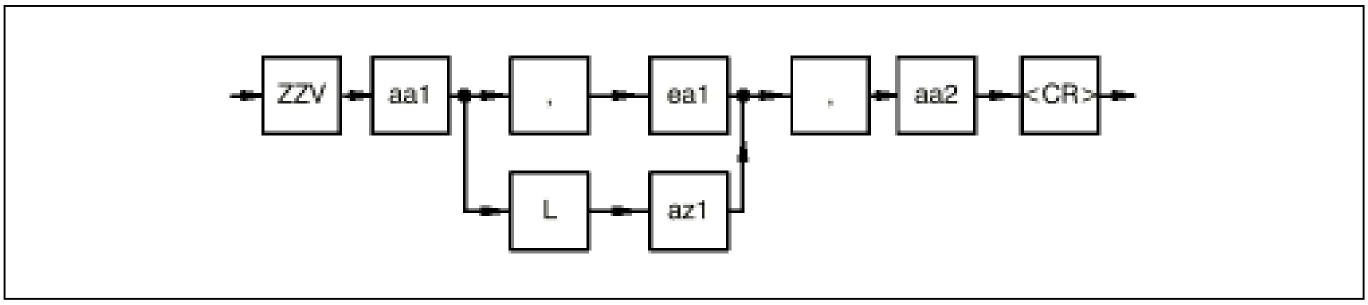
ZZF 1000:0, 100 = AAAA, BBBB <CR>

The entered string ($AAAA_H$, $BBBB_H$) is sought in the area 1000:0_H to 1000:100_H.

Compare memory areas word-serially

Command ZZV is used to compare a memory area 1 word-serially with a memory area 2. If a difference is detected, address 1, contents 1, address 2, and contents 2 are displayed on the monitor. The operation can be aborted with CTRL C.

Command:



aa1:	Start address of memory area 1
ea1:	End address of memory area 1
L:	Length (keyword)
az1:	Number of words of memory area 1
aa2:	Start address of memory area 2
,:	Keyword (separator)

Example:

ZZV A000:0 L 100, 8000:0 <CR>

The memory area 1 between A000:0_H and A000:100_H is compared with memory area 2 as of 8000:0_H.

Read INTEL HEX File

Using the R command it is possible to read in an INTEL HEX file via the serial interface COM2 of the basic units 07 KT 92/93/94 and to store the HEX file data in the PLC.

The following records are accepted in this case:

– address extension record
– data record
– end record

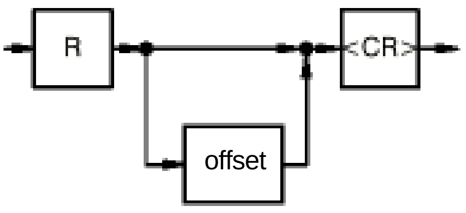
The following transfer format applies:

– 8 data bits
– no parity bit
– 1 stop bit

The data of the INTEL HEX files are stored in the PLC as of 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 when entering the command, this offset is added to the segment address in the address extension record. This results in a new segment address as of which the data of the HEX file are stored. Thus the storage area of the HEX file data can be presented in the PLC.
- The offset address is determined by the address in the data record of the INTEL HEX file.

Command:



offset:	Offset (the new segment address is calculated through addition of the offset to the segment address of the address extension record)

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 $2F00_H$ is added to the segment address of the address extension record. The resultant new segment address is the address used to store the HEX file data.

Write INTEL HEX file

The W command enables the output of a data area of the PLC as an INTEL-HEX file via the serial interface COM2 of the basic units 07 KT 92/93/94.

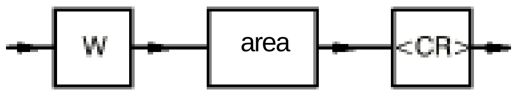
The following records are accepted in this case:

– address extension record
– data record
– end record

The following transfer format applies:

– 8 data bits
– no parity bit
– 1 stop bit

Command:



area:	Memeory area to be output as Intel Hex file

Example:

W 8000:0,FFFF <CR>	The memory area from $8000{:}0_H$ up to $8000{:}FFFF_H$ is output as INTEL HEX file via the serial interface COM2 of the PLC.
W 8000:0LFFFF <CR>	The memory area from $8000{:}0_H$ up to $8000{:}FFFF_H$ is output as INTEL HEX file via the serial interface COM2 of the PLC.

