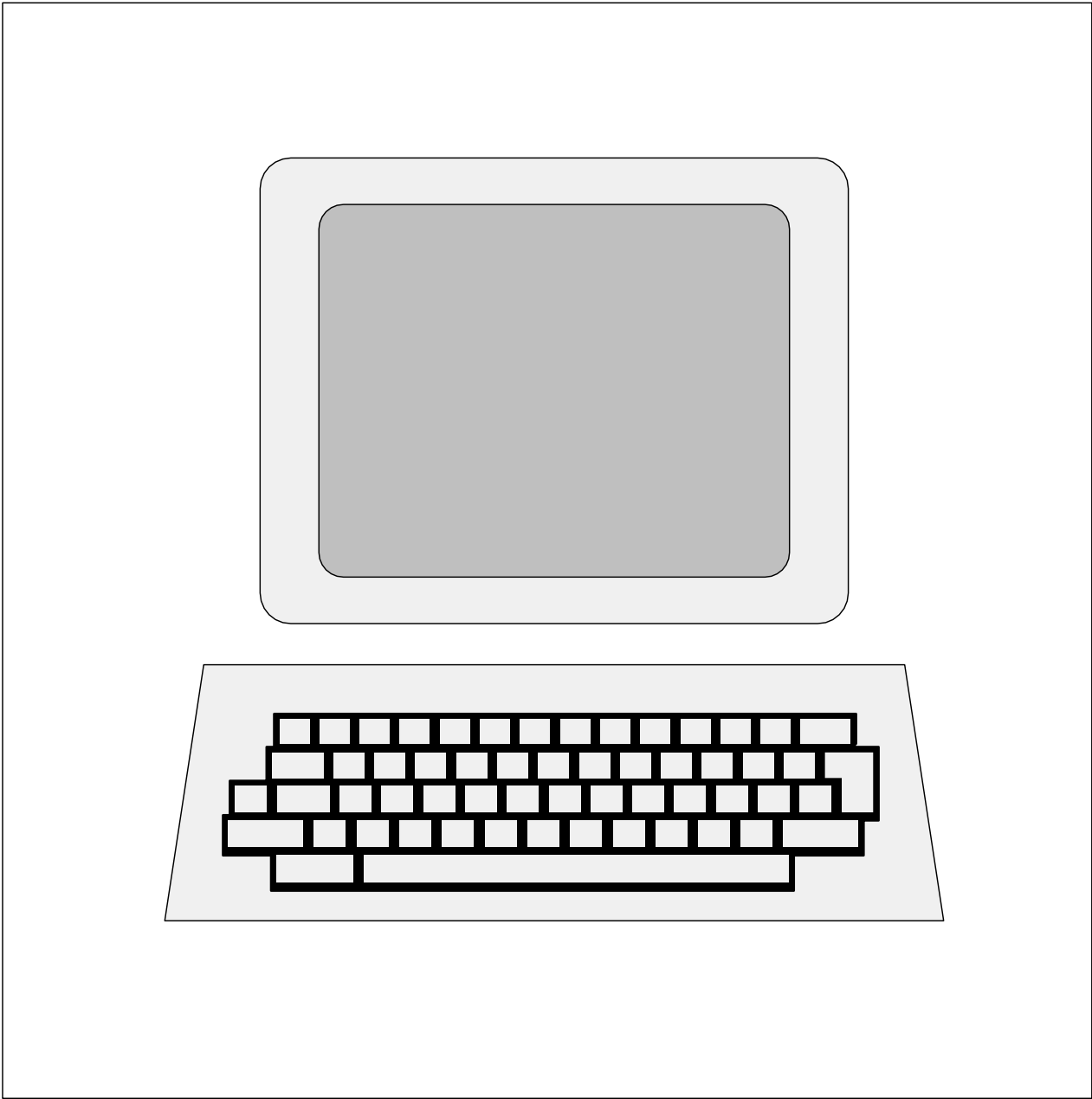


Monitor



Regulations Concerning the Setting up of Installations

Apart from the basic "Regulations for the Setting up of Power Installations" DIN VDE* 0100 and for "The Rating of Creepage Distances and Clearances" DIN VDE 0110 Part 1 and Part 2 the regulations "The Equipment of Power Installations with Electrical Components" DIN VDE 0160 in conjunction with DIN VDE 0660 Part 500 have to be taken into due consideration.

Further attention has to be paid to DIN VDE 0113 Part 1 and Part 200 in case of the control of working and processing machines. If operating elements are to be mounted near parts with dangerous contact voltage DIN VDE 0106 Part 100 is additionally relevant.

If the protection against direct contact according to DIN VDE 0160 is required, this has to be ensured by the user (e.g. by incorporating the elements in a switch-gear cabinet). The devices are designed for pollution severity 2 in accordance with DIN VDE 0110 Part 1. If higher pollution is expected, the devices must be installed in appropriate housings.

The user has to guarantee that the devices and the components belonging to them are mounted following these regulations. For operating the machines and installations, other national and international relevant regulations, concerning prevention of accidents and using technical working means, also have to be met.

The ABB Procontic devices are designed according to IEC 1131 Part 2. Meeting this regulation, they are classified in overvoltage category II which is in conformance with DIN VDE 0110 Part 2.

For the direct connection of ABB Procontic devices, which are powered with or coupled to AC line voltages of overvoltage category III, appropriate protection measures corresponding to overvoltage category II according to IEC-Report 664/1980 and DIN VDE 0110 Part 1 are to install.

Equivalent standards:

DIN VDE 0110 Part 1 \triangleq IEC 664

DIN VDE 0113 Part 1 \triangleq EN 60204 Part 1

DIN VDE 0660 Part 500 \triangleq EN 60439-1 \triangleq IEC 439-1

All rights reserved to change design, size, weight, etc.

* VDE stands for "Association of German Electrical Engineers".

ABB Schalt- und Steuerungstechnik GmbH Heidelberg

Table of Contents

1	Operating and test functions	1– 1
1.1	Commands for creating the user program	1– 3
1.2	Commands for testing the user program	1– 9
1.3	Commands for configuring	1–17
1.4	Texts in the instruction list	1–22
1.5	Syntax diagram for instruction list (IL)	1–24
1.5.1	Syntax diagram: BOOLEAN SENTENCE	1–24
1.5.2	Syntax diagram: ARITHMETIC SENTENCE	1–25
1.5.3	Syntax diagram: HYBRID SENTENCE	1–26
2	Monitor functions	2– 1
3	Memory overview	3– 1
3.1	Memory overview for 07 KR 91, 07 KT 92 and 07 KT 93	3– 1
3.1.1	User program RAM	3– 1
3.1.2	User program Flash–EPROM	3– 1
3.1.3	Operand memory	3– 2
3.1.4	Dual–port RAM	3– 2
3.2	Memory overview for 07 KR 31, and 07 KT 31	3– 3
3.2.1	System addressing (Mapping)	3– 3
3.2.2	Data addressing (Data mapping)	3– 3

1 Operating and test functions

The operating and test functions of the PLC can be used with the aid of a terminal, the TCZ service device or the ABB Procontic programming system.

Note:

If the user works with the ABB Procontic programming and test system, this provides him with a convenient operator interface. When communicating with the control system, the ABB Procontic programming and test system uses the operating and test functions described in this chapter.

The ABB Procontic programming and test system has its own operating instructions.

Operator control commands

The operator control commands can be subdivided into:

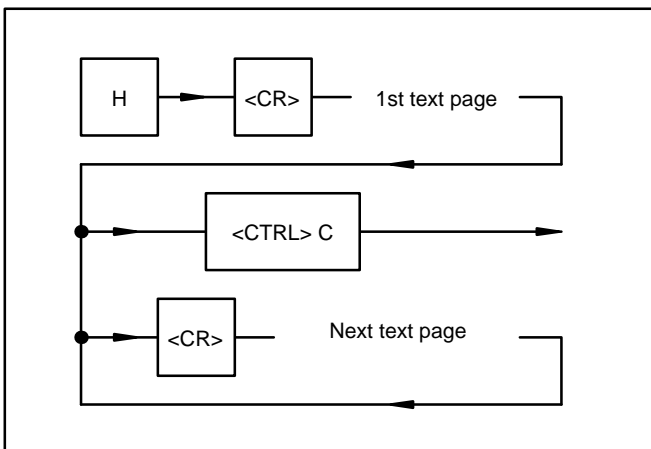
- Commands for creating and modifying user programs
- Commands for testing the user programs
- Commands for configuring the PLC

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 screen.

Help command



Function:
All available operator control and test functions are

displayed on the monitor. Use <CR> to scroll the HELP text.

Note on service device TCZ:

The four-line liquid-crystal display (LCD) of the service device TCZ does not suffice to display this command.

Commands for creating the user program (overview)

Command Function	Page
AEND Prepare a program change on a running PLC program	1- 3
AEND Reject a program change which has not yet been enabled	1- 3
ALT Reject an enabled program change on a running PLC program and reactivate the old program status	1- 3
AL Display PLC capacity utilization	1- 3
CROSS *) Display CROSS reference list	1- 4
D Display program	1- 5
DEEP Erase PLC program on Flash EPROM	1- 5
F*) Search for string in user program (Find)	1- 5
FREI Enable a program change on a running PLC program	1- 5
IDA Display program identification	1- 6
IDR Delete program identification	1- 6
IDS Enter program identification	1- 6
K Enter/edit values of indirect constants	1- 6
NOP Delete program part, i.e. overwrite program part with NOPs ...	1- 7
O Optimize program	1- 7
P Display free program memory area	1- 8
PA*) Program preparation	1- 8
S Enter/edit user program (Substitute)	1- 8
SO*) Enter/edit user program without echo	1- 9
SP Save PLC program in Flash EPROM	1- 9
V Move user program	1- 9

*) **not** with 07 KR 31 / 07 KT 31
) **only with 07 KR 31 / 07 KT 31

Commands for testing the user program (overview)

Command	Function	Page
A	Abort user program	1– 9
BA*)	Display breakpoints	1–10
BR*)	Reset breakpoints	1–10
BS*)	Set breakpoints	1–10
<CTRL>W*)	Change-over between operator control functions <—> monitor	1–10
EA*)	I/O test mode	1–11
EAA*)	Deactivate I/O test mode	1–11
ES*)	Single-step mode ON	1–11
ESA*)	Single-step mode OFF	1–11
EZ*)	Single-cycle mode ON	1–11
EZA*)	Single-cycle mode OFF	1–12
FEHLER	Display contents of the error register	1–12
FORC	Enter Force values	1–12
FORC A	Display Force value	1–13
FORC R	Delete Forcing	1–13
G	Start user program	1–13
KALT	Perform cold-start	1–14
WARM	Perform warm start	1–14
L*)	Continue user program	1–14
PS	Display program status	1–14
ST	Display PLC status	1–14
TRACE*)	TRACE mode	1–15
TRACE*)	Display TRACE memory	1–15
TRACE E*)	Activate TRACE mode	1–15
TRACE A*)	Deactivate TRACE mode	1–15
W*)	Stop user program	1–15
Y	Overwrite value of a variable with a value to be entered	1–15
Z	Display status of variables	1–15
ZZ	Display only the values of the variables	1–16
ZD	Display and continually update status of variables	1–16

Commands for configuring

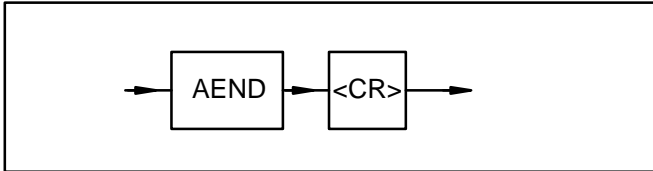
Command	Function	Page
KONFS	Display/change operating modes	1–17
MAIL	Configuration of CS31 remote modules	1–17
PASS **)	Activate / deactivate the password ..	1–21
UHR	Display time and date	1–21
UHRS	Set time and date	1–22

*) **not** with 07 KR 31 / 07 KT 31
) **only with 07 KR 31 / 07 KT 31

1.1 Commands for creating the user program

Prepare 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.

Reject a program change which has not yet been enabled

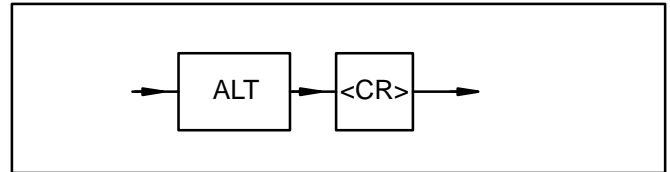
Entering the AEND command again rejects all program modifications performed to date, and the PLC is ready to accept program modifications again.

The following commands are activated with the program **running, and also** 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. Command AEND must be entered again in order to permit you to perform program modifications again.

