CL350 / CL400 / CL500

Operations List
Software Manual

Edition 112

CL350 / CL400 / CL500

Operations List
Software Manual

1070 072 127-112 (00.02) GB

1 Safety Instructions

Before you start programming and working with the CL350, CL400 or CL500 controller, we recommend that you thoroughly familiarize yourself with the contents of this instruction manual. Keep this manual in a place where it is always accessible to all users.

1.1 Standard Operation

This instruction manual presents a comprehensive set of instructions and information required for the standard operation of the described products. The described products are used for programming and operating the CL350, CL400 or CL500 control units.

The products described hereunder

- were developed, manufactured, tested and documented in accordance with the relevant safety standards. In standard operation, and provided that the specifications and safety instructions relating to the project phase, installation and correct operation of the product are followed, there should arise no risk of danger to personnel or property.

The prerequisites for trouble-free service and safe operation of the product are proper transport, handling and storage, placement and installation, plus careful operation of the equipment.

1.2 Qualified Personnel

The requirements pertaining to qualified personnel are based on the job specifications as outlined by the ZVEI (central association of the electrical industry) and VDMA (association of German machine and plant builders) professional associations in Germany. Please refer to the following German-language publication:

Weiterbildung in der Automatisierungstechnik
Hrsg.: ZVEI und VDMA
MaschinenbauVerlag
Postfach 71 08 64
60498 Frankfurt

This instruction manual specifically addresses PLC engineers. They will require specific knowledge of the CL350, CL400 or CL500 controllers.

Interventions in the hardware and software of our products which are not described in this instruction manual may only be performed by specially trained Bosch personnel.

Unqualified interventions in the hardware or software or non-compliance with the warnings listed in this instruction manual or indicated on the product may result in serious personal injury or damage of property.

Installation and maintenance of the products described hereunder is the exclusive domain of trained electricians as per VDE 1000-10, who are familiar with the contents of this manual.

Trained electricians are persons of whom the following is true:

- They are capable, due to their professional training, skills and expertise, and based upon their knowledge of and familiarity with applicable technical standards, of assessing the work to be carried out, and of recognizing possible hazards.
- They possess, subsequent to several years' experience in a comparable field of endeavour, a level of knowledge and skills that may be deemed commensurate with that attainable in the course of a formal professional education in this area.

With regard to the foregoing, please read the information about our comprehensive training program. You will find a listing of our seminars on the front inside cover of this instruction manual. The professional staff at our training centre will be pleased to provide detailed information. You may contact the centre by telephone at (+49) 6062 78-258.

1.3 Safety Labels Affixed to Components

Danger: High voltage!

Danger: Battery acid!

Electrostatically sensitive devices!

Disconnect at mains before opening!

Pin for connecting PE conductor only!

Functional earthing / low noise earth

For screened conductor only!

1.4 Safety Instructions in this Manual

DANGEROUS ELECTRICAL VOLTAGE
This symbol is used to warn of the presence of a **dangerous electrical voltage**. Insufficient compliance with or failure to observe this warning may result in **personal injury**.

DANGER
This symbol is used wherever insufficient or lacking compliance with instructions may result in **personal injury**.

CAUTION
This symbol is used whenever insufficient or lacking compliance with instructions may result in **damage to equipment or data files**.

⟹ This symbol is used to alert the user to an item of special interest.

1.5 Safety Instructions for the Described Product

DANGER
Fatal injury hazard through ineffective Emergency-OFF safety devices!
Emergency-OFF safety devices must remain effective and accessible during all operating modes of the system.
The release of functional locks imposed by Emergency-OFF devices must never be allowed to cause an uncontrolled system restart! Before restoring power to the system, test the Emergency-OFF sequence!

DANGER
Danger to Personnel and Equipment!
Test every new program before operating the system!

DANGER
Retrofits or modifications may interfere with the safety of the products described hereunder!

The consequences may be severe personal injury, damage to equipment or environmental hazards. Therefore, any system retrofitting or modification utilizing third-party components will require express approval by Bosch.

1.6 Documentation, Version and Trademarks

Documentation

The present instruction manual provides the user with comprehensive information about programming the CL350, CL400 and CL500 control units, and about the instruction set used by the referred devices.

List of instruction manuals:

Instruction manual	Language	Order no.
CL400, Manual	English	1070 072 143
CL500, Manual	English	1070 072 123
SFC Sequence Function Chart	English	1070 072 186
CL500 System Commands, Software manual	English	1070 072 137

⇒ **Throughout this instruction manual, the floppy disk driveshall always have drive letter A:, and the hard disk drive shall have drive letter C:.**

Special keys or keyboard shortcuts (key combinations) are enclosed in pointed brackets:

- Special keys, example: <Enter>, <PgUp>, <Del>
- Key combination (pressed simultaneously), e.g.: <Ctrl> + <PgUp>

Trademarks

All trademarks referring to software that is installed on Bosch products when shipped from the factory represent the property of the respective manufacturers.

When shipped from the factory, all installed software is protected by copyright. It may therefore be duplicated only with prior permission by Bosch or in accordance with the licensing agreements with the respective manufacturer or copyright owner.

MS-DOS® and Windows™ are registered trademarks of Microsoft Corporation.

2 Introduction

The purpose of the present Operations List is to support the user of the CL350, CL400 and CL500 Programmable Logic Controllers with regard to programming and startup procedures.

It should be noted that, with regard to the extent of functional features, this manual describes the ZS35x, ZS40x and ZS501 central processing units!

The functional features of the ZS35x and ZS40x central processing units differ only with respect to the following:

- The size of the program memory is 64 kB instead of 120 kB.
- The addressable inputs/outputs occupy 64 bytes instead of 256 bytes.
- System bus interface (host connection) is not available.
- Both units feature a reduced set of system commands.

In the case of the CL500 PLC controller, there exists a version-dependent difference in the extent of functional features between both the ZS500 and ZS501 central processing units, as well as between the various ZS types. For this reason, prior to configuring or programming essential functions, prior verification is required to ascertain that the desired function is actually supported by the ZS version being utilized.

This descriptive text does not furnish a detailed discussion of related aspects!

2.1 Programming Differences between PROFI and WinSPS Software

The present documentation discusses the representation of constants and program module calls in the notation generated by the PROFI programming unit software. However, due to the adaptation to IEC1131-3, when processed with the WinSPS programming unit software, the representation of constants and program module calls will differ to some extent.

Differences in programming and notation of word constants

Data Type		PLC utility programs	
Explanation	**Notation**	**PROFI**	**WinSPS**
UINT (unsigned integer)	Binary / Dual	K00000000 00000000B K11111111 11111111B	2#0000000000000000 2#1111111111111111
	Decimal, word	K00000D - K63535D	00000 - 65535
	Decimal, byte/byte	K000/000 - K255/255	Not defined in IEC1131 Part 3
	Hexadecimal	K0000H - KFFFFH	16#0000 - 16#FFFF
INT (signed integer)	Decimal, word	K-32768 - K+32767 K-32768D - K+32767D	-32768 - +32767
Text, STRING(2)	ASCII	K'AB'	'AB'
Time value, TVALUE	Time val. (+timebase r) r: 0=10ms, 1=100ms 2=1s, 3=10s	K0.r - K1023.r	T#10ms - T#10230s T#0.r - T#1023.r

Differences in programming and notation of module calls

	PLC utility programs			
	PROFI		**WinSPS**	
Program module / function call (IEC1131-3)	CM	PM	CM	FC

Differences in programming and notation of jump instructions

	PLC utility programs			
	PROFI		**WinSPS**	
Jump instruction	JPx	-label	JPx	label
Jump destination	-label		label:	

3 System Configuration Table (SC Table)

Because the CL350 does not provide system bus capability, this control unit does not feature an SC Table.

All intelligent modules operating on the system bus of the CL400 / CL500 basic unit must be declared in the SC Table.

Exceptions:
The ZS40x central processing unit and the and der SK500 system coordinator always comprise module no. 0, and therefore not explicitly mentioned in the SC Table.

Based on the emntries in the SC Table, the ZS40x and/or SK500 assumes the bus management tasks. Additional declarations, i.e., to accommodate the multiple-processor operation of the CL500, are not required.

A valid SC Table is the prerequisite for starting up both the CL400 and CL500 systems.

An SC Table that has been found to be invalid will return error code 0/2 on the 7-segment display of the ZS40x and/or SK500.

The SC Table is created with the aid of the PG programming unit, and is then loaded into the ZS40x or SK500.

⇒ **Because the ZS40x central processing unit is often used in standalone operation, i.e., without the connection of other intelligent devices, it is shipped with a "blank" SC Table, dispensing with the need to create a new one from scratch.**

3.1 SC Table Entries

3.1.1 Module Number

The module number is identical to the row number in the SC Table.

In the case of an error, the module number selected here also corresponds to the ID appearing on the top display of the ZS40x or SK500.

In addition, standard function modules, such as the R5INIT, make reference to the module number.

Data messages of the BÜP64 (3964R) data transmission protocol utilize module numbers for the purpose of CPU identification.

A module that is to be entered in the SC Table can be inserted in any desired row of the SC Table, i.e., declared as a random module number.

There is no reference between module number and block address, or to the hardware slot in which a module is installed. The exception is the module no. 0. On the CL400, this is always the ZS40x, using block address $F0_H$ and, in the case of the CL500, this is the SK500, using block address FF_H.