4 Memory overviews

4.1 Memory overviews for 07 KT 94

Important:
At blocks which work with segments (COPY, WAES, DWAES, WOL, DWOL, WOS, DWOS), the value of the descriptor must be assigned at the segment inputs.

Example:
At the 07 KT 94 the flag MW 0,00 is to be copied to MW 1,00:

COPY block:

Value at input:	FREI	$= 1_D$	
	ANZ	$= 1_D$	
	QOFF	$= 8A80_H$	(35456_D)
	QSEG	$= B8_H$	(184_D)
	ZOFF	$= 8AA0_H$	(35488_D)
	ZSEG	$= B8_H$	(184_D)

4.1.1 User program RAM

Explanation of terms:

- ARCNET intermediate store: Descriptor 176_D $(B0_H)$
- Organizational directory: Descriptor 168_D $(A8_H)$
  - PLC-specific: This is used to store organizational data relating to the entire PLC.
  - For AWP 1: This is used to store organizational data relating to program memory 1.
  - For AWP 2: This is used to store organizational data relating to program memory 2.
- Program identification:
  16 bytes for an identification, e.g. project name
- User program memory 1 and/or 2:
  Memory for the PLC program

  User program memory 1:
  Descriptor 328_D to 384_D $(148_H$ to $180_H)$

  User program memory 2:
  Descriptor 600_D to 656_D $(258_H$ to $290_H)$
- Turbo RAM program 1 and/or 2:
  Machine code for user program memory 1 and/or 2
- Constants for program 1 and/or 2:
  This area is used to store the indirect constants of the user program memory 1 and/or 2.
  Constants for program1: Descriptor 448_D $(1C0_H)$
  Constants for program 2: Descriptor 464_D $(1D0_H)$
- Operand memory: Descriptor 184_D $(B8_H)$

Turbo RAM program 2 30000_H bytes Descriptors 768_D to 784_D $(300_H$ to $310_H)$	002FFFFF 002D0000
reserved	002CFFFF 002C0000
CS31 EA copy 800_H bytes	002BF7FF 002BF000
reserved	002BEFFF 002B0000
Turbo RAM program 2 40000_H bytes Descriptors 736_D to 760_D $(2E0_H$ to $2F8_H)$	002AFFFF 00270000
Turbo RAM program 1 70000_H bytes Descriptors 392_D to 440_D $(188_H$ to $1B8_H)$	0026FFFF 00200000
Operand memory	001FFFFF 001F0000
Constants for program 2 1142_H bytes	001C0000
User program memory 2 $3EE40_H$ bytes	001BEE3F 00180000
Constants for program 1 1142_H bytes	00170000
User program memory 1 $3EE40_H$ bytes	0016EE3F 00130000
Program identification	0012FF7F 0012FF70
Organizational directory for program 2	0012FF5A
Organizational directory for program 1	0012FF44
Organizational directory PLC-specific	0012FF30
Control block 0...2	0012FF00
ARCNET intermediate store 6220_H bytes	0011621F 00110000

User program RAM

4.1.2 User program Flash EPROM

User data segment 3 7860_H bytes	02470000
User data segment 2 7860_H bytes	02460000
User data segment 1 7860_H bytes	02450000
User data segment 0 7860_H bytes	02440000
Checksum 0 to 3	0243FFF8
not used 6_H bytes	0243FFF2
User program part 3 $EE40_H$ bytes	024311B2
Constants 1142_H bytes	02430070
not used 20_H bytes	02430050
Program identification	02430040
Organizational directory for program 2	0243002A
Organizational directory for program 1	02430014
Organizational directory PLC-specific	02430000
User program part 2 10000_H bytes	02420000
User program part 1 10000_H bytes	02410000
User program part 0 10000_H bytes	02400000

- User data segment 0 to segment 3:
  The user data segment 0 to segment 3 are stored in this area. The user data are secured with a check-sum.