Reject an enabled program change on a running PLC program and reactivate the old program status

Command:



Function:

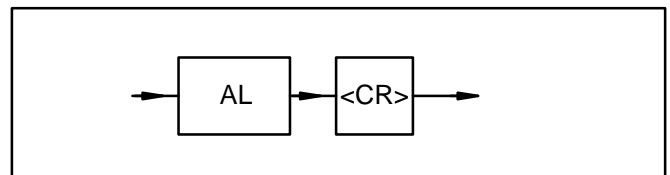
Modifications which have been performed on a running PLC program **and** which have been enabled are rejected again. In addition, the PLC restores the old program status. The old program status is the status of the program which existed before the program modification, i.e. before entry of command AEND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

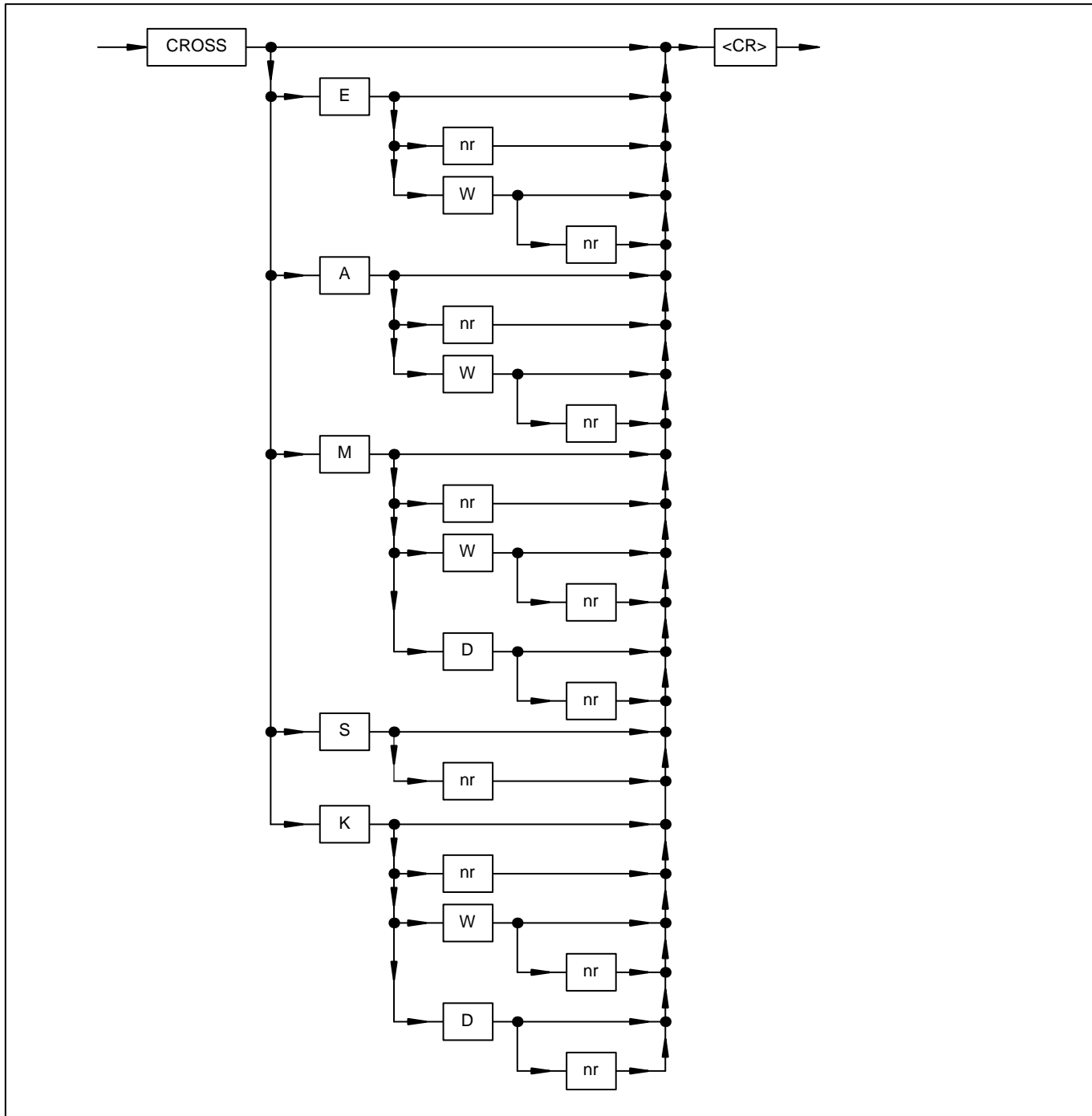
The processor capacity which corresponds to the difference between 100 % and the capacity utilization display is available for operation of the serial interfaces, i.e. for communication with the devices connected to the serial interfaces. The utilization should not be greater than 95 % for the longest 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 branches and consecutive number blocks).

Note:

The capacity utilization display provides a correct indication of the utilization caused by the user program only if *no communication* is occurring via the serial interfaces at the instant of display.

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 output for

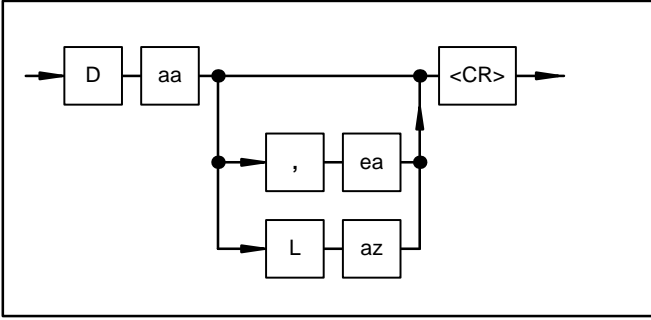
- D all operands occurring in the program, entry: CROSS <CR>
- D a specific operand type, entry, e.g.: CROSS E <CR>
- D a single operand, entry, e.g.: CROSS KD 00,12 <CR>

Note on service device TCZ:

The four-line liquid-crystal display (LCD) of the service device TCZ does not suffice to display this command.

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:

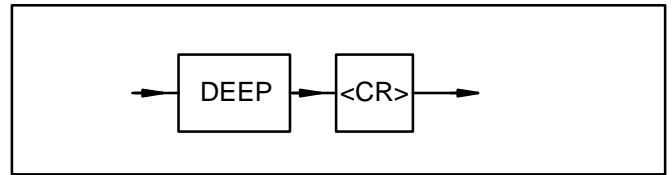
```
0000 !E 00,00
0002 &E 00,01
0004 =A 00,00
0006 !BA001
0007 AWT
0008 A 00,00
0009 KW 00,00
0010 KW 00,01
0011 AW 00,00
```

Note on service device TCZ:

This command can be used only with the following restriction: A maximum of three instructions can be displayed on the liquid-crystal display.

Erase PLC program on Flash EPROM

Command:

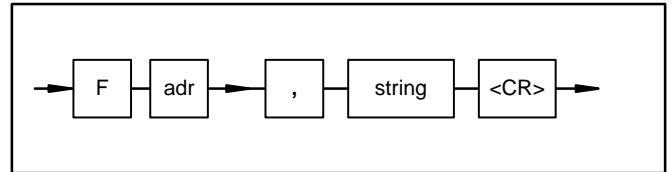


Function:

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

Search for string in 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>
```

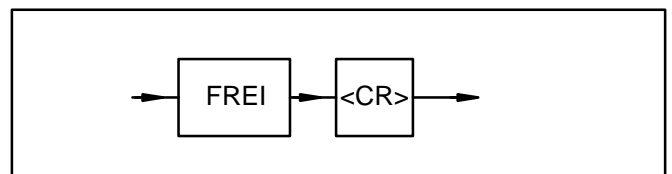
The entered string is sought as of the program memory start address 0.

```
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:

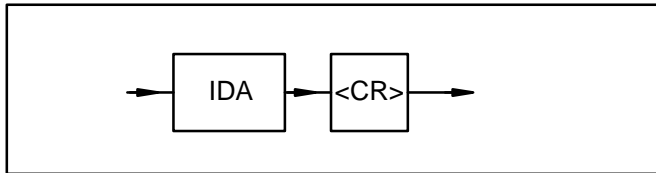
The modifications on a running PLC program performed after entry of command AEND are enabled for execution.

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

After entry of command FREI, the modifications performed 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 further program modification.

Display program identification

Command:

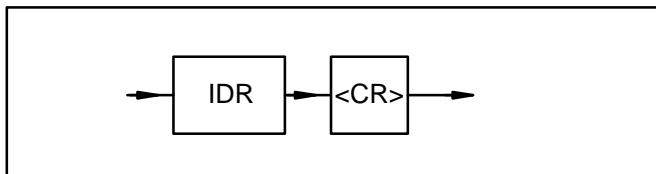


Function:

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

Delete program identification

Command:

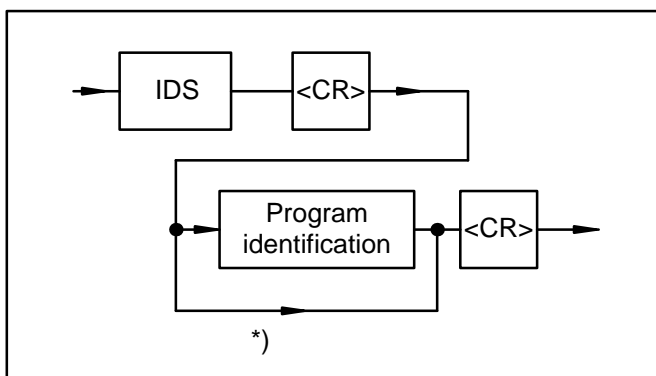


Function:

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.

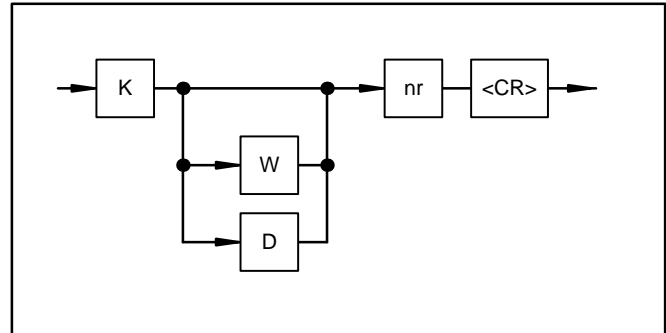
*) No program identification is entered for this path. An already existing program identification is deleted.

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

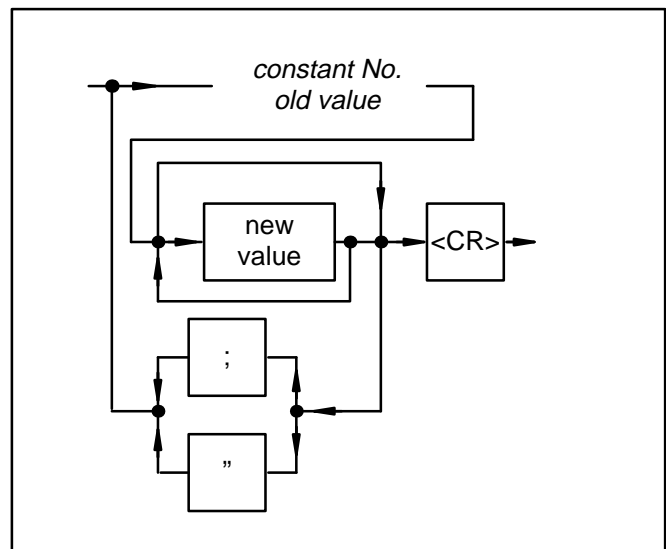
Command:



W: Abbreviation for word constants

D: Abbreviation for double–word constants

nr: Entered number of the constant



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 display of number and value of the constant with the next number down. If 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 integral multiple of the basic time of 5 ms, i.e. 5 ms, 10 ms, 15 ms etc.

Example:

K 0,0 <CR>

Output of the number and value of the binary constant 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>

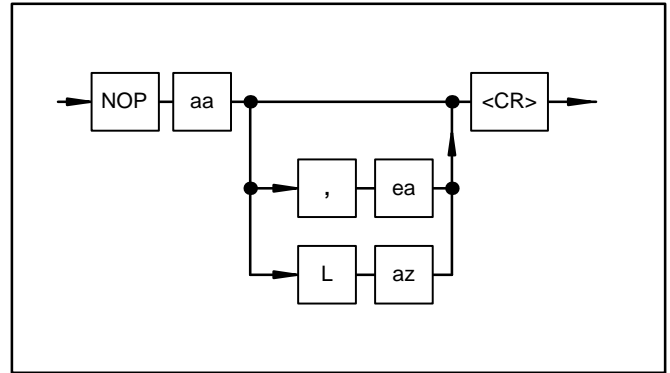
Output of the number and value of the word constant KW 00,04.

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 program part 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. A prompt is displayed in order to establish whether you really do want to delete this program part before deletion. The user must once again either confirm deletion with "J" or cancel deletion with "N".

Example:

NOP 0,20 <CR>

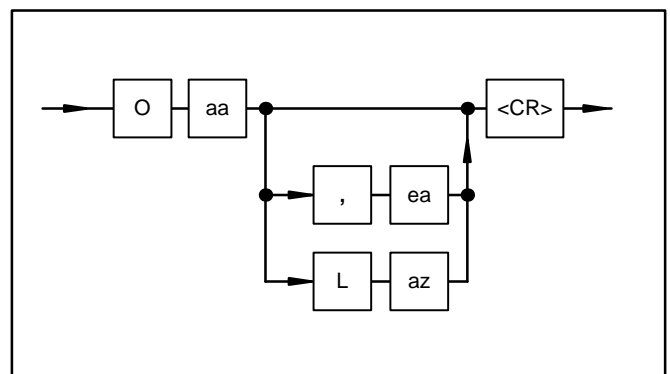
The user program is deleted from address 0 through to address 20.

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 and the program is compressed in the given program part.

Example:

O 0 <CR>

The entire program memory is optimized.

O 0,10 <CR>

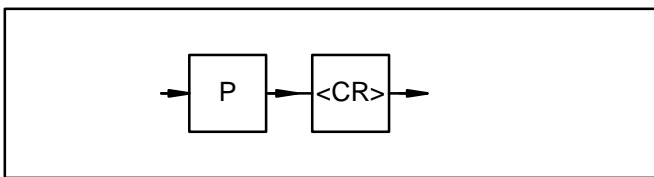
The program memory is optimized as of address 0 through to address 10.