For the CL500, it is recommended to declare the ZS50x modules as modules numbered 1 through 4.

3.1.2 Modules

The module designation of a module is entered at this point.

The following are valid designations:

CL400 & CL500	CL500 only
R500	ZS500 CAUTION: Ensure correct entry!
R500P	ZS501 CAUTION: Ensure correct entry!
R500MAP	ZS510 CAUTION: Ensure correct entry!
ZAT	PV500
DB500	

3.1.3 Rack Installation

Modules that are actually represented by physical hardware are identified by a "**J**" (Yes). Any modules that are planned but for which the hardware is not currently installed may be represented by a preliminary entry in the SC Table, and are thus marked with an "**N**" (No).

3.1.4 Block Address

The block address for the module is declared here. The address must correspond to the respective switch setting on the hardware module.

CL400/CL500

Module	permitted Block Address	Number of occupied blocks per module	Comment
R500	Multiples of 1	1	
R500P	Multiples of 4	4	
R500M	Multiples of 4	4	
ZAT	Multiples of 4	4	
PV500	Multiples of 4	4	except CL400
DB500	Multiples of 4	4	

CL500

For ZS50x modules, the following mandatory interrelation applies between the ZS no. of the block address to be selected in the PG programming unit and the peripheral start address:

ZS no. on PG	Block address (decimal)	Peripheral start address	Module number in SC Table
ZS0	0	0	Any number. Recommended: 1
ZS1	8	64	Any number. Recommended: 2
ZS2	16	128	Any number. Recommended: 3
ZS3	24	192	Any number. Recommended: 4

The following applies to the ZS no. setting on the hardware module:

ZS no. on PG	Segment positions of switch S5 onboard the ZS50x:			
	S5/1	S5/2	S5/3	S5/4
ZS0	off	off	off	off
ZS1	on	off	off	off
ZS2	off	on	off	off
ZS3	on	on	off	off

3.1.5 Synchronization Method (except CL400)

The entry affects the ZS50x and ZAT modules.

The processing of the PLC programs in different ZS501 modules can be synchronized at runtime.

To this end, there are eight synchronization methods which correspond to the eight characters in the SC Table. A "**0**" denotes no participation, while a "**1**" denotes participation in this synchronization procedure.

For a ZS500, all synchronization methods must be declared as "**0**".

3.1.6 Remanence Stop (except CL400)

The entry affects the ZS50x and ZAT modules.

Permitted entries are "**J**" (Yes) and "**N**" (No).

The entry is relevant only to the ZS501.

The entry "**J**" (YES) denotes that, in the case of a data loss in the referred ZS501, e.g., after physical module removal or deinstallation of the memory module, not only this particular ZS501 will enter STOP mode, but a system STOP condition will be triggered also.

This function can also be used, for example, to signal a ZS501 memory error via the floating-potential switching contact of the power supply. As a prerequisite, the jumpers on the NT power supply module must be set properly.

⇒ **In the event that a remanence STOP has been declared, the initiation of a system RUN, either via slide switch on the SK500 or via the PG programming unit, will also be required subsequent to a memory error with subsequent reload!**

3.1.7 Interrupt Modules

CL400

The peripheral addresses EI0, EI1 and EO0, EO1 – provided they are equipped – will always be addressed as interrupt module (II24). They are reserved for this purpose, and may not be selected for any other type of peripheral module that is operated on the EI/EO.

CL500

The entry affects the ZS50x and ZAT modules.
In conjunction with a ZS501, entries "**0**" and "**1**" are permitted.
In conjunction with a ZS500, a "**0**" must be entered!
The entry "**1**" means that an interrupt input module is being considered. This is the prerequisite for the utilization of peripheral interrupts. For more information about this subject, refer also to Section 5.14, "Interrupts".

3.1.8 Peripheral Start Address (except CL400)

The entry affects the ZS50x and ZAT modules. As discussed earlier, there exists a mandatory interrelation between the ZS50x block address and the peripheral start address:

ZS50x Block address	Peripheral start address
0	0
8	64
16	128
24	192

3.1.9 I/O & EI/EO Size

The entry affects the ZS50x and ZAT modules.

The values of both the I/O and EI/EO sizes must be identical!

As the default value for the useful size of the I/O and EI/EO area is 64 bytes, the value 64 must be entered in both columns.

When observing various constraints, a ZS501 may also be assigned I/O and EI/EO areas that are larger or smaller. This will require appropriate entries in the OM2 organization module.

A size reduction to 32 bytes, for example, is useful in cases where, in addition to four ZS50x central processing units, a ZAT with peripheral control is to be employed.

For the ZS501, a size increase of the peripheral area is also possible. Depening on the controller configuration, this will restrict the peripheral area of another ZS50x module within the CL500 system.

An additional consequence will be addressing limitations.

For more information about this subject, refer also to Section 6.14.4, "Indirect Addresses".

3.2 Peripheral Operation – CL350 / CL400

The peripheral operation from within the process range of the CL400 is assignment list-based. This means that, upon controller Power-ON, only those I/O modules that are actually equipped and/or entered in the OM2 (DW2/DW33-64) will be recognized and serviced during the I/O state. As a result, image updating time is optimized.

However, direct access via II and/or IO I/O addressing continue to be possible.

3.3 Peripheral Operation – CL500

The peripheral operation from within the process range of the CL500 is defined in the SC Table and always edffected up to the maximum address. This means that the ZS500 always services 64 I/O bytes, and the ZS501 always services the number of I/O bytes that were defined in the SC Table and in the OM2 (DW99 and DW100).

3.4 Sample SC Table for the CL500

```
+-------------------------------------------------------------------------------+
|                          CL500 System Configuration                           |
|-------------------------------------------------------------------------------|
|MOD | Modules  | In   | Block |   Synch.   |Rem. | IR  | Peri. |  I/O  | EI/EO |
|No. |          |Rack  | Addr. |   Method   |STOP | MOD | Addr. |      Size     |
|----+----------+------+-------+------------+-----+-----+-------+-------+-------|
| 1  |  ZS500   |  J   |   0   |  00000000  |  N  |  0  |   0   |  64   |  64   |
| 2  |  ZS501   |  J   |   8   |  00000000  |  J  |  0  |  64   |  64   |  64   |
| 3  |  ZS500   |  J   |  16   |  00000000  |  N  |  0  |  128  |  64   |  64   |
| 4  |  ZS501   |  N   |  24   |  00000000  |  J  |  0  |  192  |  64   |  64   |
| 5  |  R500P   |  N   |  32   |            |     |     |       |       |       |
| 6  |  R500    |  N   |  36   |            |     |     |       |       |       |
| 7  |  R500M   |  N   |  40   |            |     |     |       |       |       |
| 8  |  PV500   |  N   |  48   |            |     |     |       |       |       |
| 9  |  ZAT     |  N   |  64   |  00000000  |     |  0  |       |       |       |
| 10 |          |      |       |            |     |     |       |       |       |
|    |          |      |       |            |     |     |       |       |       |
|-------------------------------------------------------------------------------|
|          Module-specific HELP function with <F10> function key                |
+-------------------------------------------------------------------------------+
 TEST.S5K                                          Insert
+----<F1>-------<F2>-------<F3>-------<F4>-------<F5>-------<F6>---------------+
||   Link    |   Load    | (E)EPROM | About... | Config.  |   End    |          ||
+-------------------------------------------------------------------------------+
PLC Loader Program                  Version 3.20  (c) 1987-93 Robert Bosch GmbH
```

4 Status & Error Displays on the ZS35x and ZS40x

Status and error messages of the CL350 / CL400 system are indicated by both displays of the central processing units.

The top display shows the module number and the bottom display indicates the coded message (error code).

On the top display, the following module identifiers may be indicated:

Top display indication	The code indicated in the bottom display refers to the module:
0	ZS35x or ZS40x
F	Power supply module (except version 101)
1 through A	System module declared in the corresponding row of the SC Table.

The bottom display of the ZS35x / ZS40x indicates a message code.

Available codes range from 0 through F.

To provide an advanced level for the differentiation of statuses and errors, certain events cause a normal or superscript decimal point to be added to the "0 through F" codes.

4.1 Messages from the Power Supply

If "F" is shown in top display, the following will apply:

Bottom display indication	Explanation
0	Power supply fault Replace power supply unit
.5 *[1)]	Battery failure. Press RESET button on power supply to acknowledge.
.6 *[2)]	LOW BATTERY warning. Press RESET button on power supply to acknowledge.

CAUTION: Display has changed as compared to firmware 101

*[1)] Version 101: Display 0/**.**5

*[2)] Version 101: Display 0/**.**6

4.2 System Messages of the ZS35x and ZS40x

If "0" is shown in the top display, the following will apply:

Bottom display indication	Explanation:
1	Memory error Backup battery failure or MemoryCard load error. Once the condition is remedied, reloading the application program and deleting remanent areas will be required.
2	Configuration fault Discrepancy detected between SC Table and actual system equipping. When ZAT is used, also: SYSCOM not active, or wrong ZS firmware version, i.e., < 2.2
.2	A ZAT or R500MAP or R500P module has not yet logged in. Login may still occur. (This message code will be displayed, for example, during the ZAT boot phase.)
3	System bus error Hardware defect in a system module. Subsequent to replacement of defective system module, controller requires restart.
4	Configuration fault in system data fields / mailboxes The selected start addresses and sizes are causing overlaps.
.4	Peripheral initialization not yet concluded, e.g., DESI bus master not yet ready (ring interruption). The message code will be cancelled after the interval defined in the BM-DESI has timed out (default approx. 30 sec).
5	Assignment list error The assignment of I/O modules fails to match the defintions in the OM2 initialization table. CAUTION: Possible display misinterpretation with v101. Version 101: Display 0/**.**5 = battery failure
.7	Outputs disabled
8	Cycle time error
9	Programming errors - Module stack overflow - Application stack overflow - DM not active - Aborted direct access - Opcode error
A	Programming errors - Parameter error - Addressing error - Range violation - Module not available
b	HALT instruction
C	STOP request via X32 serial interface (V.24)
d	STOP request via system bus - System bus command - Module halted - Error message from system module, containing STOP request.
E	STOP request via X31 serial interface (PG programming unit)
F	System STOP via slide switch on the ZS40x

5 Status & Error Displays on the SK5xx

Status and error messages of the CL500 are indicated by both displays of the SK5xx.