4.1.3 Operand memory 07 KT 94

1FF200			SEG:F200
1FF100	**Stack 2**	100_H	SEG:F100
1FF000	**ASAS**	100_H	SEG:F000
1FEA80		580_H	SEG:EA80
1FDA80	**S**	1000_H	SEG:DA80
1FCA80	**MD**	1000_H	SEG:CA80
1F8A80	**MW**	4000_H	SEG:8A80
1F6A80	**M**	2000_H	SEG:6A80
1F6420	**AW**	660_H	SEG:6420
1F55E0	**A**	$E40_H$	SEG:55E0
1F4F80	**EW**	660_H	SEG:4F80
1F4140	**E**	$E40_H$	SEG:4140
1F3B40	**KD**	600_H	SEG:3B40
1F3000	**KW**	$B40_H$	SEG:3000
1F2FFE	**K**	2_H	SEG:2FFE
1F2EF0	**Free Pool**	$10E_H$	SEG:2EF0
1F2200		$CF0_H$	SEG:2200
1F0200	**VWS**	2000_H	SEG:0200
1F0100	**Stack 1**	100_H	SEG:0100
1F0000	**ASAS 1**	100_H	SEG:0000

Explanation of terms:

ASAS 1:	Work memory program 1
Stack1:	Stack for program 1
VWS:	Historical value memory
K:	Indirect constants BINARY K 00,00...K 00,01
KW:	Indirect constants WORD KW 00,00...KW 89,15
KD:	Indirect constants DOUBLE WORD KD 00,00...KD 23,15
E:	Process image of the inputs BINARY E 00,00...E 99,15 E 100...E 163,15 E 200...E 263,15
EW:	Process image of the inputs WORD EW 00,00...EW 34,15 EW 100,00...EW 107,15 EW 200,00...EW 207,15
A:	Process image of the outputs BINARY A 00,00...A 99,15 A 100,00...A 163,15 A 200,00...A 263,15
AW:	Process image of the outputs WORD AW 00,00...AW 34,15 AW 100,00...AW 107,15 AW 200,00...AW 207,15
M:	Flags BINARY M 00,00...M 511,15
MW:	Flags WORD MW 00,00...MW 511,15
MD:	Flags DOUBLEWORD MD 00,00...MD 63,15
S:	Step chains S 00,00...S 255,15
ASAS2:	Work memory for program 2
Stack 2:	Stack for program 2

Important:
At blocks which work with segments (COPY, WAES, DWAES, WOL, DWOL, WOS, DWOS), the value of the descriptor must be assigned at the segment inputs.

For examples see chapter 4.1.

SEG=$001F_H$
(High word of the 32 bit address,
descriptor $0B8_H = 184_{DEC}$)

4.1.3 Dual port RAM: CS31

CS31: **AW 08,00...AW 15,15**	SEG:05FF SEG:0500
CS31: **EW 08,00...EW 15,15**	SEG:04FF SEG:0400
CS31 status (EW 07,15)	SEG:03FF SEG:03FE
Read real time clock (EW 07,08...EW 07,14)	SEG:03FD SEG:03F0
Spontaneous mail box (EW 07,04...EW 07,07)	SEG:03EF SEG:03E8
Receive mail box EW 07,00...EW 07,03	SEG:03E7 SEG:03E0
CS31: **EW 00,00...EW 05,15**	SEG:03DF SEG:0300
reserved	SEG:02FF
Send mail box	SEG:02FE SEG:02F4
reserved	SEG:02F3 SEG:02E0
CS31: **AW 00,00...AW 05,15**	SEG:02DF SEG:0200
CS31: **E 00,00...E 61,15**	SEG:01FF SEG:0180
reserved	SEG:017F SEG:0080
CS31: **A 00,00...A 61,15**	SEG:007F SEG:0000

SEG=0050$_H$
(High word of the 32 bit address, descriptor 0C8$_H$ = 200$_{DEC}$)

4.1.4 Dual port RAM: Inputs and outputs of the basic module

All of the memory areas which are not allocated are **reserved.**	
EW 06,00...EW 06,15	SEG:006F SEG:0050
AW 06,00...AW 06,15	SEG:00AF SEG:0090
E 62,00...E 63,15	SEG:0033 SEG:0030
E 64,00...E 64,07	SEG:0034
A 62,00...A 63,15	SEG:0042 SEG:0040

SEG=0060$_H$
(High Word of the 32 bit address, descriptor 058$_H$ = 88$_{DEC}$)

4.2 Memory overviews for 07 KR 91

4.2.1 User program RAM

not used	38C20 38C12
Constants for program 2 702_H bytes	38510
Turbo RAM program 2 $12EB0_H$ bytes	25660
User program memory 2 7800_H bytes	1DE60
not used	1DE52
Constants for program 1 702_H bytes	1D750
Turbo RAM program 1 $12EB0_H$ bytes	0A8A0
User program memory 1 7800_H bytes	030A0
not used	03080
Program identification	03070
Organizational directory for program 2	0305A
Organizational directory for program 1	03044
Organizational directory PLC-specific	03030
Control block 0...2	03000