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 area

Command:

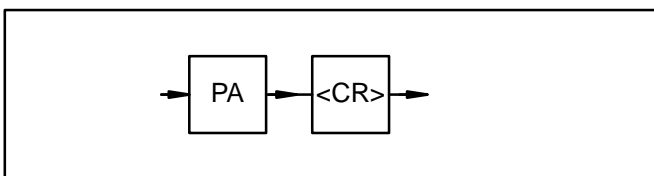


Function:

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

Program preparation

Command:

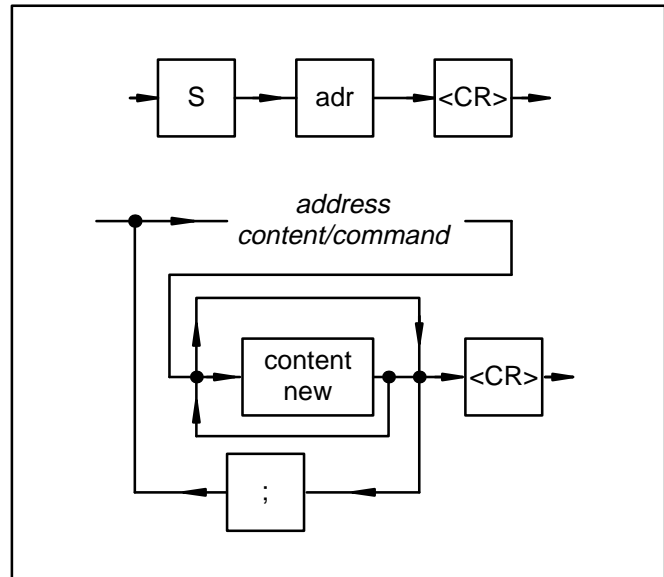


Function:

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 conducted 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.

content new: 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.

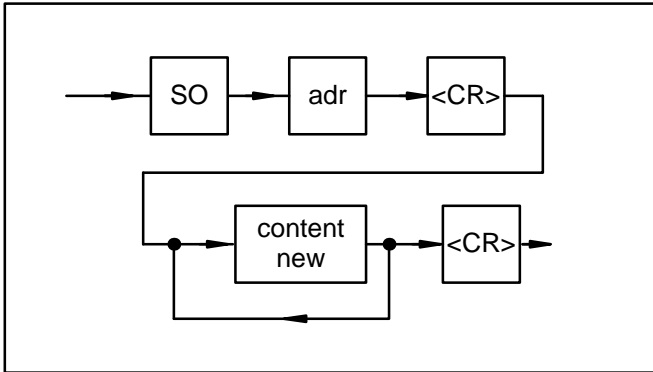
Note:

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

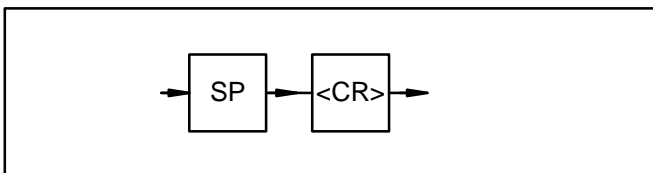
content new: New content of the user program

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

Command:

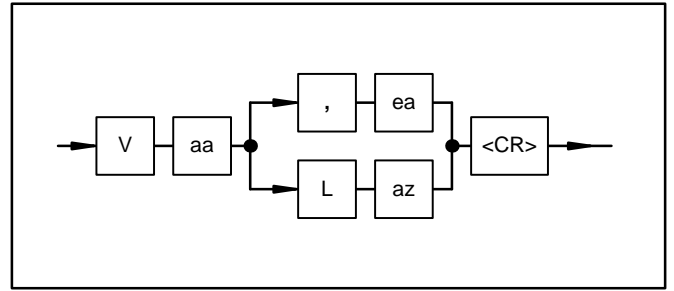


Function:

The PLC program is transferred from the RAM to the Flash EPROM. 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.

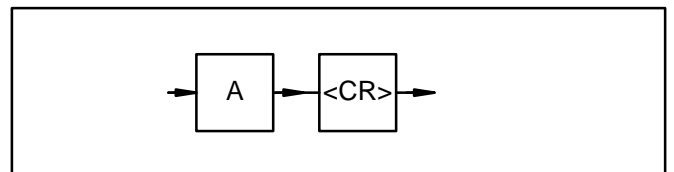
V 10 L 20 <CR>

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

1.2 Commands for testing the user program

Abort user program

Command:



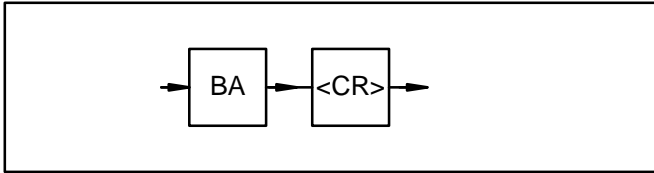
Function:

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 breakpoints

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.

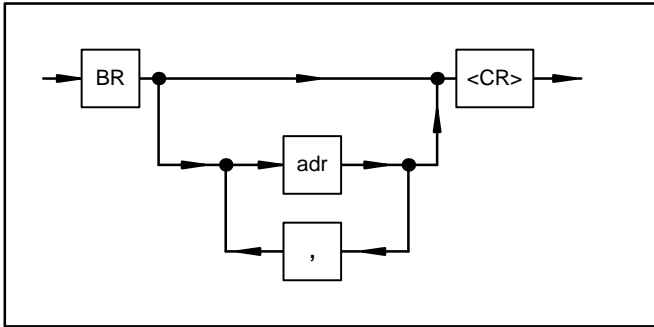
Note on service device TCZ:

This command can be used only with the following restriction:

A maximum of three program memory points at which breakpoints are set can be displayed on the liquid-crystal display (LCD).

Reset breakpoints

Command:



adr: Address of the breakpoint to be reset

,: If only specific breakpoints are reset, the individual addresses must be separated by a comma when entering.

Function:

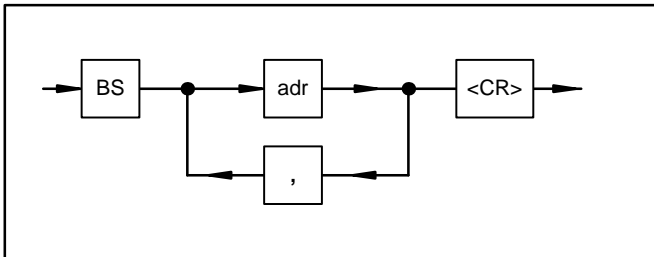
The breakpoints can be reset individually. Command

BR <CR>

resets all 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.

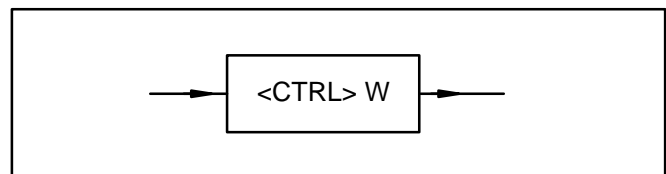
Advancing to the next breakpoint: If a *semicolon* is entered, the program runs to the next breakpoint after expiry of the cycle time and displays the program address and the command at this address. If the next breakpoint is *not* reached after a specific period, owing to a long cycle time, the display 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 branch, 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:

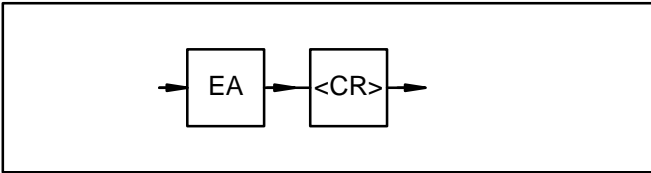


Function:

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 (see also chapter Monitor functions)

I/O test mode

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.

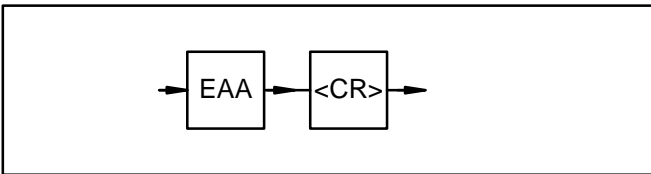
After starting the user program, it is *not* executed. Only the I/O signals planned in the program are operated, i.e. the input signals are read in and the output signals are brought out.

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:

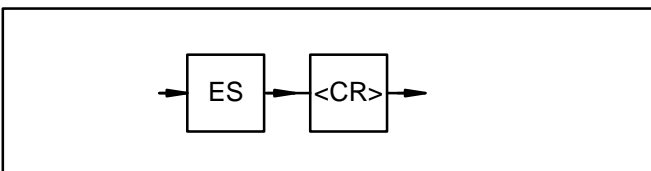


Function:

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.

Single-step mode ON

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.

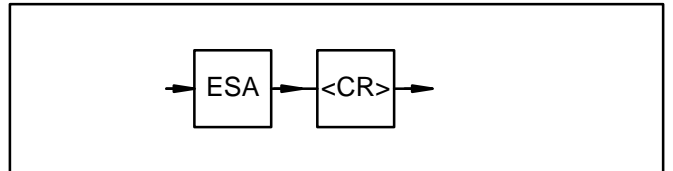
Command "ES" can also be entered with the program running. In this case, the mode does not take effect until the start of the next program cycle.

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.

Single-step mode OFF

Command:

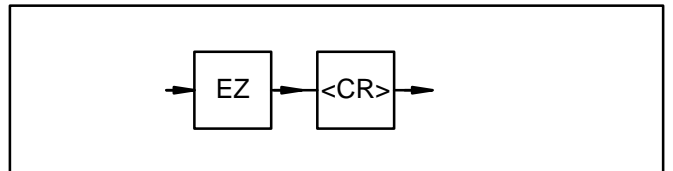


Function:

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

Single-cycle mode ON

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.

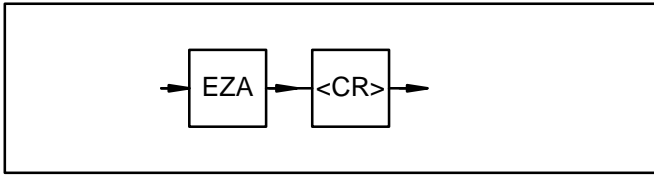
The mode does not come into effect until the start of the next program cycle.

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.

Single-cycle mode OFF

Command:

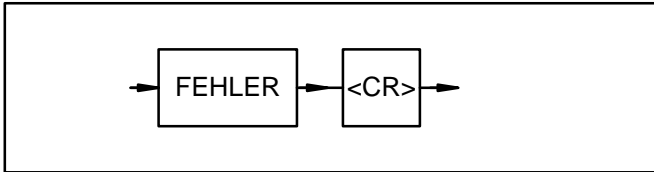


Function:

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

Display contents of the error register

Command:



Function:

The error information stored in the PLC is output.

Enter Force values

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:

- Binary inputs: 64
- Word inputs: 16
- Binary 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.

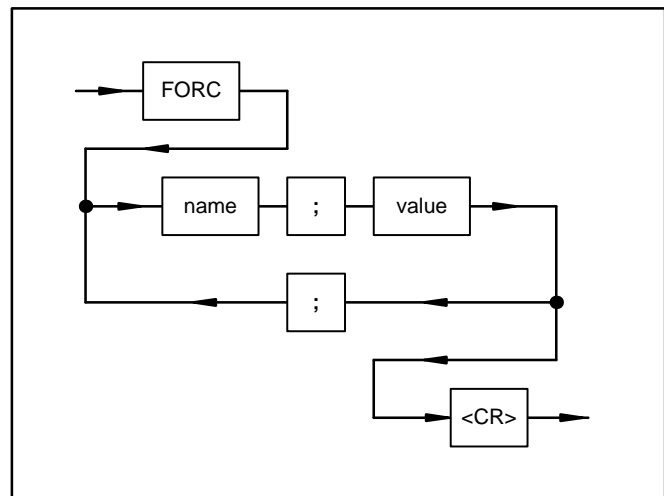
The following commands are available for forcing I/O signals:

- 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 command FORC.

Command: FORC Enter force value



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

value: Force value for the input or output

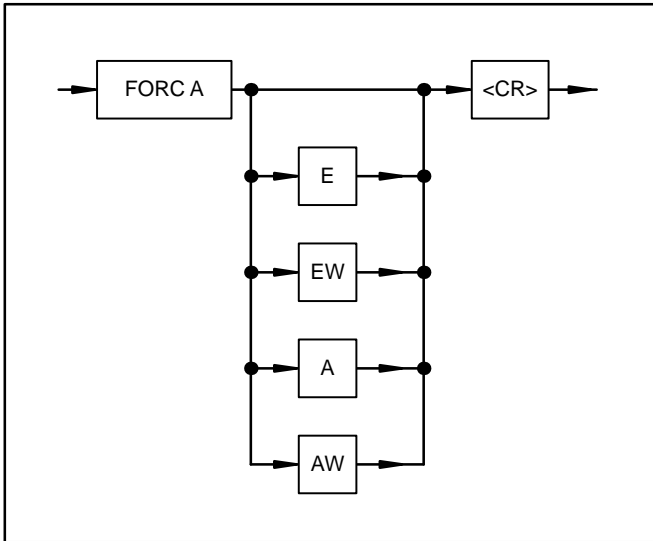
:: A semicolon is used as the separator between the name and the force value. If several inputs/outputs are to be forced, they must also be separated by a semicolon.