The top display shows the module number and the bottom display indicates the coded message (error code).

On the top display, the following module identifiers may be indicated:

SK500 top display indication	The code indicated in the bottom display refers to the module:
0	SK500
F	Power supply module
1 through A	System module declared in the corresponding row of the SC Table.

The bottom display of the SK500 indicates a message code.

Available codes range from 0 through F.

To provide an advanced level for the differentiation of statuses and errors, certain events cause a normal or superscript decimal point to be added to the "0 through F" codes.

5.1 Messages from the Power Supply

If "F" is shown in top display, the following will apply:

SK500 Bottom display indication	Explanation:
0	Power supply fault Replace power supply unit
5˙	Battery failure. Press RESET button on power supply **and** system STOP/RUN to acknowledge.
6˙	LOW BATTERY warning. Press RESET button on power supply to acknowledge.

5.2 System Messages of the SK500

If "0" is shown in the top display, the following will apply:

SK500 Bottom display indication	Explanation:
0	Hardware problem: SK500 may be defective.
1	System bus conflict, possibly caused by defective system module.
2	Configuration fault. Discrepancy detected between SC Table and actual system equipping. When ZAT is used, also: SYSCOM not active, or wrong SK version, e.g. < 5
2.	A ZAT or R500MAP or R500P module has not yet logged in. Login may still occur. (This message code will be displayed, for example, during the ZAT boot phase.)
3	A system module fails to respond to being addressed. Possible causes: Programming error on ZAT, hardware problems, etc.
3.	Temporary system bus conflict
4.	Peripheral initialization not yet concluded, e.g., DESI bus master not yet ready (ring interruption). The message code will be cancelled after the interval defined in the BM-DESI has timed out (default approx. 30 sec).
5	Peripheral error: Double addressing of I/O modules, or wrong interrupt module addressing.
b	System STOP; no detailed information available.
d	System STOP upon request by a system bus command, e.g., originating in a ZS, ZAT or R500x. Recommended remedy in the case of a newly inserted SK500: **First**, load the valid SC Table, **then** acknowldege the battery fault on the power supply!
E	System STOP upon request via the SK500 interface for PG programming unit.
F	System STOP via slide switch on the SK500

5.3 Messages from System Modules, e.g., ZS500

If "1" through "A" are shown in the top display, the following will apply:

SK500 bottom display indication	Explanation:
0	Hardware problem
1	Memory error, e.g., caused by removal of memory module, or of the ZS, the SK or NT. Subsequent to memory error, loading plus deleting remanent areas are required! Subesequnt to loading: Enable via PG programming unit required! In the case of remanence STOP declaration: **System RUN** required!
2	Invalid OM2, wrong ZS no. entered in OM2, or no memory module installed.
3.	The cyclical master monitoring by the SK was disabled upon request by the system module identified in the top display, e.g., during the boot process on the ZAT.
3	
4	ZS501: Configuration fault. Wrong OM2 size, or errored mailbox initialization.
5	I/O assigtnemt error, in the case of ZS500, wrong ZS no. may have been entered in OM2.
6	Time expired during direct access. **Caution**: Do not confuse this error code with "**b**"!
8	Cycle time error
9	Application stack error, or no DM active, or attempted write-access to EPROM, or nesting depth exceeded.
A	Parameter error, or address error, or module nonexistent. Verify availability of time OMs declared in the OM2.
b	HALT instruction found.
C	System STOP upon request by the system module identified in the top display, e.g., during boot process of ZAT or R500M_EN.
d	System STOP upon request by a system bus command, e.g., originating in a ZS501, ZAT, SK500 or R500x.
E	Module STOP upon request via the serial interface of the ZS50x. On ZS500, also: Module STOP via slide switch on either ZS500 or SK500.
F	Module STOP via slide switch on the ZS501.

6 Programming Basics

Programmable Logic Controllers execute a machine program describing the tasks to be performed by the controller. To do this, a special programming language is used which may be displayed and printed out via various methods of representation or notations.

6.1 Representation Methods

Instruction List (IL)

Structure of Controller Instructions

Control Instruction			
Operation part	Operand attribute	Source operand	Destination operand
OPP	OPA	SRC ,	DEST

Examples:

```
A     B     I0.0
A     W     -name   ,   O
L     BY    O0      ,   B
T     W     C       ,   M10
MUL   W     K1234D  ,   D
```

Ladder Diagram (LD)

When using the LD representation method, the controller tasks are described by means of standard circuit diagram symbols.

Function Diagram (FUD)

When using the FUD representation method, a graphical symbol display (flow chart) illustrates the logical links.

Sequential Function Chart (SFC)

The SFC represents a graphical programming interface, which is used to describe the sequentially processed machine tasks in the form of a cascade sequence. Before it can be loaded into the PLC, this representation is then translated into the executable IL programming language.

6.2 Program Structure

To attain a clear PLC program structure which is easily readable, Bosch uses a consistent approach to structured programming for its programmable logic controllers, i.e., the programs are divided into functionally interconnected program segments or modules. To support the referred structuring, several module types performing various special functions are available.

Module Types

The controllers utilize the following module types:

- Organization modules
- Program modules
- Data modules

All modules are enabled by being invoked and/or activated in the course of program processing. Such a module call may occur either unconditionally or contingent on a binary link, on the result of a Compare function, and/or on an arithmetical operation.

6.3 Organization Modules (OM)

The organization modules perform all administrative or management functions for the controller program. Although they are programmed in the same manner as the program modules, only the system program invokes organization modules. All organization modules make use of the full instruction set of the PLC. There is no limitation to module size.

Organization modules may be divided into 6 function groups:

OM1 Module which is cyclically called by the system program, and which may be utilized as a distribution module for the entire program.

OM2 Non-executable definition module (initialization table), in which specifications (remanence limits) for the controller system are declared by modifying certain entries.

OM5, OM7 Startup modules for processing a variety of program sequences during a controller power-up or restart.

OM9 Error module which processes responses to program errors or fault conditions.

OM10-OM17 Interrupt modules for direct responses to peripheral events.

OM26-OM41 or system access.

OM18-OM25 Time-controlled processing (time matix selectable in OM2).

OM42 CL350 / CL400 only. Time-controlled interrupt invoked by the ms timer.

To ensure subsequent processing of the input/output cycle (I/O state), the OM1 must always be concluded with an End Of Program (EP) instruction. With the exception of OM2, and dependent upon the specific tasks to which they are assigned, the remaining organisation modules may be concluded with either the EP instruction or with End Of Module (EM). For programming the OM9 error module, it is useful to insert a definite HALT instruction (HLT) into the program immediately following the error response.

6.4 Program Modules (PM)

The program modules (PM) contain program segments that are technically and functionally interrelated. From within program modules, any number of additional program modules and data modules may be called. In addition, all program modules have access to the entire command set of the PLC. The modules are not subject to a size limit.

As a rule, program modules are concluded with an End of Module (EM) instruction. If the End of Program (EP) instruction is used, the program will be aborted immediately after the instruction has been processed, the input/output cycle activated, and further program processing again commence with the OM1 organization module.

Due to the option of parameterization, the program modules may be written independently of absolute operands. During the module call-up, the operands required for the current processing task are transferred to the program module in the form of parameter values.

The following input and output parameters may be specified:

Input parameters: Operands, constants and modules

Output parameters: Operands

6.5 Data Modules (DM)

The data modules (DM) serve as storage areas for all fixed and variable values and text blocks that are used by the program. Therefore, during PLC program processing, there exists the option of always keeping two data modules enabled, each of which provides up to 512 bytes of memory capacity.

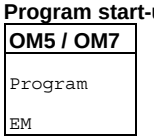
The following applies to the processing of data modules:

- Before their respective data may be accessed, the data modules must be enabled from within the program by means of module call instructions (i.e., CM for the 1st DM, and CX for the 2nd DM).
- Within a given organization module (OM) or program module (PM), the data modules remain current until other data modules are enabled by the program.
- After the return to the primary module, the data modules active at the time of the call-up of the base module are again activated.
- When the OM1 (cyclical program processing), and the start-up modules OM5 and OM7 are called, no data module is active.