4.2.2 User program Flash EPROM

Checksum	A7FFE
not used $8C_H$ bytes	A7F72
User programm 7800_H bytes	A0772
Constants 702_H bytes	A0070
not used 20_H bytes	A0050
Program identification	A0040
Organizational directory for program 2	A002A
Organizational directory for program 1	A0014
Organizational directory PLC-specific	A0000

Explanation of terms:

- Organizational directory
  - PLC-specific: This is used to store organizational data relating to the entire PLC.
  - For AWP 1: This is used to store organizational data relating to program memory 1.
  - For AWP 2: This is used to store organizational data relating to program memory 2.
- Program identification:
  16 bytes for an identification, e.g. project name
- User program memory 1:
  Memory for the PLC-program
- Turbo RAM program 1:
  Machine code for user program memory 1
- Constants for program 1:
  This area is used to store the indirect constants of the user program memory 1.
- User program memory 2:
  Speicher für das SPS-Programm
- Turbo RAM program 2:
  Machine code for user program memory 2
- Constants for program 2:
  This area is used to store the indirect constants of the user program memory 2.

4.2.3 Operand memory

40000			SEG:F3E0
3FFF0	not used	10_H	SEG:F3D0
3FE60	I/O configuration list 2	190_H	SEG:F240
3FCD0	I/O configuration list 1	190_H	SEG:F0B0
3FCC8	not used	8_H	SEG:F0A8
3FA40	I/O force lists	288_H	SEG:EE20
3FA30	not used	10_H	SEG:EE10
3F930	Stack 2	100_H	SEG:ED10
3F830	ASAS 2	100_H	SEG:EC10
3E030	VWS	1800_H	SEG:D410
3D830	S	800_H	SEG:CC10
3D030	MD	800_H	SEG:C410
3B030	MW	2000_H	SEG:A410
3A030	M	1000_H	SEG:9410
39F30	AW	100_H	SEG:9310
39B30	A	400_H	SEG:8F10
39A30	EW	100_H	SEG:8E10
39630	E	400_H	SEG:8A10
39430	KD	200_H	SEG:8810
38F30	KW	500_H	SEG:8310
38F2E	K	2_H	SEG:830E
38E20	Free Pool	$10E_H$	SEG:8200
38D20	Stack 1	100_H	SEG:8100
38C20	ASAS 1	100_H	SEG:8000
			SEG=30C2

Explanation of terms:

ASAS 1:	Work memory program 1
Stack 1:	Stack for program 1
K:	Indirect constant BINARY
KW:	Indirect constant WORD
KD:	Indirect constant DOUBLE WORD
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:	Flags BINARY
MW:	Flags WORD
MD:	Flags DOUBLE WORD
S:	Step chains
VWS:	Historical value memory
ASAS 2:	Work memory for program 2

I/O force lists:
This is where the I/O signals to be forced and their force values are entered.

I/O configuration list 1:
This is where the I/O signals planned in program 1 are entered so that they are taken into consideration when generating and outputting the process image.

I/O configuration list 2:
This is where the I/O signals planned in program 2 are entered so that they are taken into consideration when generating and outputting the process image.

4.2.4 Dual-Port RAM

CS31 status (EW 07,15)	C03FF C03FE
Read real time clock EW 07,08...EW 07,14	C03FD C03F0
Spontaneous mail box EW 07,04...EW 07,07	C03EF C03E8
Receive mail box EW 07,00...EW 07,03	C03E7 C03E0
Direct: EW 06,00...EW 06,15 **CS31: EW 00,00...EW 05,15**	C03DF C0300
reserved	C02FF
Send mail box	C02FE C02F4
reserved	C02F3 C03E0
Direct: AW 06,00...AW 06,15 **CS31: AW 00,00...AW 05,15**	C02DF C0200
Direct: E 62,00...E 63,15 **CS31: E 00,00...E 61,15**	C01FF C0180
reserved	C017F C0100
Read/write permission **Read/write request**	C00FF C00FE
reserved	C00FD C0080
Direct: A 62,00...A 63,15 **CS31: A 00,00...A 61,15**	C007F C0000

5 Functions in the instruction list for 07 KR 91 and 07 KT 92 to 07 KT 94

5.1 Texts in the instruction list

Some PLC blocks (DRUCK, EMAS) operate with texts stored in the user program.