Function:

Entering the I/O signals to be forced and their values. The list specifying which inputs/outputs are to be forced is stored power-fail-safe in the operand memory of the PLC (if a battery is fitted).

Display force value

Command:



Function:

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

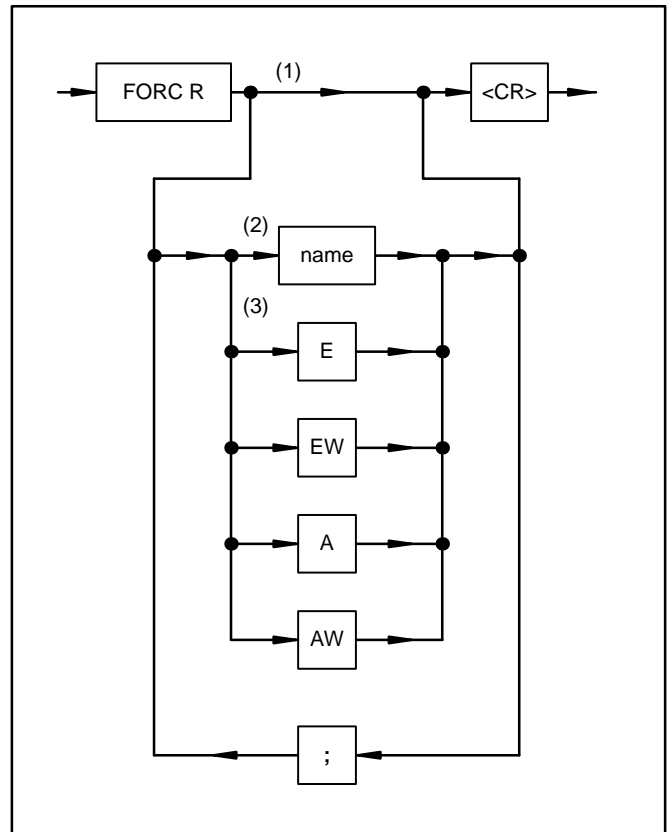
Note on service device TCZ:

This command can be used only with the following restriction:

A maximum of three I/O signals with related force values can be displayed on the liquid-crystal display (LCD).

Delete forcing

Command:



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

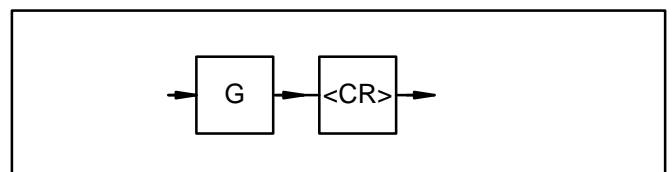
:: If forcing is terminated only for specific inputs/outputs, the individual names must be separated by a semicolon when entering them.

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:



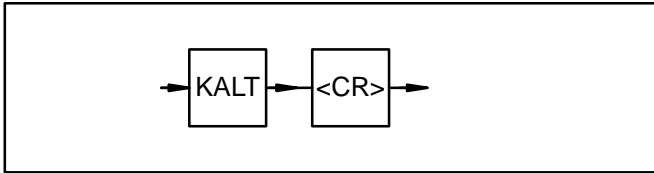
Function:

The user program is started and the operands are initialized.

The operand areas are initialized in accordance with the corresponding system constant.

Perform cold-start

Command:



Function:

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

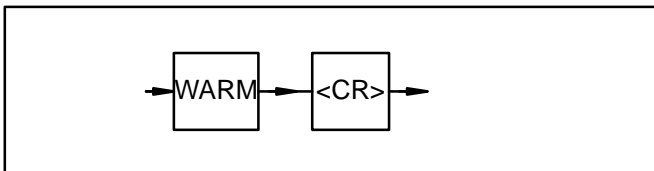
- All RAM memories will be tested and deleted.
- If there is *no user program* in the Flash EPROM, the default values will be set to all system constants (same as factory setting).
- If there is *a user program* in the Flash EPROM, this will be stored in the RAM inclusive the system constants.
- The operating modes defined by the system constants will be adjusted.
- The CS31 system bus will be initialized again (only in case of CS31 system bus master)

Performing a cold start

- Command KALT <CR> in terminal mode or
- Voltage OFF/ON, when *no* battery is existing or
- menu item "Kaltstart" in the programming system

Perform warm start

Command:



Function:

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

Warm start

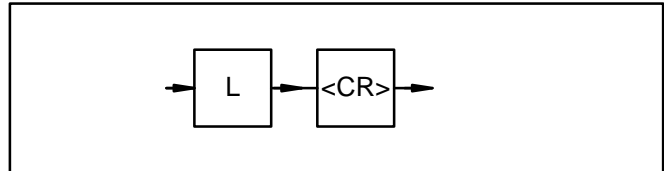
- All RAM memories will be tested and deleted with the exception of program memories and operand memories.
- If there is *a user program* in the Flash EPROM, this will be stored in the RAM inclusive the system constants.
- The operating modes defined by the system constants will be adjusted.
- The CS31 system bus will be initialized again (only in case of CS31 system bus master)

Performing a warm start

- Command WARM <CR> in terminal mode or
- Voltage OFF/ON, if a battery is existing or
- menu item "Enable PLC mode" in the programming system

Continue user program

Command:



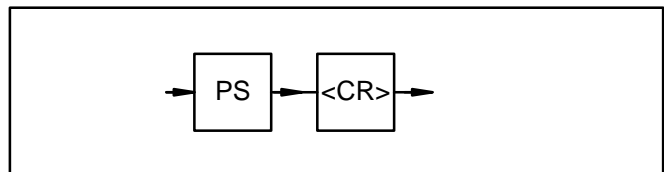
Function:

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:

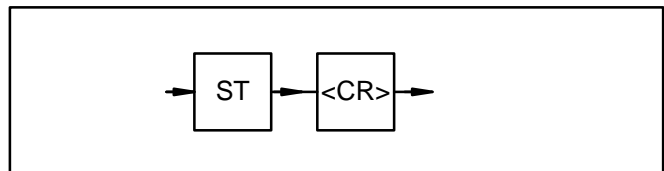


Function:

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

Display PLC status

Command:



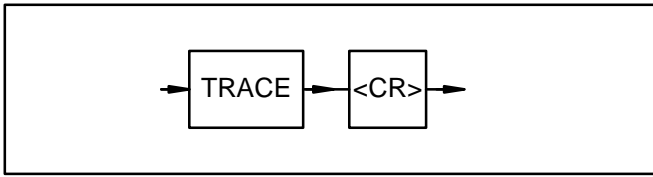
Function:

The entire PLC status is displayed as follows:

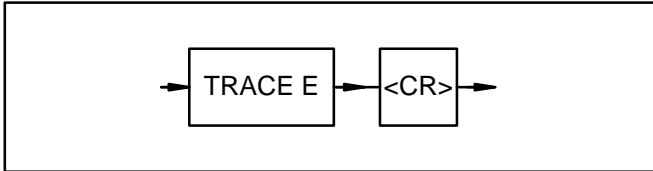
- Program identification
- Cycle time
- Program status
- Active test functions
- TRACE registers
- Error messages
- Capacity utilization

TRACE mode

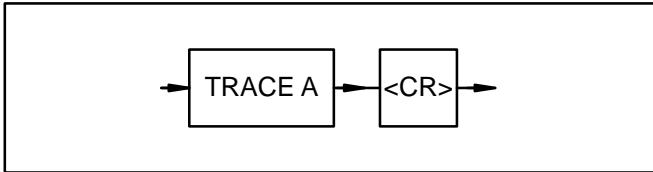
Command: Display TRACE memory



Command: Activate TRACE mode



Command: Deactivate TRACE mode

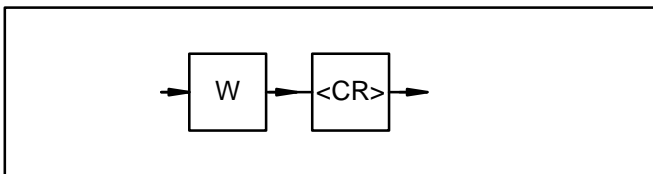


Function:

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:



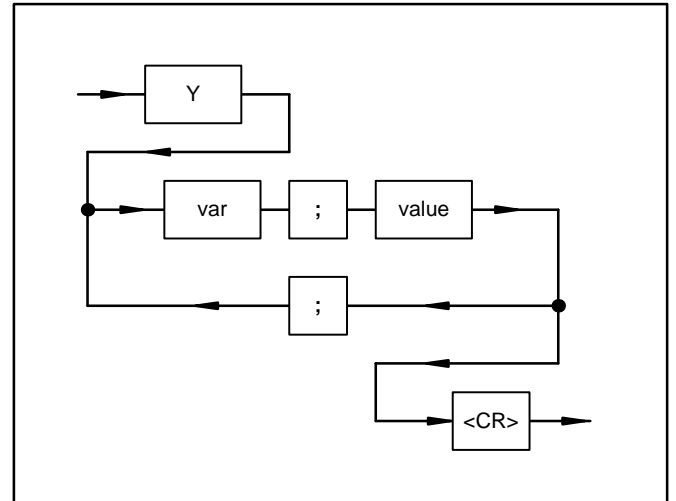
Function:

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 a 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.

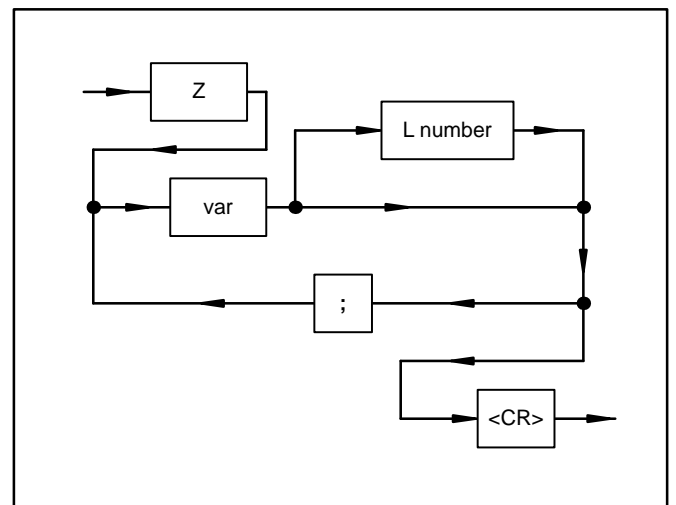
Note:

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

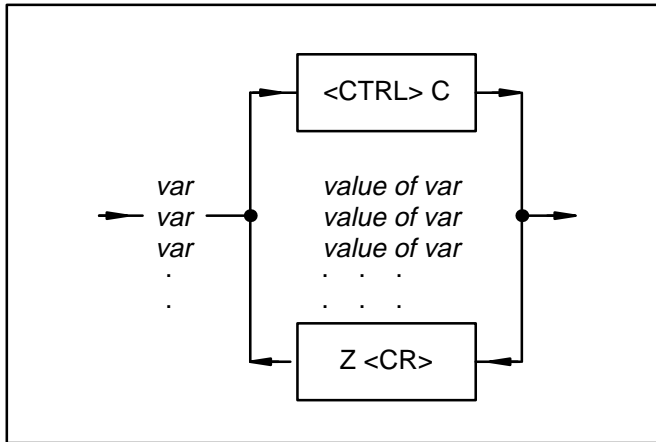
Command:



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 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.

Note on service device TCZ:

This command can be used only with the following restriction: Only the status of one variable is displayed.

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.

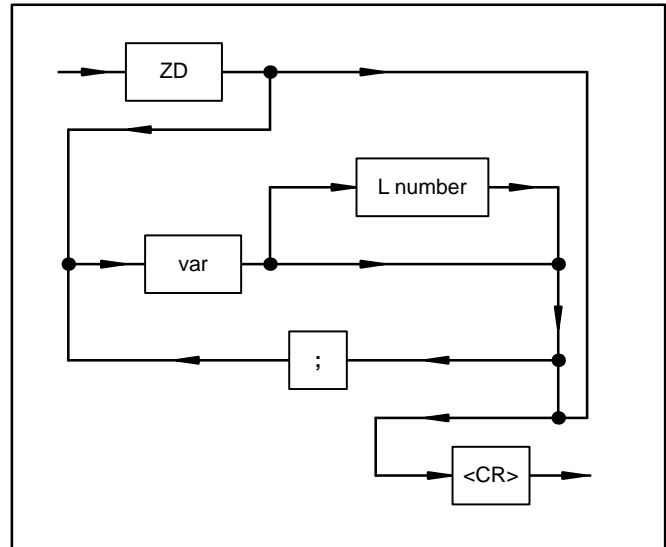
Screen control: In the case of commands Z, ZO and ZD, the following control characters are used by the PLC for screen control:

- 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:



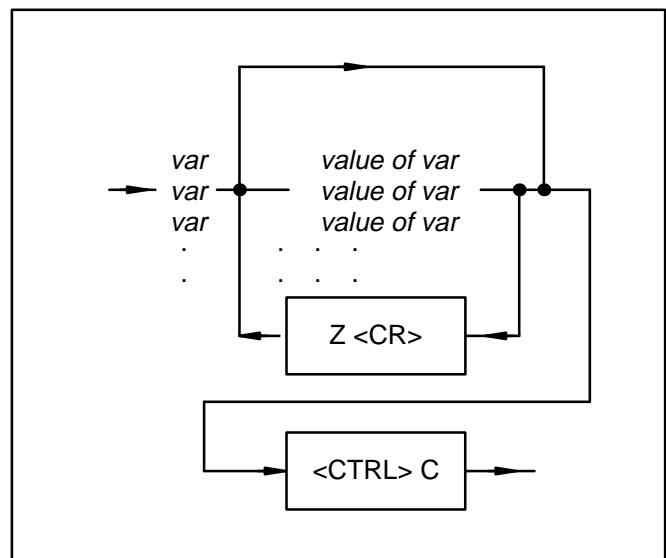
var: Variable (flag, input, output, indirect constant) to be displayed

:: 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.