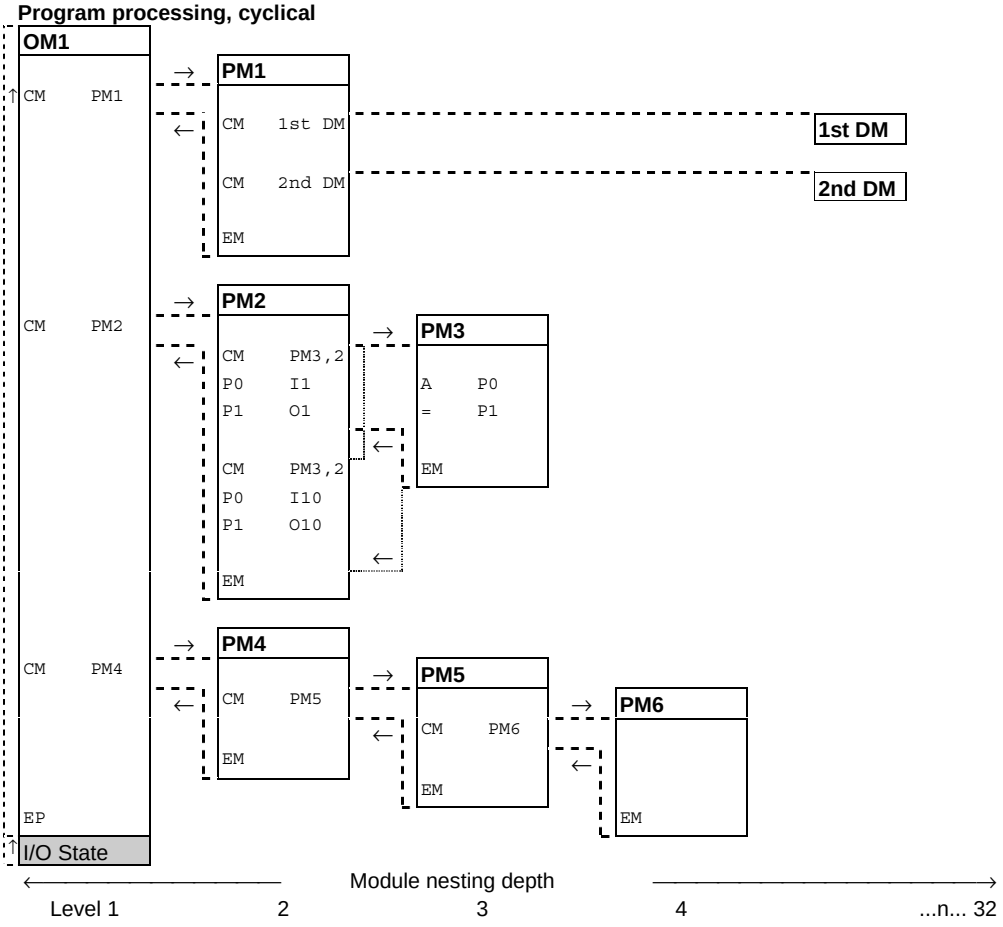
6.6 Program Structure

With the aim of providing a clear overview of the basic organization of program management, the following diagram shows an example of the program structure.

Program start-up, one-time only

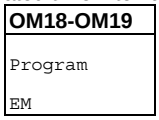


Program processing, cyclical



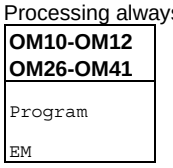
Time-controlled program processing

Processing always commences subsequent to the change of module (not module call) that follows the expiry of the associated time interval.



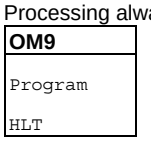
Interrupt-controlled program processing

Processing always commences immediately after the program error has been detected.



Program processing subsequent to PGM error

Processing always commences immediately after the program error has been detected.



6.7 OM2 Initialization Module for CL350 and CL400

The OM2 initialization module comprises a system initialization table that is linked with the PLC program as required.

A PLC program working without an OM2 utilizes preselected default values that are sufficiently useful for many applications.

Deviations from the preselected system defaults are declared in the OM2.

This may be used for example, to shift remanence limits, set cycle time limits, etc.

The time matrix definition for the time OMs is also handled in the OM2.

The declarations and definitions stored in the OM2 are adopted by the system upon Power-ON or in the case of a STOP/RUN command, even before processing a startup OM that may be present; a part of the OM2 is copied into the system area.