Entering the texts in the user program

With a terminal, text is entered embedded in the code characters #" and "#. The key code character #" identifies the start of a text string and the key code character "# identifies the end of a text string.

All ASCII characters between 0_H and $7F_H$ may be entered.

Storing the texts in the user program

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

Example:

Text entry and storage as of address 100 in the PLC program:

```
Entry:    S 100  <CR>
          00100  NOP
          #"ABB"#<CR>

Storage:  00100  FA41
          00101  FA42
          00102  FA42
```

Overview of the possible text characters, their entry and their display on the monitor

ASCII character	Hex-value	User entry	Display	ASCII character	Hex-value	User entry	Display
NUL	00	<CTRL><SP>	<NUL>*	0	30	0	0
SOH	01	<CTRL> A	<SOH>	1	31	1	1
STX	02	<CTRL> B	<STX>	2	32	2	2
ETX	03	<CTRL> C	<ETX>	3	33	3	3
EOT	04	<CTRL> D	<EOT>	4	34	4	4
ENQ	05	<CTRL> E	<ENQ>	5	35	5	5
ACK	06	<CTRL> F	<ACK>	6	36	6	6
BEL	07	<CTRL> G	<BEL>	7	37	7	7
BS	08	<CTRL> H	<BS>	8	38	8	8
HT	09	<CTRL> I	<HT>	9	39	9	9
LF	0A	<CTRL> J	<LF>	:	3A	.	.
VT	0B	<CTRL> K	<VT>	;	3B	;	:
FF	0C	<CTRL> L	<FF>	<	3C	<	<
CR	0D	<CTRL> M	<CR>	=	3D	=	=
SO	0E	<CTRL> N	<SO>	>	3E	>	>
SI	0F	<CTRL> O	<SI>	?	3F	?	?
DLE	10	<CTRL> P	<DLE>	@	40	@	@
DC1	11	<CTRL> Q	<DC1>	A	41	A	A
DC2	12	<CTRL> R	<DC2>	B	42	B	B
DC3	13	<CTRL> S	<DC3>	C	43	C	C
DC4	14	<CTRL> T	<DC4>	D	44	D	D
NAK	15	<CTRL> U	<NAK>	E	45	E	E
SYN	16	<CTRL> V	<SYN>	F	46	F	F
ETB	17	<CTRL> W	<ETB>	G	47	G	G
CAN	18	<CTRL> X	<CAN>	H	48	H	H
EM	19	<CTRL> Y	<EM>	I	49	I	I
SUB	1A	<CTRL> Z	<SUB>	J	4A	J	J
ESC	1B	<CTRL> [	<ESC>	K	4B	K	K
FS	1C	<CTRL> \	<FS>	L	4C	L	L
GS	1D	<CTRL>]	<GS>	M	4D	M	M
RS	1E	<CTRL> ~	<RS> *	N	4E	N	N
US	1F	<CTRL> ?	<US> *	O	4F	O	O
SP	20	<SP>	<SP> **	P	50	P	P
!	21	!	!	Q	51	Q	Q
"	22	"	"	R	52	R	R
#	23	#	#	S	53	R	R
$	24	$	$	T	54	T	T
%	25	%	%	U	55	U	U
&	26	&	&	V	56	V	V
'	27	'	'	W	57	W	W
(	28	(	(	X	58	W	W
)	29	)	)	Y	59	Y	Y
*	2A	*	*	Z	5A	Z	Z
+	2B	+	+	[	5B	[	[
,	2C	,	,	\	5C	\	\
–	2D	–	–	]	5D	]	]
.	2E	.	.	^	5E	^	^
/	2F	/	/	_	5F	_	_

ASCII character	Hex-value	User entry	Display
`	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
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>

* The follwing applies with older terminals:

NUL	00	<CTRL>@	<NUL>
RS	1E	<CTRL>^	<RS>
US	1F	<CTRL>_	<US>

** In the case of text entry a SPACE is displayed as a blank. When displaying the user program it is displayed as <SP> for easier recognition.