Note on service device TCZ:

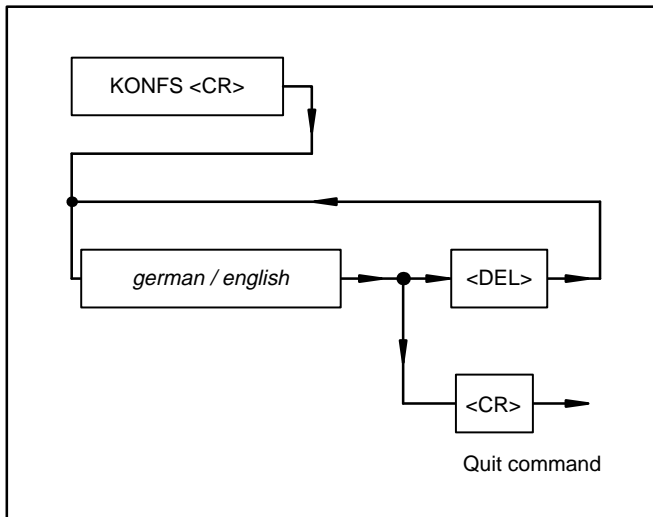
This command can be used only with the following restriction:

Only the status of one variable is displayed.

1.3 Commands for configuring

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:

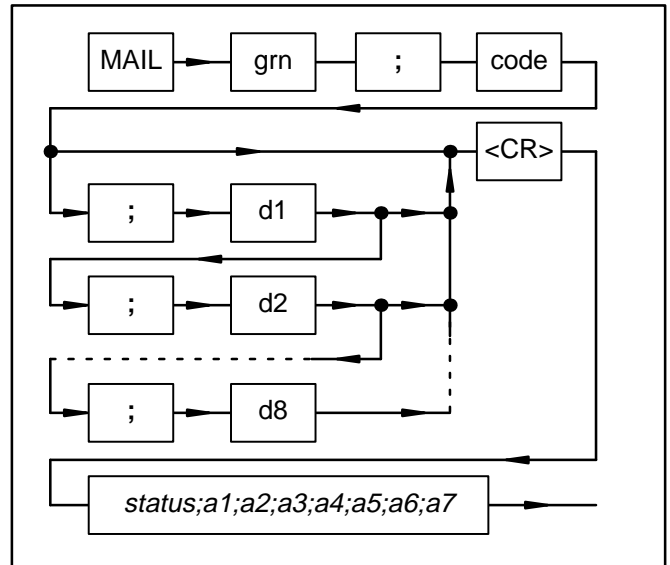
The DELETE key is frequently not available on personal computers. The key code (7FH) of the DELETE key can be generated on such keyboards by pressing two keys. In general, these keys are keys <CTRL> and the Backspace key.

Note on service device TCZ:

On the servicing unit TCZ, you switch over by pressing keys <CTRL> and one after the other.

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

Command:



grn: Group number with which the local 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 response 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 relevant jobs.

The not-OK responses of the individual jobs are always 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 not free within 3 sec.
 - 11 = Job has been aborted owing to
actuation 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, bits 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 deactivating 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

- **Acknowledging error on remote module**

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

- **Job**

- grn: Group number 0...63
 - code: 232
 - 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 = Binary input
 - 1 = Analog input
 - 2 = Binary output
 - 3 = Analog output
 - 4 = Binary 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

- **Acknowledging errors on remote module and resetting configuration values to default setting**

In addition to job "Acknowledging error on remote module", all configurable settings are reset to the 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 = Binary input
 - 1 = Analog input
 - 2 = Binary output
 - 3 = Analog output
 - 4 = Binary 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

- **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:* 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 \pm 10V
a2...a7: 0
- **Configuration of an analog input**
 - **Job**
grn: Group number 0...63
code: 170
d1: Channel number 0...15
d2: 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 \pm 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:
Bits 0...3: Number of process data bytes (binary module) or words (word module) sent by the module to the Master.
Bits 4...7: Number of process data bytes (binary module) or words (word module) sent by the Master to the module.
a2: Group number (0...63)
- **Reading 1...6 bytes (07 KR 91, 07 KT 92, 07 KT 93)**
 - **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 = Binary input
1 = Analog input
2 = Binary output
3 = Analog output
4 = Binary 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 1 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 = Binary input
1 = Analog input
2 = Binary output
3 = Analog output
4 = Binary 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...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)**

• **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 = Binary input
 1 = Analog input
 2 = Binary output
 3 = Analog output
 4 = Binary 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 = Binary input
 1 = Analog input
 2 = Binary output
 3 = Analog output
 4 = Binary input/output
 5 = Analog input/output

Note:
 Bit: even number (0, 2, 4)
 Word: odd number (1, 3, 5)

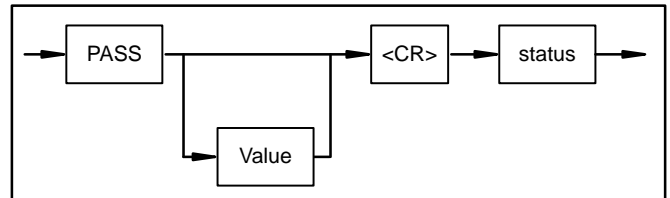
d3: Byte start address (low byte)
 d4: Byte start address (high byte)
 d5: Bit position within byte 0...7
 d6: Bit value (0 or 1)
 d7...d8: Not used

• **OK response**

status: 51
a1...a7: 0

Password (only with 07 KR 31 / 07 KT 31)

Command:



Function:

The command PASS value activates or deactivates the password. As a password, any 4-digit hexadecimal number (except 0000) can be used. If a password is activated, the following commands are disabled: AEND, D, DEEP, FREI, N, NOP, O, S, V.

Value: Any 4-digit hexadecimal number.

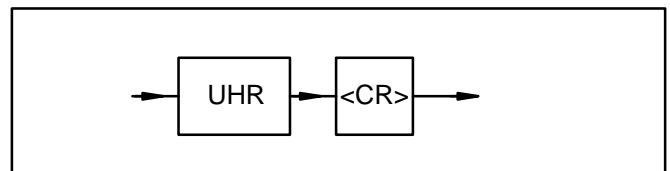
Caution: The value of 0000 has no effect.

status: The activation or deactivation of the password is displayed.

Display time and date

(07 KR 91, 07 KT 92, 07 KT 93)

Command:



Function:

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

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

where:

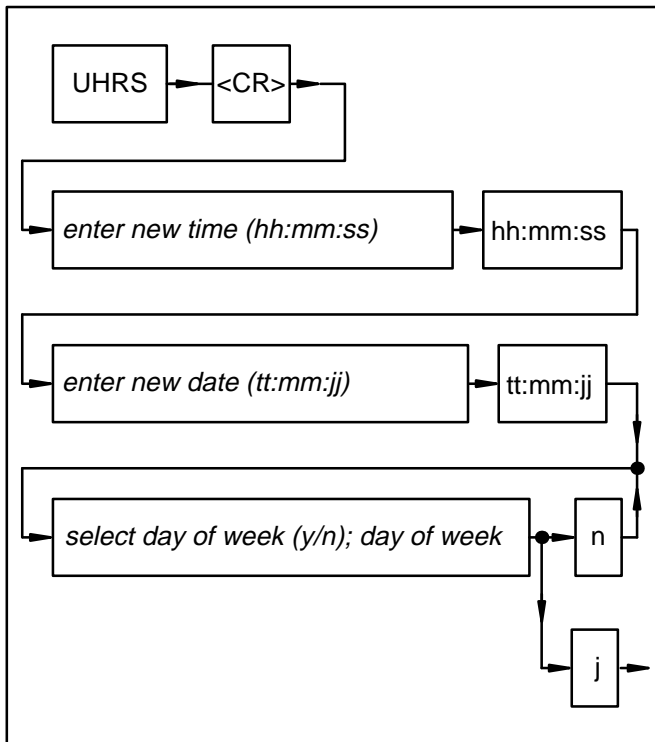
HH: Hours
 MM: Minutes
 SS: Seconds
 DAY OF WEEK: Name of the day of the week
 TT: Day
 MM: Month
 JJ: Year

Note on service device TCZ:

The four-line liquid-crystal display (LCD) of the servicing unit TCZ does not suffice to display this command.

Set time and date

Command:



Function:

Setting the time and date. For the day of the week, the clock manages 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 or *hh*: Hours
mm or *mm*: Minutes
ss or *ss*: Seconds
tt or *tt*: Day
mm or *mm*: Month
jj or *jj*: Year
day of week: Name of the day of the week
n: Enter for 'no'
j: Enter for 'yes'

1.4 Texts in the instruction list

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

Entering the texts in the user program

A text is entered with the terminal or service device TCZ 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, how they are entered and how they are displayed on a monitor

ASCII char-acter	Hex-value	User entry	Display	ASCII char-acter	Hex-value	User entry	Display	ASCII char-acter	Hex-value	User entry	Display
NUL	00	<CTRL><SP>	<NUL> *					b	62	b	b
SOH	01	<CTRL>	A <SOH>					c	63	c	c
STX	02	<CTRL>	B <STX>					d	64	d	d
ETX	03	<CTRL>	C <ETX>					e	65	e	e
EOT	04	<CTRL>	D <EOT>					f	66	f	f
ENQ	05	<CTRL>	E <ENQ>					g	67	g	g
ACK	06	<CTRL>	F <ACK>					h	68	h	h
BEL	07	<CTRL>	G <BEL>					i	69	i	i
BS	08	<CTRL>	H <BS>					j	6A	j	j
HT	09	<CTRL>	I <HT>	:	3A	:	:	k	6B	k	k
LF	0A	<CTRL>	J <LF>	;	3B	;	;	l	6C	l	l
VT	0B	<CTRL>	K <VT>	<	3C	<	<	m	6D	m	m
FF	0C	<CTRL>	L <FF>	=	3D	=	=	n	6E	n	n
CR	0D	<CTRL>	M <CR>	>	3E	>	>	o	6F	o	o
SO	0E	<CTRL>	N <SO>	?	3F	?	?	p	70	p	p
SI	0F	<CTRL>	O <SI>	@	40	@	@	q	71	q	q
DLE	10	<CTRL>	P <DLE>	A	41	A	A	r	72	r	r
DC1	11	<CTRL>	Q <DC1>	B	42	B	B	s	73	s	s
DC2	12	<CTRL>	R <DC2>	C	43	C	C	t	74	t	t
DC3	13	<CTRL>	S <DC3>	D	44	D	D	u	75	u	u
DC4	14	<CTRL>	T <DC4>	E	45	E	E	v	76	v	v
NAK	15	<CTRL>	U <NAK>	F	46	F	F	w	77	w	w
SYN	16	<CTRL>	V <SYN>	G	47	G	G	x	78	x	x
ETB	17	<CTRL>	W <ETB>	H	48	H	H	y	79	y	y
CAN	18	<CTRL>	X <CAN>	I	49	I	I	z	7A	z	z
EM	19	<CTRL>	Y 	J	4A	J	J	{	7B	{	{
SUB	1A	<CTRL>	Z <SUB>	K	4B	K	K		7C		
ESC	1B	<CTRL>	[<ESC>	L	4C	L	L	}	7D	}	}
FS	1C	<CTRL>	/ <FS>	M	4D	M	M	~	7E	~	~
GS	1D	<CTRL>] <GS>	N	4E	N	N	DEL	7F		
RS	1E	<CTRL>	~ <RS> *	O	4F	O	O				
US	1F	<CTRL>	? <US> *	P	50	P	P				
SP	20	<SP>	<SP> **	Q	51	Q	Q				
!	21	!	!	R	52	R	R				
"	22	"	"	S	53	S	S				
#	23	#	#	T	54	T	T				
\$	24	\$	\$	U	55	U	U				
%	25	%	%	V	56	V	V				
&	26	&	&	W	57	W	W				
'	27	'	'	X	58	X	X				
(28	((Y	59	Y	Y				
)	29))	Z	5A	Z	Z				
*	2A	*	*	[5B	[[
+	2B	+	+	\	5C	\	\				
,	2C	,	,]	5D]]				
-	2D	-	-	↑	5E	↑	↑				
.	2E	.	.	—	5F	—	—				
/	2F	/	/	'	60	'	'				
0	30	0	0	a	61	a	a				

* To older terminals applies:

NUL 0 <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> in order to permit it to be recognized easier.