The following printout of an OM2 exemplifies all options of exercising control over the system initialization:

```
;*************************************************************************
;***                                                    ***
;***              I N I T I A L I Z A T I O N  T A B L E          ***
;***                                                 ***
;***             C o n t r o l   P r o c e s s o r   Z S 4 0 0        ***
;***                                                 ***
;*************************************************************************
;*** Proposed: 8. 9. 1994  (Staab/ESP2)                              ***
;*************************************************************************
;
;*************************************************************************
; OM2 : ZS400 Initialization table
;*************************************************************************
;
;     - must be integrated in every ZS400 user program which
;       uses different default settings
;
;     - if no OM2 entry in the ZS400 symbol file is made, the
;       default settings will be used
;
;     I M P O R T A N T   N O T E , please observe in any case
;     ========================================================
;
;     EVERY change of data words (DW) in forbidden address ranges
;     =====
;     can result in undefined system performance of the ZS400.
;
;*************************************************************************
;
;DW 1:      Block address (default value, must not be changed)
;--------------------
;
DEFW  W     16#00F0
;
```

```
;DW 2:      Initialization flag (entries permitted)
;-----------------------------
;           Entry 0 = Function  n o t  checked or executed
;           Entry 1 = Function checked or executed
;
DEFW  W     2#0000000000000000
;          ******||****||||   *: not used
;                  ||    |||+------ Check configuration list
;                  ||    ||+------- Check nominal cycle time
;                  ||    |+-------- Perform I/O State by nom. config. list
;                  ||    |          (otherwise by actual configuration)
;                  ||    +--------- Suppress cycle time monitor. at start-up
;                  ||
;                  |+-------------- Copy data module to data buffer
;                  +--------------- Disable outputs
;                                  0: does not work on extended outputs
;                                  1: works on extended outputs too
;                                  (last change 14.03.95)
;

;DW 3:      System bus handling (entries permitted)
;-----------------------------
;           Bit 0: System bus requests for background processing
;                  Entry 0 = Number not monitored
;                  Entry 1 = Number (16) monitored with negative ACK
;
;           Bit 1: Command passing retries with background processing
;                  Entry 0 = 100
;                  Entry 1 =  16
;
;           Bit 2: System background command buffer (SHB buffer)
;                  Entry 0 = SHB buffer overflow results in error message
;                  Entry 1 = SHB buffer overflow results in immediate
;                        command execution
;
;           Bits 3/4: Monitoring of command execution with immediate
;                 processing
;                  Entry 00 = 400 ms
;                  Entry 01 = 100 ms
;                  Entry 10 =  40 ms
;                  Entry 11 =  20 ms
;
;           Bits 5/6: Type of command passing for system background commands
;                 with flow coordination markers
;                  Entry x0 = in the background
;                  Entry 01 = immediately
;                         If command passing is not possible,
;                         acknowledgement is made at the user level and
;                         the command is not placed in the SHB.
;                  Entry 11 = immediately
;                         If command passing was successful, a positive
;                         report 70H is made. If command passing is not
;                         possible, the command is placed in the SHB
;                         and processed in the background.
;
;           Bits 7/8: Deletion of background commands in STOP
;           Bit 7  Entry 1 =  All background commands are deleted
;           Bit 8  Entry 1 =  All background commands are deleted for
;                        specific modules. The module numbers are
;                        specified in the system range by bits set
;                        in data bytes S38 and S39.
;                        Example: S38 Bit 0 = 1 --> Module no. 0
;
```

```
DEFW  W     2#0000000000000000
;          *******|||||||||         *: not used
;                 ||||||||+------      max./min. system bus requests (0/1)
;                 |||||||+-------      max./min. command passing retries (0/1)
;                 ||||||+--------      immediately upon SHB buffer overflow
;                 ||||++---------      timer monitoring immediate processing
;                 ||++-----------      proc. flow coord. background commands
;                 |+-------------      delete all background commands in STOP
;                +--------------      delete module-specific backgr. commands
;                                    in STOP

;*************************************************************************
;           System memory
;           =============
;           (must NOT be changed by the user!)
;                 ===
;           Default for data word DW4 = K0000H

DEFW  W     16#0000
;*************************************************************************

;DW 5:      Maximum cycle time (entries permitted)
;----------------------------
;           Entries as multiples of time base 10 ms of K1D and K200D
;           (10 ms to 2000 ms) for cycle time monitoring.
;           Function execution at DW2 / Bit 1 = 1.
;
DEFW  W     200
;

;DW 6:      Copy data module to data buffer (entries permitted)
;-----------------------------------------
;           Entry of K0D - K255D (data module number 0 - 255) possible.
;           (Function execution at DW 2 / Bit 8 = 1).
;
DEFW  W     0
;

;DW 7:      Number of first remanent timer (entries permitted)
;---------------------------------------
;           Entries of K0D and K128D are possible
;           K64D  = Remanency for timer loops T64 through T127
;           K128D = no remanency
;
DEFW  W     64
;

;DW 8:      Number of first remanent counter (entries permitted)
;---------------------------------------
;           Entries of K0D and K128D are possible
;           K64D  = Remanency for counters C64 through C127
;           K128D = no remanency
;
DEFW  W     64
;

;DW 9:      Number of first remanent marker (entries permitted)
;---------------------------------------
;           Entries of K0D and K256D are possible
;           K128D = Remanency from marker byte M128/marker bit M128.0,
;                 definition of remanency boudary via byte addresses
;           K256D : no remanency
;
DEFW  W     128
;
;
```

```
;DW 10: First remanent address in data buffer (entries permitted)
;-----------------------------------------
;           Entries of K0D and K512D are possible
;           K256D = Remanency from data buffer byte DP256
;           K512D : no remanency
;
DEFW  W     256
;
;

;           Definition of Timer OMs (entries permitted)
;           ========================
;           Entries as multipliers of time base 10 ms of K1D - K65535D
;           e.g. K0D  = no timer-based processing
;                K11D = 11 x 10 ms = 110 ms interval of processing time
;
;DW 11: Timer OM18
;-----------------
DEFW  W     0

;DW 12: Timer OM19
;-----------------
DEFW  W     0

;DW 13: Timer OM20
;-----------------
DEFW  W     0

;DW 14: Timer OM21
;-----------------
DEFW  W     0

;DW 15: Timer OM22
;-----------------
DEFW  W     0

;DW 16: Timer OM23
;-----------------
DEFW  W     0

;DW 17: Timer OM24
;-----------------
DEFW  W     0

;DW 18: Timer OM25
;-----------------
DEFW  W     0

;
;
;           Definition of mailboxes (ZS501 only) (entries permitted)
;           ===================================
;           Entries are possible from K0D to K24448D (DF0 to DF24448).
;           The available address range from the start address to the stop
;           address (DF24575) must not be exceeded by the mailboxes.
;
;DW 19: Start address of mailbox range in data field
;-------------------------------------------------
DEFW  W     0
;
```

```
;           Definition of mailbox sizes BK0 - BK7
;           -------------------------------------
;           The entry for the mailbox size is made in the corresponding
;           data words as multiples of 128 bytes.
;
;           Mailbox size = n x 128 bytes where n = 1H - 20H (max. 4kbytes)
;
;           Mailboxes are assigned to the bytes of the corresponding data
;           word as follows:  Left byte  = BKo (o = odd)
;                             Right byte = BKe (e = even)
;
;
;DW 20: Mailbox sizes BK1 and BK0
;-------------------------------------
DEFW  W     16#0000
;
;DW 21: Mailbox sizes BK3 and BK2
;-------------------------------------
DEFW  W     16#0000
;
;DW 22: Mailbox sizes BK5 and BK4
;-------------------------------------
DEFW  W     16#0000
;
;DW 23: Mailbox sizes BK7 and BK6
;-------------------------------------
DEFW  W     16#0000
;
;
;
;           Definition of system data fields (entries permitted)
;           ================================
;           Entries are possible from K0D to K24448D (DF0 to DF24448).
;           The available address range from the start address to the end
;           address (DF24575) must not be exceeded by the system data fields.

;DW 24: Start address of system data fields in data field
;-----------------------------------------------------
DEFW  W     0
;

;           Definition of system data field sizes DF0 - DF15
;           ------------------------------------------------
;           The entry for the system data field size is made in the
;           corresponding data words as multiple of 128 bytes.
;
;           DF size = n x 128 bytes where n = 1H - 20H (max. 4kbytes)
;
;           Data fields are assigned to the bytes of the corresponding
;           data field as follows:  Left byte  = DFo (o = odd)
;                                   Right byte = DFe (e = even)
;

;DW 25: System data field size: DF1 and DF0
;----------------------------------------
DEFW  W     16#0000
;
;DW 26: System data field size: DF3 and DF2
;----------------------------------------
DEFW  W     16#0000
;
;DW 27: System data field size: DF5 and DF4
;----------------------------------------
DEFW  W     16#0000
;
```

```
;DW 28: System data field size: DF7 and DF6
;-----------------------------------------
DEFW  W     16#0000
;
;DW 29: System data field size: DF9 and DF8
;-----------------------------------------
DEFW  W     16#0000
;
;DW 30: System data field size: DF11 and DF10
;-----------------------------------------
DEFW  W     16#0000
;
;DW 31: System data field size: DF13 and DF12
;-----------------------------------------
DEFW  W     16#0000
;
;DW 32: System data field size: DF15 and DF14
;-----------------------------------------
DEFW  W     16#0000
;
;


;           Definition of peripheral configuration lists
;           (entries permitted)
;           ===========================================
;
;           The configuration lists are used to enter the peripheral bytes
;           which are configured in the CL400 and for which a configuration
;           test shall be performed (see DW2, Bit 0). A test for mimimal
;           configuration is made upon start-up. Any additional
;           configuration is not checked.
;           Each I/O and EI/EO byte which is configured in the controller
;           and which shall be tested is marked with a "1" in the
;           corresponding data word. Byte which are not configured or shall
;           not be tested are to be marked "0".
;           16 I/O or EI/EO bytes are to be marked in each data word.
;
;           If in DW2, Bit3 the nominal configuration list is selected as
;           the basis for configuration-specific peripheral servicing in
;           the I/O state, only those peripheral bytes which have been
;           marked "1" in the configuration list will be serviced in the
;           I/O state.
```

```
;           Input configuration list
;           ------------------------
;
;DW 33: I-Byte    15 ..... 0
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 34: I-Byte    31 .... 16
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 35: I-Byte    47 .... 32
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 36: I-Byte    63 .... 48
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 37: I-Byte    79 .... 64
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 38: I-Byte    95 .... 80
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 39: I-Byte    111 ... 96
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 40: I-Byte    127 .. 112
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 41: I-Byte    143 .. 128
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 42: I-Byte    159 .. 144
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 43: I-Byte    175 .. 160
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 44: I-Byte    191 .. 176
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 45: I-Byte    207 .. 192
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 46: I-Byte    223 .. 208
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 47: I-Byte    239 .. 224
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 48: I-Byte    255 .. 240
;-------------------------
DEFW  W     2#0000000000000000
;
```

```
;           Output configuration list
;           -------------------------
;
;DW 49: O-Byte    15 ..... 0
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 50: O-Byte    31 .... 16
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 51: O-Byte    47 .... 32
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 52: O-Byte    63 .... 48
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 53: O-Byte    79 .... 64
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 54: O-Byte    95 ...      80
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 55: O-Byte    111 ... 96
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 56: O-Byte    127 .. 112
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 57: O-Byte    143 .. 128
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 58: O-Byte    159 .. 144
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 59      O-Byte      175 .. 160
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 60      O-Byte      191 .. 176
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 61: O-Byte    207 .. 192
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 62: O-Byte    223 .. 208
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 63: O-Byte    239 .. 224
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 64: O-Byte    255 .. 240
;--------------------------
DEFW  W     2#0000000000000000
```

```
;           Extended Input Range configuration list
;           ---------------------------------------
;
;DW 65: EI-Byte  15 .... 0
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 66: EI-Byte  31 ... 16
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 67: EI-Byte  47 ... 32
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 68: EI-Byte  63 ... 48
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 69: EI-Byte  79 ... 64
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 70: EI-Byte  95 ... 80
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 71: EI-Byte 111 ... 96
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 72: EI-Byte 127 .. 112
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 73: EI-Byte 143 .. 128
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 74: EI-Byte 159 .. 144
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 75: EI-Byte 175 .. 160
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 76: EI-Byte 191 .. 176
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 77: EI-Byte 207 .. 192
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 78: EI-Byte 223 .. 208
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 79: EI-Byte 239 .. 224
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 80: EI-Byte 255 .. 240
;-------------------------
DEFW  W     2#0000000000000000
;
```

```
;           Extended Output Range configuration list
;           ----------------------------------------

;DW 81: EO-Byte  15 .... 0
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 82: EO-Byte  31 ... 16
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 83: EO-Byte  47 ... 32
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 84: EO-Byte  63 ... 48
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 85: EO-Byte  79 ... 64
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 86: EO-Byte  95 ... 80
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 87: EO-Byte 111 ... 96
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 88: EO-Byte 127 .. 112
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 89: EO-Byte 143 .. 128
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 90: EO-Byte 159 .. 144
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 91: EO-Byte 175 .. 160
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 92: EO-Byte 191 .. 176
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 93: EO-Byte 207 .. 192
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 94: EO-Byte 223 .. 208
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 95: EO-Byte 239 .. 224
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 96: EO-Byte 255 .. 240
;-------------------------
DEFW  W     2#0000000000000000
```

```
;            !!!   Internal system memory data   !!!
;            ========================================
;
;           The following default settings must not be changed.
;           ===================================================

;DW 97: Data word (default K00BEH)
DEFW  W     16#00BE
;DW 98: Data word (default K0000H)
DEFW  W     16#0000
;DW 99: Data word (default K00C2H)
DEFW  W     16#00C2
;DW 100: Data word (default K7F00H)
DEFW  W     16#7F00
;DW 101: Data word (default K00C6H)
DEFW  W     16#00C6
;DW 102: Data word (default KFF00H)
DEFW  W     16#FF00
;DW 103: Data word (default K00C8H)
DEFW  W     16#00C8

;Default for data words DW 104 - DW 127 = K0000H
;----------------------------------------------
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
;*************************************************************************
EM
```

6.8 OM2 Initialization Module for CL500

The OM2 initialization module comprises a system initialization table that is linked with the PLC program as required. A PLC program working without an OM2 utilizes preselected default values that are sufficiently useful for many applications. Deviations from the preselected system defaults are declared in the OM2.

This may be used for example, to shift remanence limits, set cycle time limits, etc. The time matrix definition for the time OMs is also handled in the OM2.