Note:
Interrelationship between value and ASCII character:

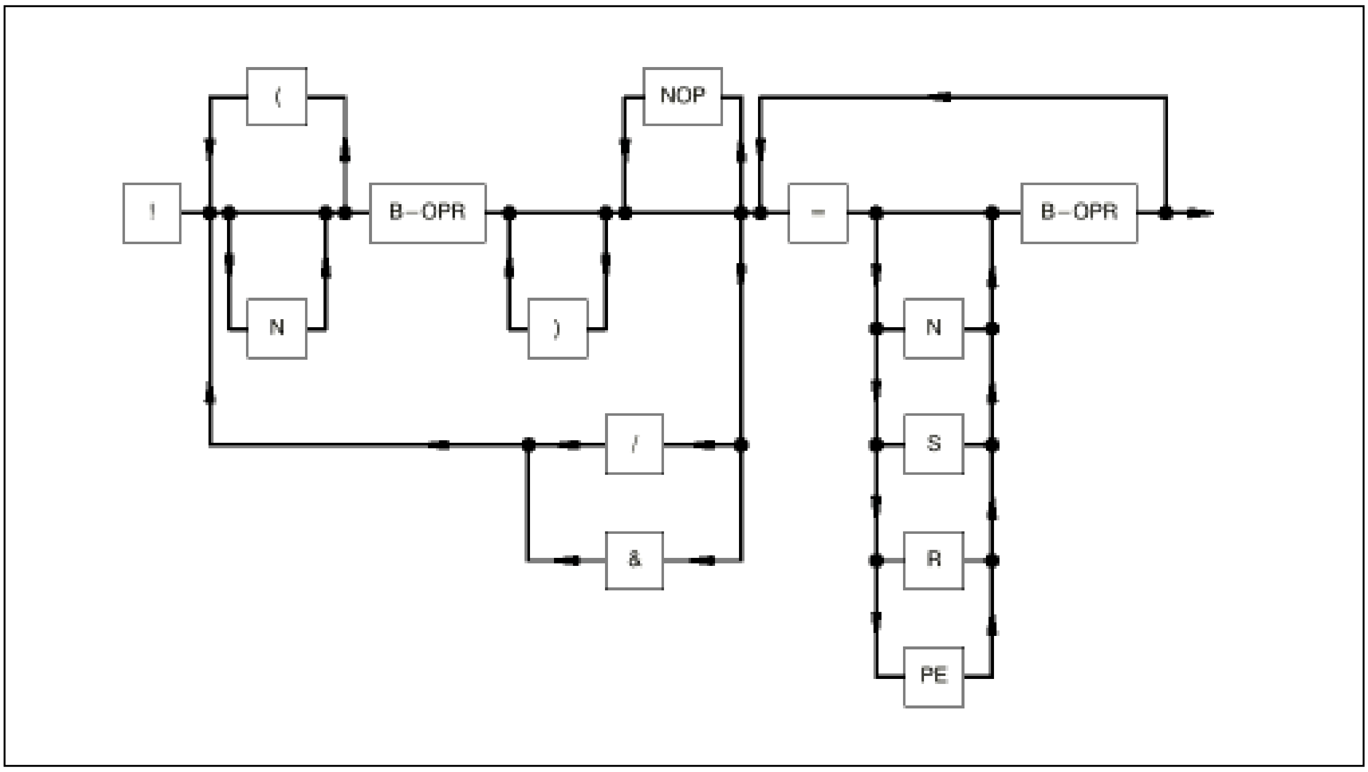
Binary value in the computer:
1011 ($0B_H$, 11_{DEC})