Note:

Interrelationship between value and ASCII character

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

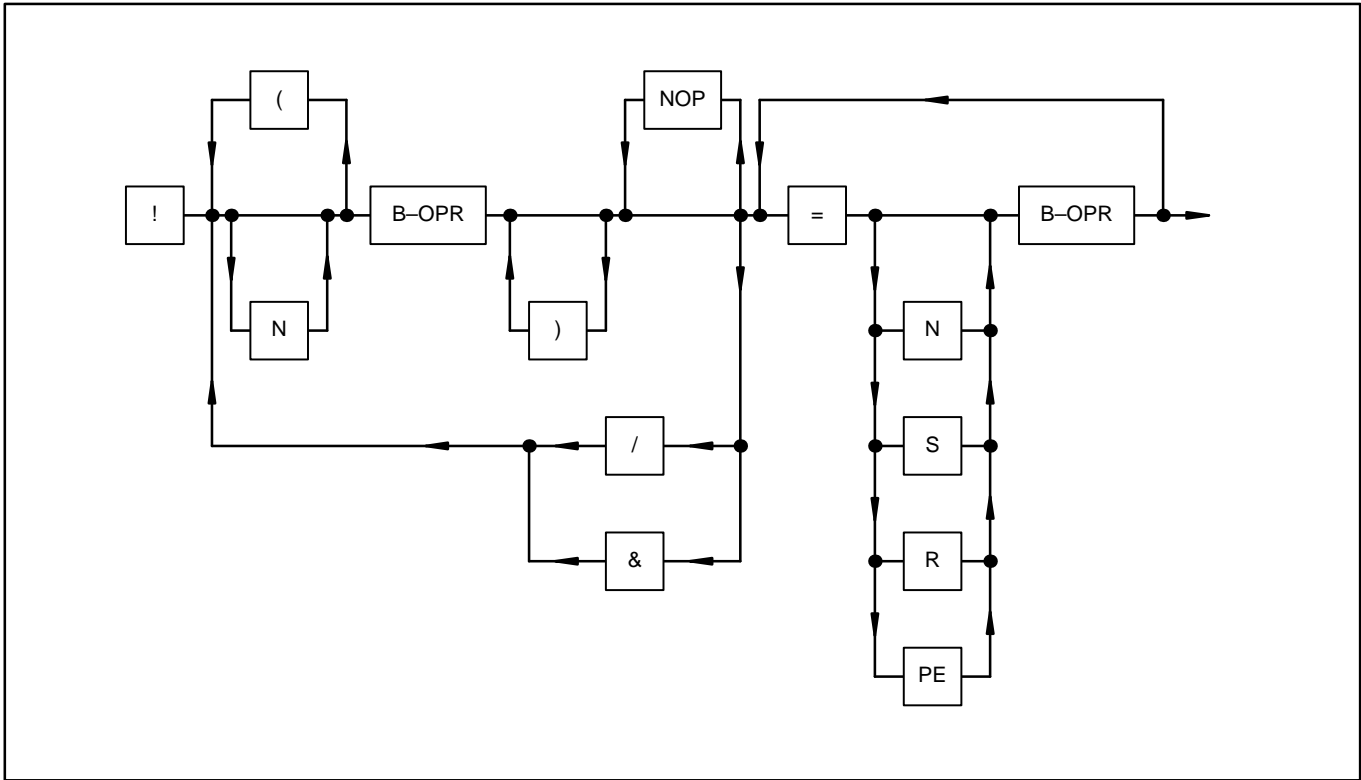
Represented as:

- Decimal ASCII: 31_H, 31_H
- Hexadecimal ASCII: 42_H

Output by DRUCK as hexadecimal value: 1011

1.5 Syntax diagrams for instruction list (IL)

1.5.1 Syntax diagram: BOOLEAN SENTENCE

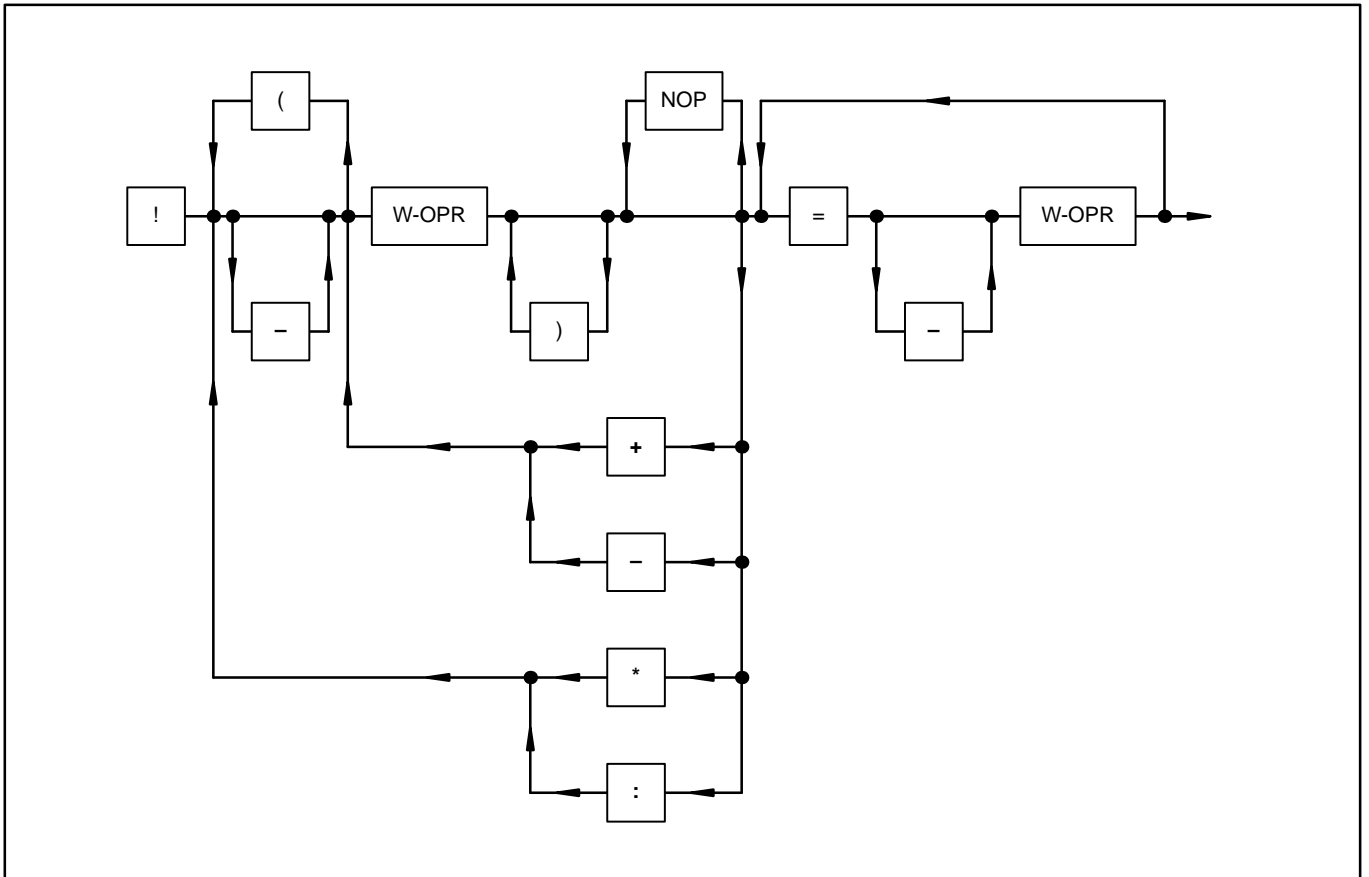


Signal flow: in the direction of arrow, otherwise from left to right.

Brackets: sum "LEFT BRACKET" = sum "RIGHT BRACKET", nesting depth: 15.

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

1.5.2 Syntax diagram: ARITHMETIC SENTENCE



Signal flow: in direction of arrow, otherwise from left to right.

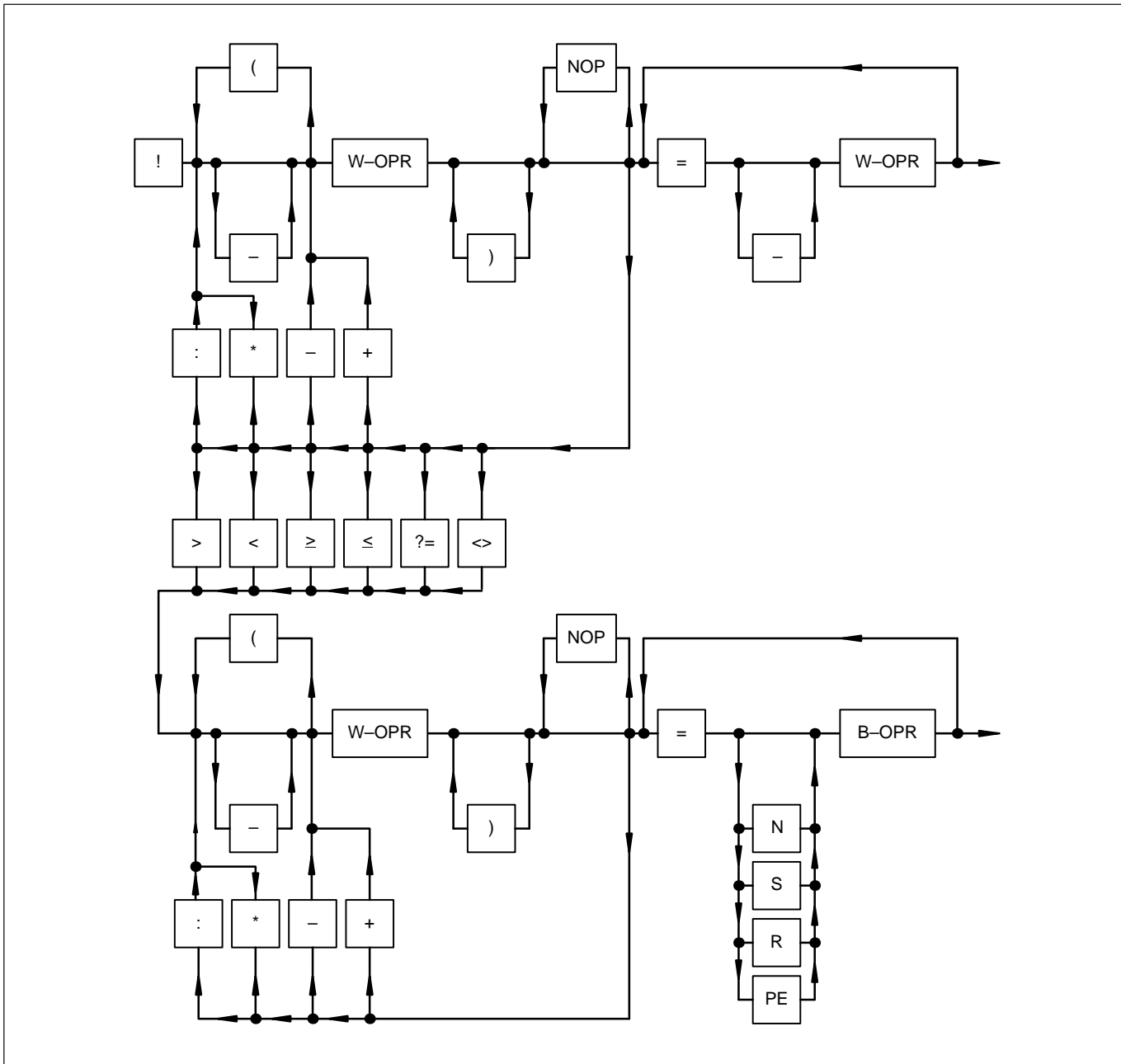
Brackets: sum "LEFT BRACKET" = sum "RIGHT BRACKET", nesting depth: 15.

W-OPR: Word operand (EW, AW, MW, KW)

Example: EW 03,05 AW 11,12 MW 22,15 KW 09,06

1.5.3 Syntax diagram: HYBRID SENTENCE

See also chapter "Language repertoire", Relational operators



Signal flow: in direction of arrow, otherwise from left to right.

Brackets: sum "LEFT BRACKET" = sum "RIGHT BRACKET", nesting depth: 15.

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

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

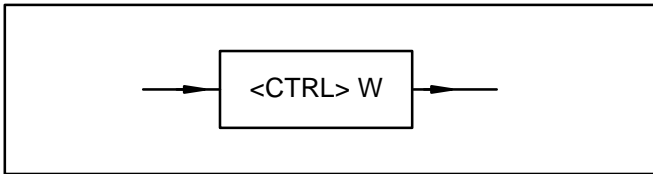
2 Monitor functions

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 areas may endanger the functionality of the PLC. For this reason, take care when using the monitor functions.

Switchover Operator-control functions \longleftrightarrow Monitor functions

Command:



Not available in 07 KR 31 / 07 KT 31

Function:

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

Explanation of the syntax:

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

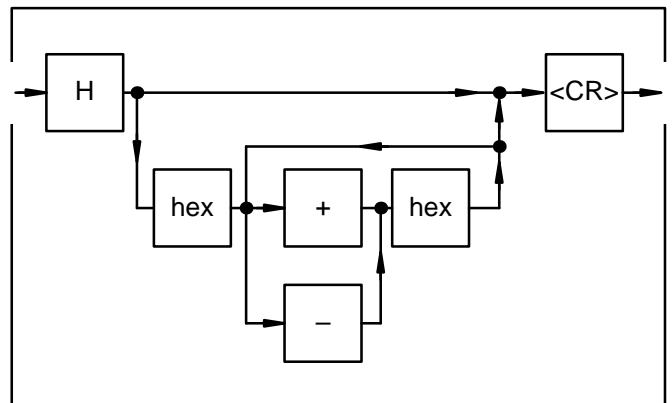
Overview of the monitor functions

Function	Explanation	Page
H	Display help text / calculate hexadecimally	2-1
D	Display memory contents	2-2
I	Fill memory area with a value	2-2
M	Transfer memory areas	2-3
P	Read/write port	2-3
S	Display/edit memory contents	2-4
U/V	Edit address output format	2-4
Y	Display/edit working segment	2-4
ZA	Cyclic read and write	2-4
ZB	Cyclic read and write with waiting time	2-5
ZC	Read and write on keystroke	2-5
ZD	Cyclic write	2-5
ZE	Cyclic read	2-5
ZF	Cyclic write and read	2-5
ZG	Simultaneous output of 3 values	2-6
ZR	RAM test	2-6
ZZA	Output of 3 values after entering a semicolon (;)	2-7
ZZF	Search for string	2-7
ZZV	Compare memory areas word-serially	2-8
R	Read Intel HEX file	2-9
W	Write Intel HEX file	2-9

Display help text / calculate hexadecimally

When command H <CR> is entered, all available functions of the monitor are displayed on the screen. In addition, this command permits you to calculate simple hexadecimal arithmetic expressions.