The declarations and definitions stored in the OM2 are adopted by the system upon Power-ON or in the case of a STOP/RUN command, even before processing a startup OM that may be present; a part of the OM2 is copied into the system area.

The following printout of an OM2 exemplifies all options of exercising control over the system initialisation:

```
;*************************************************************************
;*************************************************************************
;***                                                                   ***
;***             I N I T I A L I Z A T I O N   T A B L E               ***
;***                                                                   ***
;***               C o n t r o l   P r o c e s s o r                   ***
;***             Z S 5 0 0 / Z S 5 0 1 / Z S 5 1 0  /  Z S 5 3 0         ***
;***                                                                   ***
;*************************************************************************
;*** Last change: 27. 4. 1994 (ESP2)                                  ***
;*************************************************************************
;*************************************************************************
; OM2 : ZS500/ZS510/ZS501/ ZS510 / ZS530 Initialization Table          ***
;*************************************************************************
;     - must be integrated in every ZS user program
;       which uses different default settings
;     - if no OM2 entry in the ZS symbol file is made,
;       the default settings will be used
;
;     I M P O R T A N T   N O T E , please observe in any case
;     ========================================================
;
;     EVERY change of data words (DW) in forbidden address ranges
;     =====
;     can result in undefined system performance of the ZS.
;
;*************************************************************************
;
;DW 1:      Block address (entries permitted)
;-------------------------------------------
;           - must match the DIP switch setting on the corresponding
;             ZS
;
DEFW  W     2#0000000000000000
;          ***********||***       *: not used
;                      |+---------- ZS Number Low-Bit
;                      +----------- ZS Number High-Bit
;           High-Bit Low-Bit  | ZS Number
;           ----------------------------
;            0    0     |  0
;            0    1     |  1
;            1    0     |  2
;            1    1     |  3
;
```

```
;DW 2:      Initialization flag (entries permitted)
;-----------------------------
;           Entry 0 = Function  n o t  checked or executed
;           Entry 1 = Function checked or executed
;
DEFW  W     2#0000000000000000
;          *******|******||         *: not used
;                 |      |+------      Check configuration list
;                 |      +-------      Check nominal cycle time (ZS501 only)
;                 +--------------      Copy data module to data buffer
;
;DW 3:      System bus handling (entries permitted)
;-----------------------------
;           Bit 0: System bus requests for background processing
;                  Entry 0 = Number not monitored
;                  Entry 1 = Number (16) monitored with negative ACK
;
;           Bit 1: Command passing retries with background processing
;                  Entry 0 = 100
;                  Entry 1 =  16
;
;           Bit 2: System background command buffer (SHB buffer)
;                  Entry 0 = SHB buffer overflow results in error message
;                  Entry 1 = SHB fuffer overflow results in immediate
;                        command execution
;
;           !!!!    The following selection options (bit 3 .. 9) in    !!!!
;           !!!!    data word 3 are only effective with ZS501 , ZS510  !!!!
;           !!!!    and ZS500 (update 201) modules.                    !!!!
;
;           Bits 3/4: Monitoring of command execution with immediate
;                 processing
;                  Entry 00 = 400 ms
;                  Entry 01 = 100 ms
;                  Entry 10 =  40 ms
;                  Entry 11 =  20 ms
;
;           Bits 5/6: Type of command passing for system background commands
;                 with flow coordination markers
;                  Entry x0 = in the background
;                  Entry 01 = immediately
;                         If command passing is not possible,
;                         acknowledgement is made at the user level and
;                         the command is not placed in the SHB buffer.
;                  Entry 11 = immediately
;                         If command passing was successful, a positive
;                         report 70H is made. If command passing is not
;                         possible, the command is placed in the SHB
;                         and processed in the background.
;
;           Bits 7/8: Deletion of background commands in STOP
;           Bit 7  Entry 1 =  All background commands are deleted
;           Bit 8  Entry 1 =  All background commands are deleted for
;                         specific modules. The module numbers are
;                         specified in the system range by bits set
;                         in data bytes S38 and S39.
;                         Example: S38 Bit 0 = 1 --> Module no. 0
;
;           Bit 9: Maximum number of commands to be executed per I/O cycle
;                  (OM1-/EP-/IO-coordinated commands).
;                  Entry 0 =  command priority, up to 20 commands/IO cycle
;                  Entry 1 =  cycle priority, up to 8 commands/IO cycle
;
;           !!!!    The following selection option (bit10) in          !!!!
;           !!!!    data word 3 is only effective with ZS501 , ZS510   !!!!
;           !!!!    and ZS500 (update 201) modules.                    !!!!
;
```

```
;           Bit 10: Polling of command- and acknowledgement buffer
;                   Entry   0 = every 40 milliseconds
;                   Entry   1 = every 20 milliseconds
;
DEFW  W     2#0000000000000000
;           *****|||||||||||         *: not used
;                |||||||||+------ max./min. system bus requests (0/1)
;                |||||||+------- max./min. command passing retries (0/1)
;                |||||||+-------- immediately upon SHB buffer overflow
;                ||||||++--------- timer monitoring immediate processing
;                ||||++----------- processing flow coord. backgr. commands
;                |||+------------- delete all backgr. commands in STOP
;                ||+-------------- delete module-specific backgr. commands
;               ||                in STOP
;                |+--------------- number of coord. commands in I/O cycle
;               +---------------- Polling of command- and acknowledgement
;                                   buffer
;

;DW 4:      Error handling (entries permitted)
;-------------------------
;           Entry 0 = system stop at error  n o t  executed
;           Entry 1 = system stop at error executed
;
DEFW  W     2#0000000000000000
;           *|||||||********       *: not used
;            ||||||+-------------- Rem. start-up desired but forbidden
;            |||||+--------------- Config. list conflict
;            ||||+---------------- Direct access takes too long
;            |||+----------------- Program error
;            ||+------------------ HALT command
;            |+------------------- Stop request from PG or SB
;            +-------------------- Stop switch
;

;DW 5:      Maximum cycle time (not effective with ZS500 < update 201)
;----------------------------
;           Entries as multiples of time base 10 ms of K1D and K200D
;           (10 ms to 2000 ms) for cycle time monitoring.
;           Function execution at DW2 / Bit 1 = 1.
;
DEFW  W     200
;
;DW 6:      Copy data module to data buffer (entries permitted)
;----------------------------------------
;           Entry of K0D - K255D (data module number 0 - 255) possible.
;           (Function execution at DW 2 / Bit 8 = 1).
;
DEFW  W     0
;
;DW 7:      Number of first remanent timer (entries permitted)
;----------------------------------------
;           Entries of K0D and K128D are possible
;           K64D  = Remanency for timer loops T64 through T127
;           K128D = no remanency
;
DEFW  W     64
;
;DW 8:      Number of first remanent counter (entries permitted)
;----------------------------------------
;           Entries of K0D and K128D are possible
;           K64D  = Remanency for counters C64 through C127
;           K128D = no remanency
;
```

```
DEFW  W     64
;

;DW 9:      Number of first remanent marker (entries permitted)
;-----------------------------------------
;           Entries of K0D and K256D are possible
;           K128D = Remanency from marker byte M128/marker bit M128.0,
;                 definition of remanency boundary via byte addresses
;           K256D : no remanency
;
DEFW  W     128
;
;
;DW 10: First remanent address in data buffer (entries permitted)
;-----------------------------------------
;           Entries of K0D and K512D are possible
;           K256D = Remanency from data buffer byte DP256
;           K512D : no remanency
;
DEFW  W     256
;
;
;           Definition of Timer OMs (entries permitted)
;           ======================
;           Entries as multiplier of time base 10 ms of K1D - K65535D
;           e.g. K0D  = no timer-based processing
;                K11D = 11 x 10 ms = 110 ms interval of processing time
;
;DW 11: Timer OM18
;-----------------
DEFW  W     0

;DW 12: Timer OM19
;-----------------
DEFW  W     0

;DW 13: Timer OM20
;-----------------
DEFW  W     0

;DW 14: Timer OM21
;-----------------
DEFW  W     0

;***********************************************
;  For the ZS500 < update 201, the TIMER OMs OB22
;  - OB25 must first be enabled programmatically
;***********************************************

;DW 15: Timer OM22 (must be enabled programmatically for ZS500 < 201)
;-----------------
DEFW  W     0

;DW 16: Timer OM23 (must be enabled programmatically for ZS500 < 201)
;-----------------
DEFW  W     0

;DW 17: Timer OM24 (must be enabled programmatically for ZS500 < 201)
;-----------------
DEFW  W     0

;DW 18: Timer OM25 (must be enabled programmatically for ZS500 < 201)
;-----------------
DEFW  W     0
```

```
;
;           Definition of mailboxes
;           =======================
;
;           !!!!    The following selection options in data word       !!!!
;           !!!!    19 to 23 are only effective with ZS501, ZS510      !!!!
;           !!!!    and ZS500 (update 201) modules.                    !!!!
;
;           Entries are possible from K0D to K24448D (DF0 to DF24448).
;           The available address range from the start address to the stop
;           address (DF24575) must not be exceeded by the mailboxes.
;
;DW 19: Start address of mailbox range in data field
;---------------------------------------------------
DEFW  W     0
;
;           Definition of mailbox sizes BK0 - BK7
;           -------------------------------------
;           The entry for the mailbox size is made in the corresponding
;           data words as multiples of 128 bytes.
;
;           Mailbox size = n x 128 bytes where n = 1H - 20H (max. 4kbytes)
;
;           Mailboxes are assigned to the bytes of the corresponding data
;           word as follows:  Left byte  = BKo (o = odd)
;                             Right byte = BKe (e = even)
;
;DW 20: Mailbox sizes BK1 and BK0
;--------------------------------
DEFW  W     16#0000
;
;DW 21: Mailbox sizes BK3 and BK2
;--------------------------------
DEFW  W     16#0000
;
;DW 22: Mailbox sizes BK5 and BK4
;--------------------------------
DEFW  W     16#0000
;
;DW 23: Mail boxes BK7 and BK6
;-----------------------------
DEFW  W     16#0000
;
;
;           Definition of system data fields (entries permitted)
;           ================================
;           Entries are possible from K0D to K24448D (DF0 to DF24448).
;           The available address range from the start address to the end
;           address (DF24575) must not be exceeded by the system data fields.

;DW 24: Start address of system data fields in data field
;-------------------------------------------------------
DEFW  W     0
;
;           Definition of system data field sizes DF0 - DF15
;           ------------------------------------------------
;           The entry for the system data field size is made in the
;           corresponding data words as multiple of 128 bytes.
;
;           DF size = n x 128 bytes where n = 1H - 20H (max. 4kbytes)
;
;           Data fields are assigned to the bytes of the corresponding
;           data field as follows:  Left byte  = DFo (o = odd)
;                                   Right byte = DFe (e = even)
;
```

```
;DW 25: System data field size: DF1 and DF0
;-----------------------------------------
DEFW  W     16#0000
;
;DW 26: System data field size: DF3 and DF2
;-----------------------------------------
DEFW  W     16#0000
;
;DW 27: System data field size: DF5 and DF4
;-----------------------------------------
DEFW  W     16#0000
;
;DW 28: System data field size: DF7 and DF6
;-----------------------------------------
DEFW  W     16#0000
;
;DW 29: System data field size: DF9 and DF8
;-----------------------------------------
DEFW  W     16#0000
;
;DW 30: System data field size: DF11 and DF10
;-----------------------------------------
DEFW  W     16#0000
;
;DW 31: System data field size: DF13 and DF12
;-----------------------------------------
DEFW  W     16#0000
;
;DW 32: System data field size: DF15 and DF14
;-----------------------------------------
DEFW  W     16#0000
;
;

;           Definition of peripheral configuration lists
;           (entries permitted)
;           ===========================================
;           The configuration lists are used to enter the peripheral bytes
;           which are configured in the CL500 and for which a configuration
;           test shall be performed (see DW2, Bit 0). A test for mimimal
;           configuration is made upon start-up. Any additional
;           configuration is not checked.
;           Each I/O and EI/EO byte which is configured in the controller
;           and which shall be tested is marked with a "1" in the
;           corresponding data word. Bytes which are not configured or
;           shall not be tested are to be marked "0".
;           16 I/O or EI/EO bytes are to be marked in each data word.
;
```

```
;           Input configuration list
;           ------------------------
;
;DW 33: I-Byte    15 ..... 0
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 34: I-Byte    31 .... 16
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 35: I-Byte    47 .... 32
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 36: I-Byte    63 .... 48
;--------------------------
DEFW  W     2#0000000000000000
;

;*****************************************************************************
;           System memory
;           =============
;           (must  N O T  be modified by the user !)
;                  =====
;           Default for data words DW 37 - DW 48 = K0000H
;
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
;*****************************************************************************
;
;

;           Output configuration list
;           -------------------------
;
;DW 49: O-Byte    15 ..... 0
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 50: O-Byte    31 .... 16
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 51: O-Byte    47 .... 32
;--------------------------
DEFW  W     2#0000000000000000
;
;DW 52: O-Byte    63 .... 48
;--------------------------
DEFW  W     2#0000000000000000
;
```

```
;*************************************************************************
;           System memory
;           ==============
;           (must  N O T  be modified by the user !)
;                  =====
;           Default for data words DW 53 - DW 64 = K0000H
;
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
;*************************************************************************
;
;

;           Extended Input Range configuration list
;           ---------------------------------------
;
;DW 65: EI-Byte  15 .... 0
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 66: EI-Byte  31 ... 16
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 67: EI-Byte  47 ... 32
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 68: EI-Byte  63 ... 48
;-------------------------
DEFW  W     2#0000000000000000
;

;*************************************************************************
;           System memory
;           ==============
;           (must  N O T  be modified by the user !)
;                  =====
;           Default for data words DW 69 - DW 80 = K0000H
;
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
;*************************************************************************
;
;
```

```
;           Extended Output Range configuration list
;           ----------------------------------------

;DW 81: EO-Byte  15 .... 0
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 82: EO-Byte  31 ... 16
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 83: EO-Byte  47 ... 32
;-------------------------
DEFW  W     2#0000000000000000
;
;DW 84: EO-Byte  63 ... 48
;-------------------------
DEFW  W     2#0000000000000000
;

;*****************************************************************************
;           System memory
;           =============
;           (must  N O T  be modified by the user !)
;                  =====
;           Default for data words DW 85 - DW 96 = K0000H
;
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
;*****************************************************************************
;


;           Clear Output
;           ------------

;DW 97: (default K00BEH , extension K7518H)
;
;     the extension activates the settings in data word 98.
;
DEFW  W     16#00BE
;
;DW 98: (default K0000H)
;
;     K0000H : clear output will delete normal range only (default)
;     K0001H : clear output will delete normal and extended range
;
DEFW  W     16#0000
```

```
;           Extended I/O Range
;           ------------------
;       !!!!    The following selection options (in data words   !!!!
;       !!!!    99 and 100) are only effective with ZS501,       !!!!
;       !!!!    ZS510 and ZS530 modules                          !!!!
;
;DW 99: (Default K00C2H, enable value K4193H)
;
;     If the 64-byte standard address range is not used, the entry
;     K4193H must be made in DW99, deviating from default K00C2H,
;     and DW100 needs to be modified accordingly !
;
DEFW  W     16#00C2
;
;DW 100: (default K7F00H)
;
;     Range definition for deviations from standard I/O range
;     in connection with entry K4193H in DW99 :
;
;     K7F00H ===>  64 bytes (default)
;     K7F01H ===>  32 bytes
;     K7F02H ===>  64 bytes
;     K7F04H ===> 128 bytes
;     K7F08H ===> 256 bytes
;     K7F0FH ===>   0 bytes
;
;     For the ZS530 with associated busmaster DP, it is possible to define
;     a remote extended I/O range. The following settings are possible:
;
;     K7F24H ===>  64 bytes central +  64 bytes remote extended I/O
;     K7F28H ===>  64 bytes central + 192 bytes remote extended I/O
;
;     NOTE: In the SK500/510 system configuration list, only the 64 bytes
;           of central I/O must be specified for the definition of remote
;           extended I/O !

DEFW  W     16#7F00
;
;
;DW 101: Data word (default K00C6H, enable value K52A4H)
;
;     For the ZS530, the coupling field address of an associated busmaster
;     can be specified in data word 102 in order to use remote extended
;     I/O. If a coupling field address other than EI/EO0 is to be used, the
;     enable value needs to be entered in DW101 and the EI/EO address in
;     low byte DW102.
;

DEFW  W     16#00C6

;DW 102: Data word (default KFF00H)
;
;     Coupling field address of the associated busmaster. The low byte may
;     contain an even-numbered byte address fromthe 64-byte extended range ; of the
ZS530. The value will only be accepted if the enable value is
;     specified in DW101.
;
DEFW  W     16#FF00
;
```

```
;
;           !!!    Internal system memory data      !!!
;           ========================================
;
;        The following default settings must not be changed.
;        ===================================================
;
;DW 101: Data word (default K00C6H)
DEFW  W     16#00C6
;DW 102: Data word (default KFF00H)
DEFW  W     16#FF00
;DW 103: Data word (default K00C8H)
DEFW  W     16#00C8

;Default for data words DW 104 - DW 127 = K0000H
;-----------------------------------------------
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
DEFW  W     16#0000
;*****************************************************************************
EM
```