Represented as:
– Decimal ASCII: 31_H, 31_H
– Hexadecimal ASCII: 42H

Output by DRUCK as hex value: 1011

5.2 Syntax diagrams for instruction list (IL)

5.2.1 Syntax diagram: BOOLEAN SENTENCE

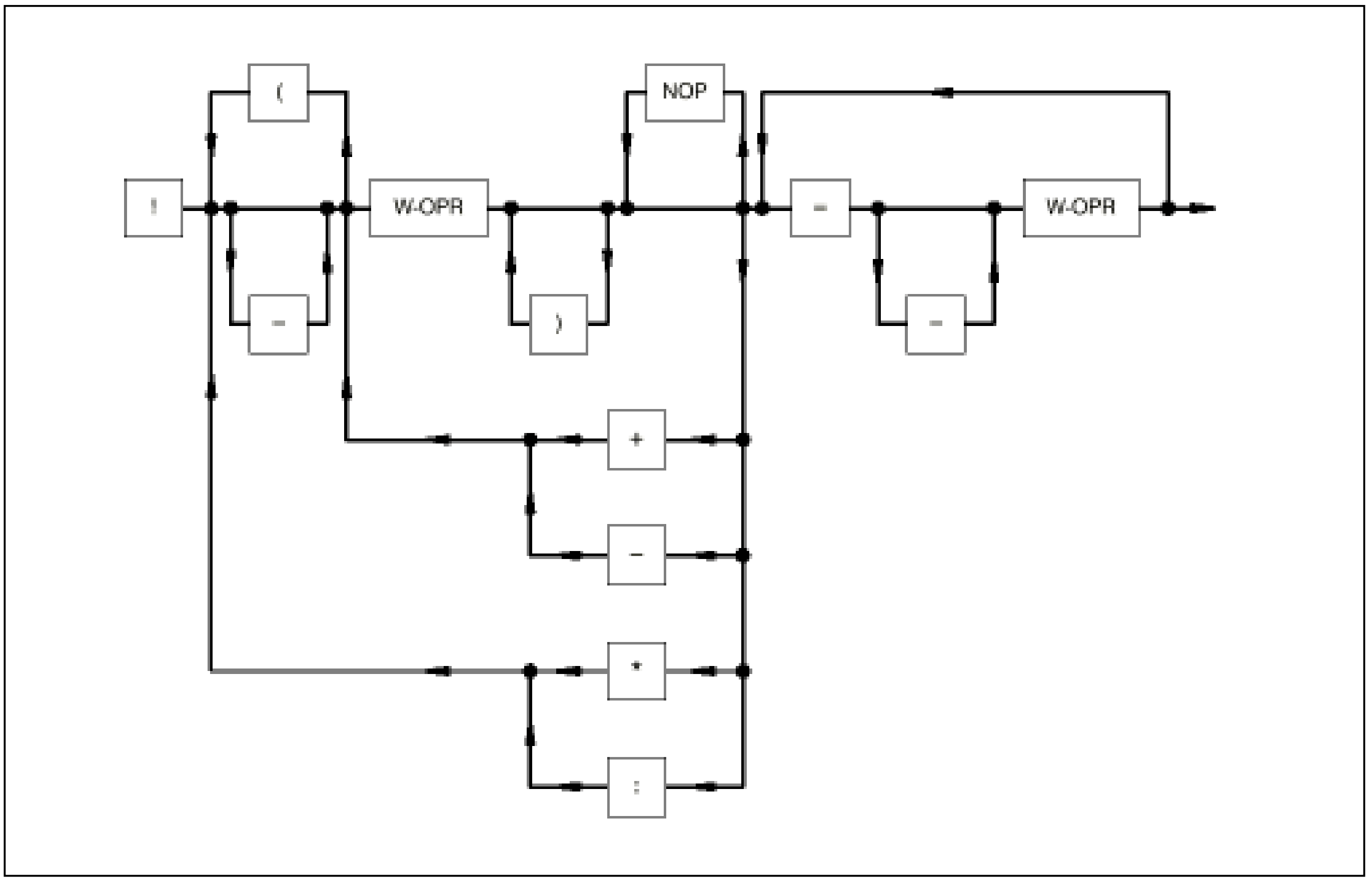


Signal flow: In the direction of the arrow, otherwise from left to right

Brackets: Sum "LEFT BRACKET" = Sum "RIGHT BRACKET", nesting depth: 15

B-OPR: Binary operand (E, A, M, S, K)
Examples: E 00,03 A 07,15 M 05,01 S 05,04 K 00,01