Command:

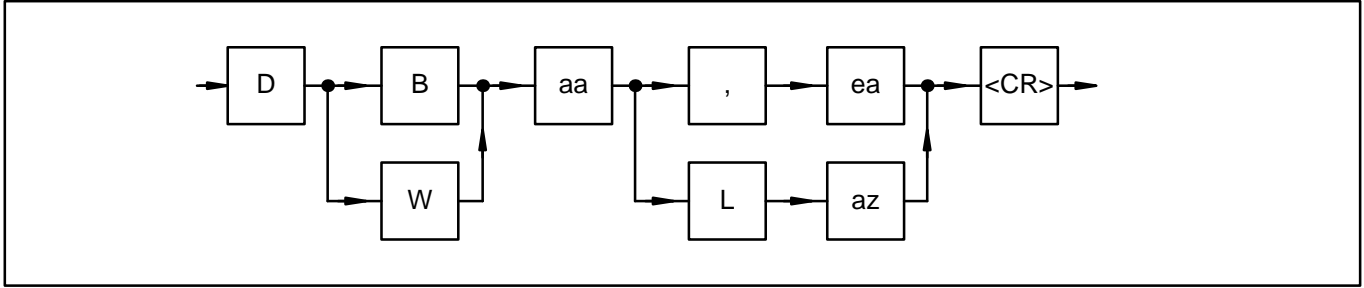


hex: Hexadecimal number

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

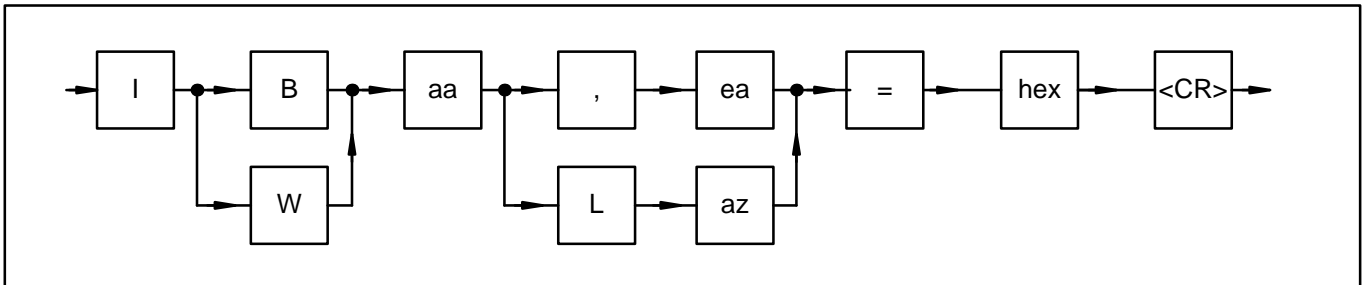
0000:0000 02 00 Monitor display

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

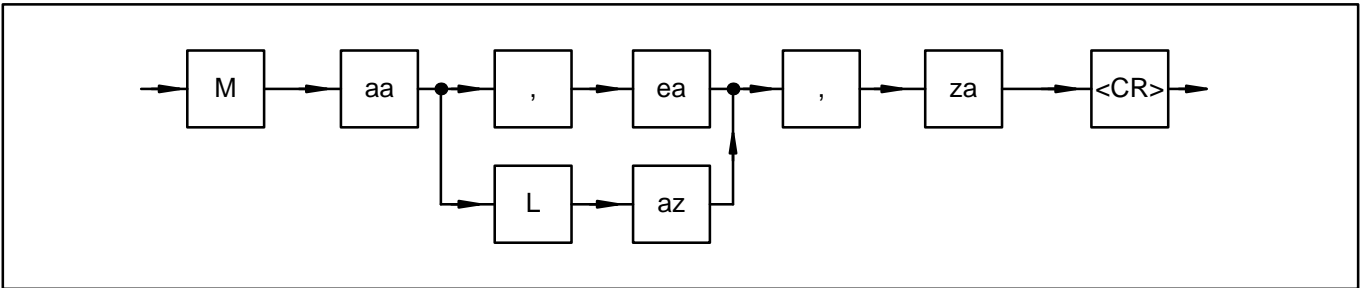
Example:

IB 8000:80L3=FF<CR> The memory contents of 8000:80_H, 8000:81_H and 8000:82_H is overwritten with FF

Transfer 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. one word is transferred in the case of az = 3).

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 are copied:

8000H:80H —> 8000H:90H

:81H —> :91H

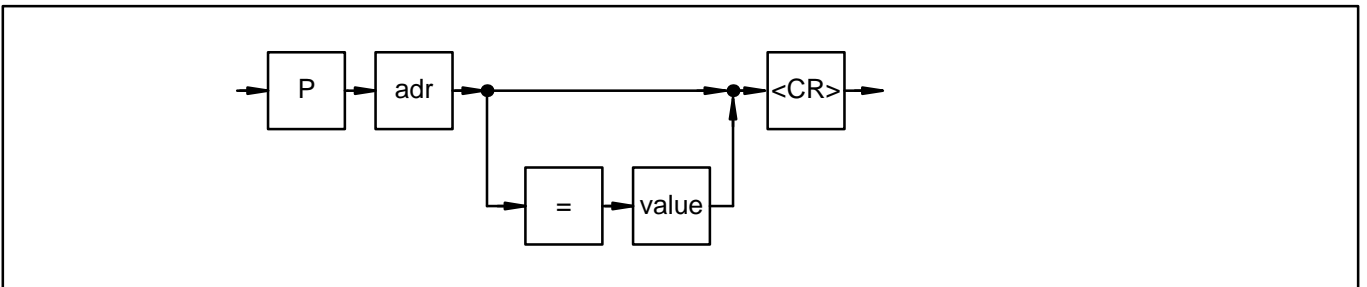
:82H —> :92H

:83H —> :93H

Read/write port

A value from the I/O area is displayed and modified byte-serially.

Command:



adr: I/O address

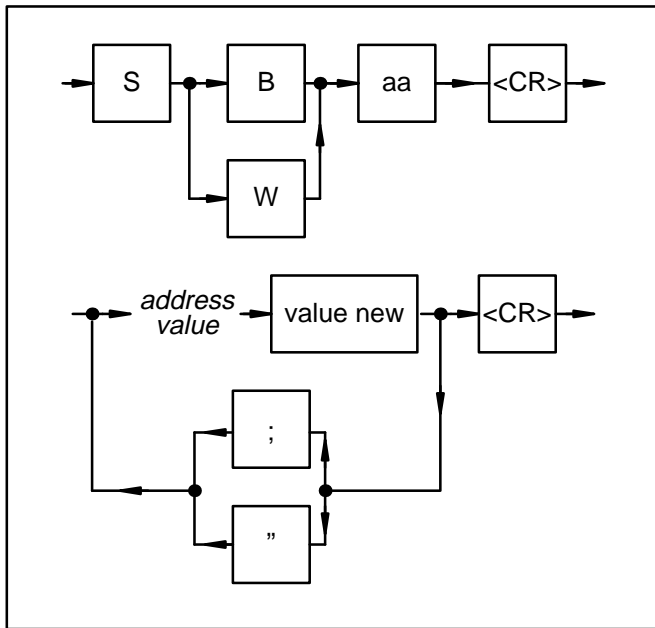
value: Byte value to be written to the I/O address

=: Keyword

Display/edit memory contents

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

Command:

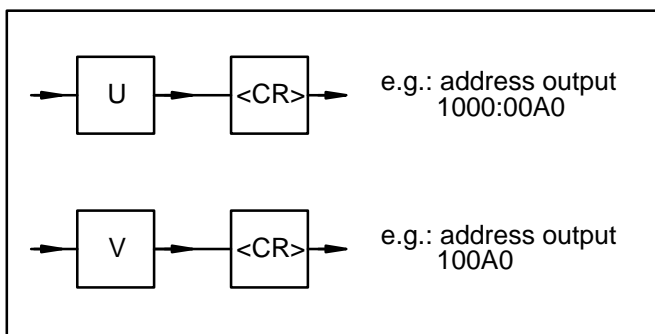


- B: Byte-serially (keyword)
- W: Word-serially (keyword)
- aa: Start address as of which the memory contents are to be displayed/modified
- address: Address of the memory contents
- value: Value of the memory contents
- value new: New value of the memory contents (user entry)
- :: Entering a semicolon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

Edit address output format

The monitor program is set to SEGMENT:OFFSET address format when it is initialized, and this format is used for each address output. The address output format can be freely selected with the user commands U<CR> (Segment:Offset format) and V<CR> (Absolute format).

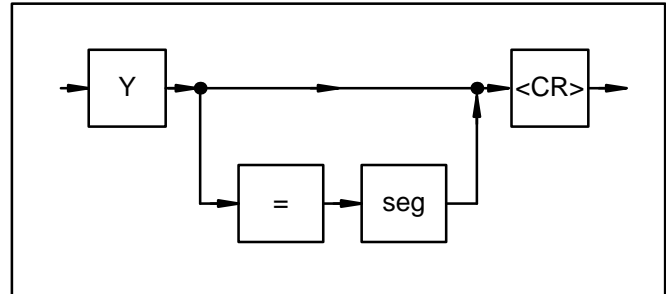
Command:



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.

Command:



- seg: New segment address of the working segment
- =: Keyword

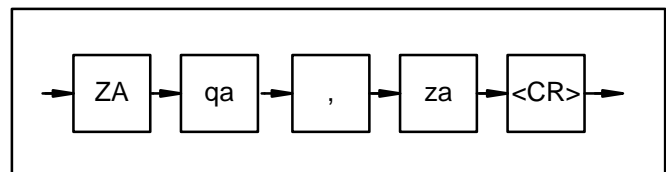
Example:

Y <CR>	User command: Display working segment
Y 0000	Screen display
DB 0L2	Display memory contents byte-serially
0000:0000 02 00	Screen display
Y=8000<CR>	Modify working segment

Cyclic read and write

A value is read cyclically from a source address and written to a target address. The operation can be aborted with CTRL C.

Command:



- qa: Source address from which the value is read
- za: Target address to which the value is written
- ,: Keyword (separator)

Example:

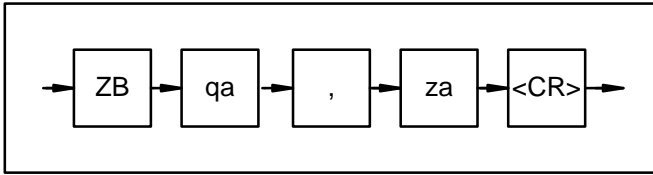
ZA 1000:0, 1000:100 <CR>

A value is read cyclically from address 1000:0_H and written cyclically to address 1000:100_H.

Cyclic read and write with waiting time

A value is read cyclically from a source address and written cyclically to a target address. The operation can be aborted with CTRL C. The waiting time between two read cycles is approximately 1 ms.

Command:



qa: Source address from which the value is read
za: Target address to which the value is written
,: Keyword (separator)

Example:

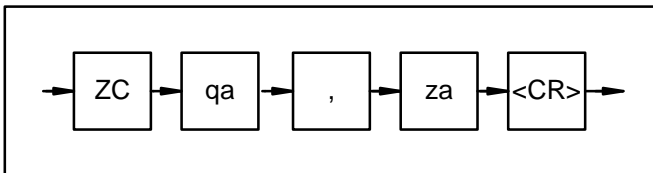
ZA 1000:0, 1000:100 <CR>

A value is read cyclically from address 1000:0_H and written cyclically to address 1000:100_H.

Read and write on keystroke

After each keystroke, a value is read from the source address and written to a target address. The operation can be aborted with CTRL C.

Command:



qa: Source address from which the value is read
za: Target address to which the value is written
,: Keyword (separator)

Example:

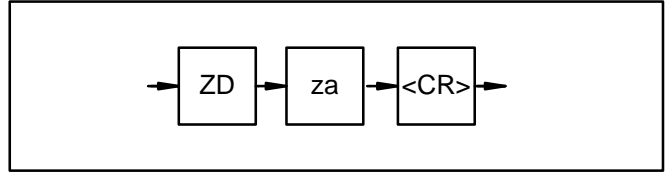
ZC 1000:0, 1000:100 <CR>

With each keystroke, a value is read from address 1000:0_H and written to address 1000:100_H.

Cyclic write

The value of a counter is decremented and written to a target address. The operation can be aborted with CTRL C.

Command:



za: Target address to which the value is written

Example:

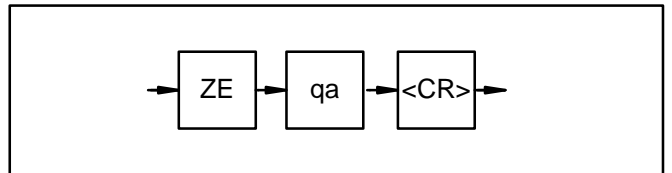
ZE 1000:100 <CR>

The value of a counter is written to 1000:100_H. The counter is decremented after each write operation.

Cyclic read

A source address is read cyclically. The operation can be aborted with CTRL C.