6.9 Reference List

The reference list comprises a Table of Contents listing the available modules. The initial 2k words in the program memory are reserved for the reference list of the lower 64k word segment in the program memory. For each possible module, two words are defined in the reference list.

The entries for a given module are structured as follows:

Word 0	Address of first instruction and/or of first data word
Word 1	Module size in words, incl. 16 words for module header

A module that is available in the ZS35x/ZS40x/ZS50x is identified by these two entries.

For modules that are not available, each word contains the entry FFFFH.

The reference list is structured as follows:

Word offset of entry for start address	Module
0	DM0
510	DM255
512	PM0
1534	PM511
1536	OM0
2046	OM255

In the PLC program, for example, the reference list entries can be used to check whether modules are present or available and, helpful in the case of data modules, to check the size of a module.

Note
Although the ZS235x / ZS40x / ZS50x, upon loading and transferring data words, checks for the presence of the data module, it does not verify that the DM has been set up in sufficient size!

Utilization of a 128 k words program memory module

When using a 128 k words program memory module and/or the ZS40x, a second reference list covers the upper 64 k memory segment. It is located at the start of the second half of the memory area, i.e., at 64 kB.

⇒ **Because the program memory of the CL350 has a size of only 64 k words, it does not feature a second reference list.**

The seccond reference list is structured as follows:

Word offset of start address entry, referenced to the start of the second 64 k word program memory segment.	Module	Comment
0	DM256	currently not usable!
510	DM511	
512	PM512	
1534	PM1023	
1536	free	
2046		

The entries in the second reference list can be read with the use of the LMBX instruction.