5.2.2 Syntax diagram: ARITHMETIC SENTENCE



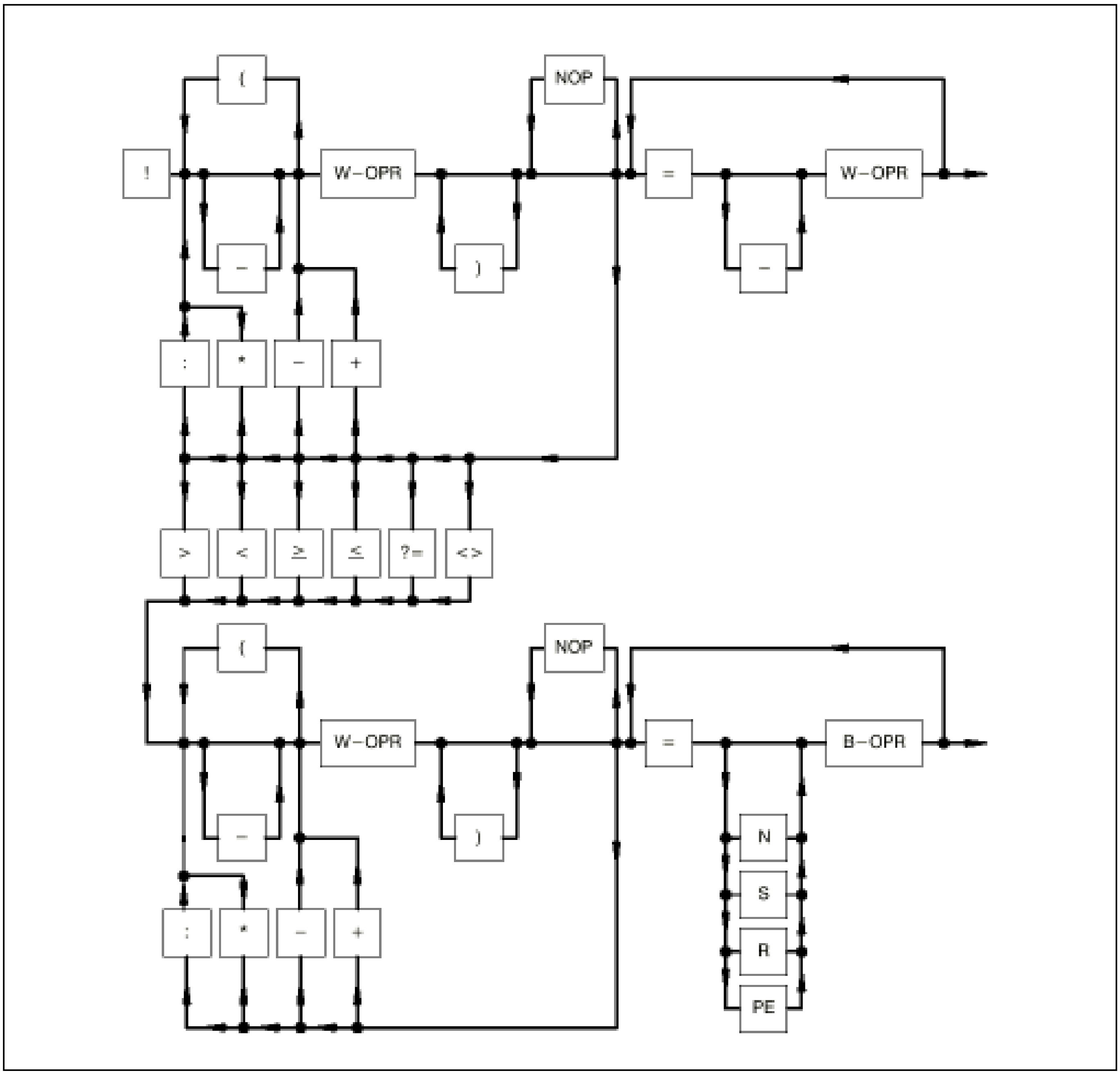
Signal flow: In the direction of the arrow, otherwise from left to right

Brackets: Sum "LEFT BRACKET" = Sum "RIGHT BRACKET", nesting depth: 15

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

5.2.3 Syntax diagram: HYBRID SENTENCE

see also chapter "Language repertoire", Relational operators



Signal flow: In the direction of the arrow, otherwise from left to right

Brackets: Sum "LEFT BRACKET" = Sum "RIGHT BRACKET", nesting depth: 15

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

B-OPR: Binary operand (E, A, M, S, K)
Examples: E 00,03 A 07,06 M 05,01

Printed on chlorine-free bleached paper

ABB Schalt- und Steuerungstechnik GmbH

Eppelheimer Straße 82	Postfach 10 50 09
D-69123 Heidelberg	D-69040 Heidelberg

Telephone	+49 6221 777-0
Telefax	+49 6221 777-111
E-Mail	desst.helpline@de.abb.com
Internet	http://www.abb-sst.de

Printed in the Federal Republic of Germany (09.99)