Command:



qa: Source address from which the value is read

Example:

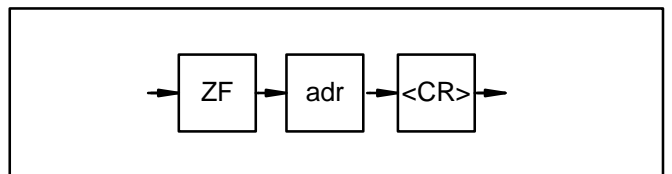
ZE 1000:100 <CR>

The value of address 1000:100_H is read cyclically.

Cyclic write and read

The value of a counter is written cyclically to an address and then read again. The operation can be aborted with CTRL C.

Command:



adr: Address to which the value of the counter is written and from which the value is read

Example:

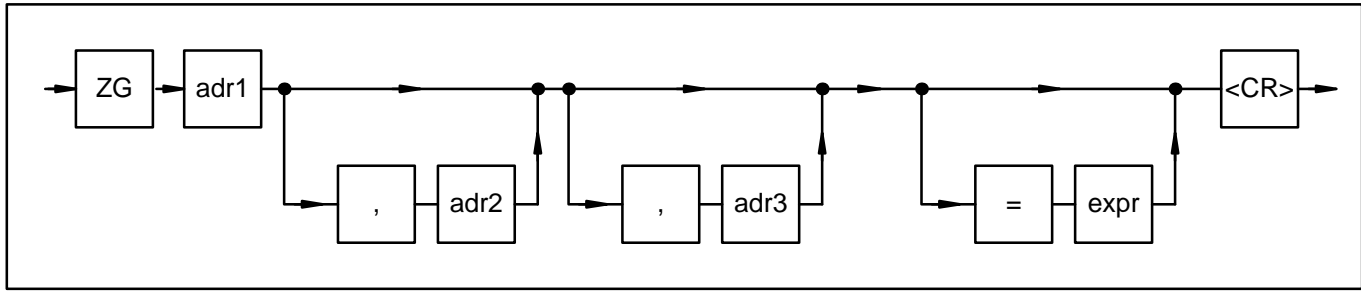
ZF 1000:0 <CR>

The value of a counter is written to address 1000:100_H. After each write operation, the value is read from address 1000:100_H and the counter is decremented.

Simultaneous output of 3 values

The ZG command permits the values of maximum 3 addresses to be displayed. 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>

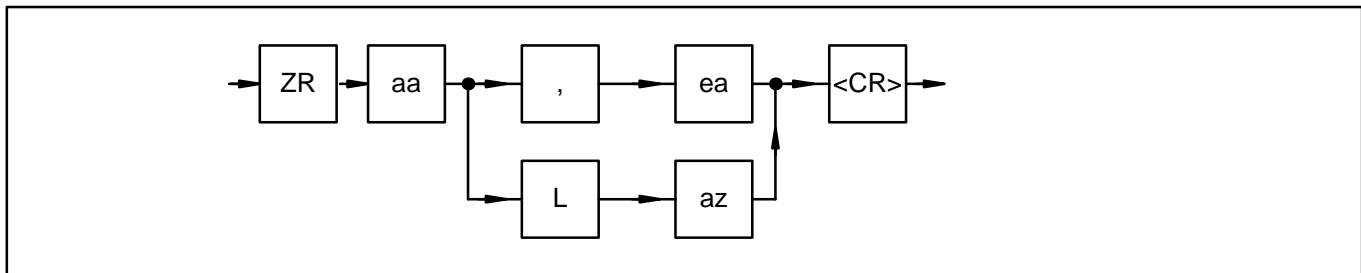
The values of addresses 1000:0_H and 1000:100_H are displayed on the monitor. If the value of address 1000:0_H changes, the values of the two addresses are updated on the monitor.

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)

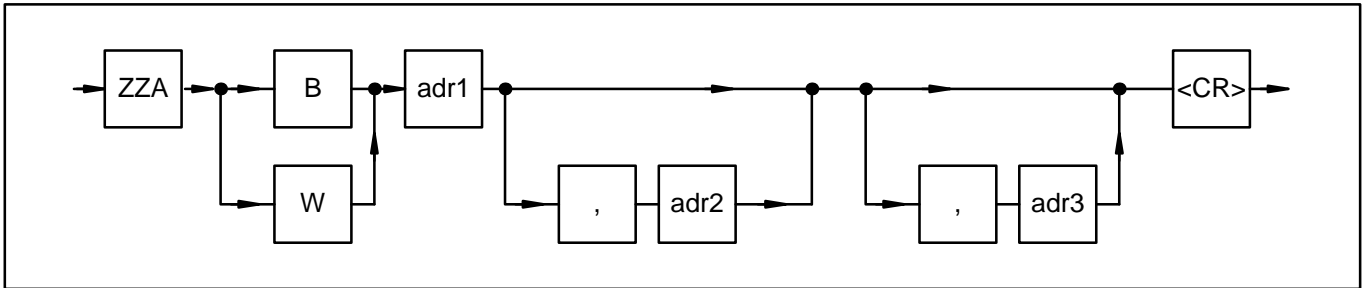
Example:

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

Output of 3 values after entering a semicolon (;)

Command ZZA permits you to display the values (byte or word) of maximum 3 addresses each 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:

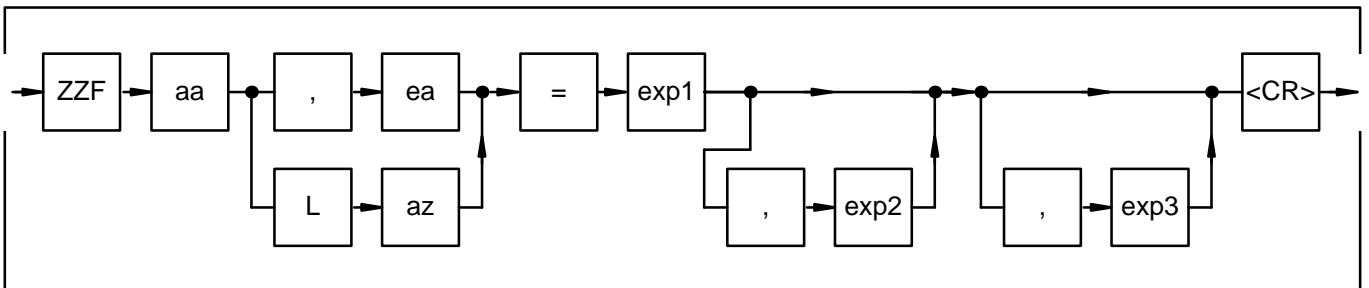
ZZA 1000:0, 1000:100 <CR>

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, 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 in the memory area
- exp1: 1st word of the string
- exp2: 2nd word of the string
- exp3: 3rd word of the string
- ,: Keyword (separator)

Example:

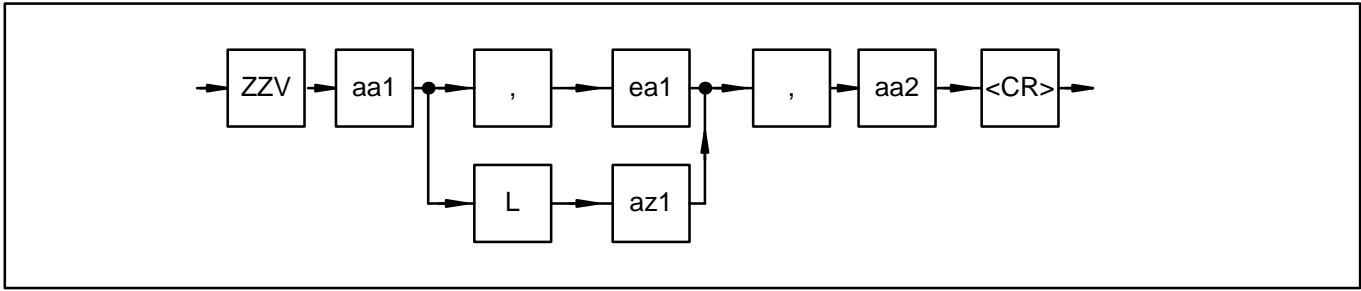
ZZF 1000:0, 100 = AAAA, BBBB <CR>

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 established, the address 1, the contents 1, the address 2 and the contents 2 are displayed on the monitor. The operation can be aborted with CTRL C.

Command:



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

Example:

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

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 COM2 serial interface of unit 07 KT 92 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

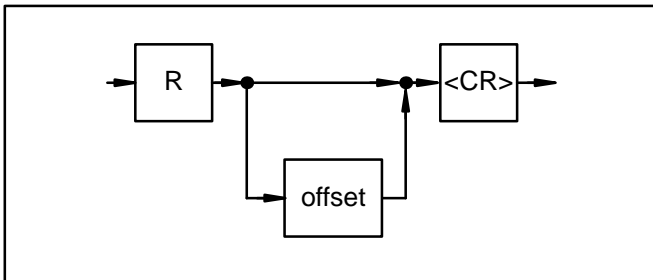
The following transfer format applies:

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

The data of the INTEL HEX file 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. This permits the storage area for the HEX file data in the PLC to be preset.
- The offset address is determined by the address in the data record of the INTEL HEX file.

Command:



offset: Offset (by addition to the segment address of the address extension record, this results in the new segment address)

Example:

R <CR> The PLC is ready to receive an INTEL HEX file.

R 2F00 <CR> The PLC is ready to receive an INTEL HEX file. The HEX value 2F00_H is added to the segment address of the address extension record. The resultant new segment address is the address used for storing the HEX file data.

Write INTEL HEX file

The W command permits a data area of the PLC to be output as an INTEL HEX file via the serial interface COM2 of the unit 07 KT 92.

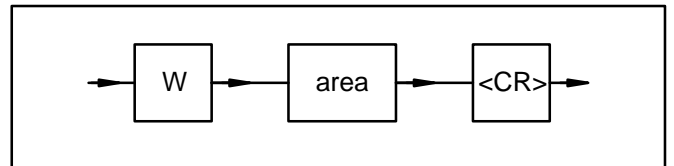
The following records are generated in this case:

- address extension record
- data record
- end record

The following transfer format applies:

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

Command:



area: Memory area to be output as an INTEL HEX file.

Example:

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

W 8000:0LFFFF <CR> The memory area from 8000:0_H up to and including 8000:FFFE_H is output as an INTEL-HEX file via serial interface COM2 of the PLC.

3 Memory overview

3.1 Memory overview for 07 KR 91, 07 KT 92 and 07 KT 93

3.1.1 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

Explanation of terms:

- Organizational directory
 - PLC-specific: This is used to store organizational data relating to the entire PLC.
 - for user program 1: This is used to store organizational data relating to program memory 1.
 - for user program 2: This is used to store organizational data relating to program memory 2.
- Program identification: 16 bytes for an identification, e.g. project name.

3.1.2 User program Flash-EPROM

Checksum	A7FFE
Not used 8C _H bytes	A7F72
User program 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

- 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: Memory for the PLC program.
- 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.

3.1.3 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 constants BINARY

KW: Indirect constants WORD

KD: Indirect constants 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

Stack 2: Stack 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 allowed for 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 allowed for when generating and outputting the process image.

3.1.4 Dual-port RAM

CS31–status (EW 07,15)	C03FF C03FE C03FD
read real time clock EW 07,08...EW 07,14	C03F0 C03EF
spontaneous mail box (EW 07,04...EW 07,07)	C03E8 C03E7
receive mail box (EW 07,00...EW 07,03)	C03E0 C03DF
direct: EW 06,00...EW 06,15 CS31: EW 00,00...EW 05,15	C0300
reserved	C02FF
send mail box	C02FE C02F4 C02F3 C02E0 C02DF
reserved	
direct: AW 06,00...AW 06,15 CS31: AW 00,00...AW 05,15	C0200 C01FF
direct: E 62,00...E 63,15 CS31: E 00,00...E 61,15	C0180 C017F C0100 C00FF
reserved	
read/write permission read/write request	C00FE C00FD C0080 C007F
reserved	
direct: A 62,00...A 63,15 CS31: A 00,00...A 61,15	C0000

3.2 Memory overview for 07 KR 31 and 07 KT 31

3.2.2 Data addressing (Data mapping)

3.2.1 System addressing (Mapping)

FFF F D800 D7FF	Compiled program 1
B000 AFFF	Compiled program 2
AF00 AEFF	Reserved
AC00 ABFF	I/O data
A800 A7FF	Reserved
8981 8980	Micro code in RAM
8800 87FF	Constants
8000 7FFF	RAM non-safeguarded
5000 4FFF	Reserved
4000 3FFF	Data
2000 1FFF	UAR T
1000 0FFF	ASIC 2 – input ASIC
0000	ASIC 1 – output ASIC

AEFF	EW 15,15
AD00 ACFF	EW 00,00
AC80 AC7F	E 63,15
AC00	E 00,00
	A 63,15
	A 00,00
47D1	S 015,15
47B2	S 000,00
467F	MD 001,15
4600	MD 000,00
8980 897F	K 00,00; K 00,01
8900 88FF	KD 01,15
	KD 00,00
8800	KW 07,15
	KW 00,00
85FF	AW 07,15
8500	AW 00,00
4581	MW 255,15
4542 4541	MW 254,00
	MW 239,15
4402 4401	MW 230,00
	MW 005,15
	MW 000,00
4342 4341	M 255,15
4340 433F	M 255,00
	M 239,15
432C 432B	M 230,00
	M 021,15
4300 42FF	M 000,00
	Historical values
4100 40FF	
4000	Timers