Examples:

Reference list in lower 64 k memory segment

```
;Check whether or not a DM120 of a minimum length of 211 words
(through D420 at least) is available:
L    W  K120D,B     ; DM no. 120
SLL  W  B,1         ; x 2 (each module occupies 2 words)
INC  W  B,1         ; + 1 (returns the size, incl. header)
LMB  W  [B],B       ; Read size in words, incl. header
SUB  W  K16D,B      ; Subtract 16-word module header
CPLA W  K211D,B     ; DM size through D420 = 211 words
JPM     -DB_nio     ; smaller → not OK
```

Reference list in upper 64 k memory segment

```
;Check whether or not a DM511 of a minimum length of 211 words
(through D420 at least) is available:
L    W  K511D,B     ; DM no. 511
SUB  W  K256D,B     ; Subtract segment offset (256 DM)
SLL  W  B,1         ; x 2 (each module occupies 2 words)
INC  W  B,1         ; + 1 (returns the size, incl. header)
LMBX W  [B],B       ; Read size in words, incl. header
SUB  W  K16D,B      ; Subtract 16-word module header
CPLA W  K211D,B     ; DM size through D420 = 211 words
JPM     -DB_nio     ; smaller → not OK
```

⇒ **The reference list MAY NOT BE CHANGED by the user!**

6.10 OM5 & OM7 Startup Modules

Two startup modules, OM5 and OM7, are available.

If a startup module is linked with the PLC program, it will be automatically processed during the startup routine of the controller.

The start-up is governed by the following criteria:

OM5:
Startup module following restart, always processed subsequent to Power-On. This will apply also if, upon Power-On, the ZS35x / ZS40x / ZS50x is in STOP mode. In this case, the OM5 is processed upon changing the operating mode via a Stop/Run command.

Program loading:

For the ZS5xx, the program loading is followed by processing the OM7. In conjunction with the two trigger pulses, this facilitates the selection of any possible startup option.

For the ZS40x, depending on the version, the relevant startup module is contingent on the type of loading (firmware v2.2, PROFI PLC v4.50):

- Subsequent to loading with Reset of remanent operands: OM5
- Subsequent to loading without Reset of remanent operands: OM7

With regard to the instruction set, no restrictions exist in the startup module. In remanent operation it may be possible, however, that non-remanent operands are not predefined in the startup module. For more information, refer to Section 6.12, "Remanence Characteristics".

A cycle time ceiling that can be software defined via OM2, DW2/DW5 has not yet been implemented in the startup modules.

The hardware-dependent cycle time ceiling still applies.

Exceeding this limit will cause a system STOP.

In addition, the CL400 provides the option to fully disable the cycle time monitoring function during the startup module processing.

As a concluding instruction for the modules, the EP command is recommended.

For the ZS35x, ZS40x and, version-dependent also for the ZS500 / ZS501, the EM instruction may also be used. It should be noted that in this position there is no fucntional difference between EP and EM.

In the event that, during the processing of startup OMs, program modules are called, the close instructions of such program modules will have the established meaning:

EM**:**	Return to the startup module that included the call.
EP**:**	Cancel, continue with OM1.

6.11 OM9 Error Module

OM9 is the error module. If this module is linked with the PLC program, any error occurrences which would normally cause an immediate Stop of the central processing unit, the OM9 will be called automatically.

The same happens in all cases of errors which are also identified by setting a special marker bit in SM28 or SM29.

Exception: If no cycle time limit was designated in the software, and the hardware-dependent cycle time limit is reached due to a programming error, the entire control unit will automatically enter Stop mode.

In this case, enabling an error OM will no longer be possible.

The error module can be programmed with remedial measures to be launched in the event of an error occurrence.

For example, designated data, including the error bits in the system area, can be copied to nonvolatile memory areas.

As a follow-up, it is urgently recommended to program a HLT instruction in order to avoid consequential problems.

However, it should be stated that errors can also be acknowledged by resetting the respective special marker bit via indirect addressing.

The concluding program instruction in the OM9 must be EM.

The OM9 error module can be used in the CL350, CL400 and, in conjunction with the ZS501, also in the CL500.

6.12 Remanence Characteristics

The ZS35x, ZS40x and ZS550x central processing units feature a front panel slide switch. It is used to select the remanence characteristics:

- Remanence switch in "UP" position denotes remanent operation.
- Remanence switch in "DOWN" position denotes non-remanent operation.

6.12.1 Remanent Operation

In remanent operation, the statuses of the designated remanent operands are retained after a STOP / RUN and Power-On/Off mode change. As a precondition, no battery failure may exist.

In the absence of specific declarations in either OM2 or system area, this means that the following areas are remanent or non-remanent:

- The upper half of the marker range, i.e., M128 through M255, is remanent.
- The upper half of the data buffer, i.e., DB256 through DB511, is remanent.
- The upper half of the counters, i.e., C64 through C127, is remanent.
- The upper half of the timers, i.e., T64 through T127, is remanent.
- The entire data field, DF0 through DF24575 is always remanent, regardless of the remanence switch position.
- FIFOs are always remanent, regardless of the remanence switch position.
- All data modules are stored in the program memory module and are therefore always remanent, regardless of the remanence switch position.
- The MS timer of the CL350 / CL400 is always non-remanent.

The user can shift the stated remanence limits in accordance with his requirements. To this end, manipulation options exist in the OM2 and in the system range.

To ensure compatibility with the CL300, the remanence limits can also be modified in the OM5 and OM7 startup organization modules. Requiring appropriate entries in the system area, the remanence limits will then apply from OM1 and up.

This means: In remanent operation, once the startup OMs have been processed, the areas declared as non-remanent will be cleared.

This presents the following consequence:

In remanent operation, it is not possible to predefine non-remanent operands!

6.12.2 Non-remanent operation

In non-remanent operation, i.e., with the front panel remanence switch in the "DOWN" position, a STOP / RUN operating mode change or Power - Off/On cycle will be followed by clearing all markers, timers, counters, including flushing the data buffer.

This occurs even prior to processing any startup OM.

As a consequence, non-remanent operation permits predefining all operands in the startup OM.

The entire data field, DF0 through DF24575 is always remanent, regardless of the remanence switch position.

FIFO buffers are always remanent, regardless of the remanence switch position.

All data modules are stored in the program memory module and are therefore always remanent, regardless of the remanence switch position.

6.13 Fixation

The ZS35x, ZS40x, and ZS5xx central processing units provide the option to fix operands.

In contrast to the effect of the "Control" command of the PG, operands can be fixed permanently to specific bit statuses and/or values.

Fixable data range	Comment
Inputs	
Outputs	
Markers	without PG support

6.13.1 Remanence of Fixation

An established fixation remains enabled under the following conditions:

- Always after a STOP / RUN change of operating mode.
- Always after reloading (applies through PG version 4.50),
- After Power-Off/On only with Remanence slide switch set to "Rem" (UP).

6.14 Interrupts

The ZS35x, ZS40x, and ZS50x central processing unit utilize several groups of interrupts:

TI	Program interruption by means of time-controlled OM
SI	Programmunterbrechung durch Systembefehl
PI	Program interruption by means of a peripheral event – ONLY ZS35x / ZS40x / ZS501! For the ZS35x and ZS40x, an additional interrupt is provided for the ms timer. It has the lowest priority of all peripheral interrupts, and is handled by the OM42.

When an interrupt occurs, normal program execution is interrupted, and the associated interrupt module is activated.

The lowest priority is assigned to the group of timed interrupts, and the highest to the group of peripheral interrupts.

Within individual groups, the interrupt assigned to the lowest OM number has the highest priority.

6.14.1 Time Interrupts

For each time the time OM is called, the following must be true:

1. The designated time interval has expired, **and**

2. sequential processing has reached a change of module.

Neither a DM call-up nor an EP instruction is considered a change of module!

6.14.2 System Interrupts

A system interrupt is triggered by a system command, either in the local ZS or in a partner ZS.

A system interrupt is not linked to a module change but occurs instantly, i.e., subsequent to processing a given PLC program instruction, a jump occurs directly into the declared interrupt OM. In this process, the flag register, i.e., RES, etc., is rescued by the system. The user is responsible for effecting a possible rescue of register contents, scratch markers, etc.

6.14.3 Peripheral Interrupts

A peripheral interrupt is triggered by a 0→1 (LOW → HIGH) signal change on the associated input (EI1.0 through EI1.7) of the interrup input module.

As is true of the system interrupt, a peripheral interrupt is not linked to a module change. Instead, it branches into the respective interrupt OM immediately after processing a suitable instruction in the PLC program. In this process, the flag register, i.e., RES, etc., and the system register contents are rescued. The user is responsible for effecting a possible rescue of scratch markers, etc.

6.14.3.1 Hardware Prerequisites

The utiliziation of peripheral interrupts is possible in the **ZS35x**, **ZS40x**, and **ZS501 (not ZS500)**. It presupposes the use of the II24 interrupt input module. This module must occupy the peripheral addresses EI0, EI1 und EO0, EO1, and these addresses may not be used by other peripheral modules. The 8 interrupt inputs utilize the EI/EO Byte 1. In this context, the inputs of Byte 0 can be used as high-speed inputs.

Special requirements when using the ZS501:

The interrupt input module must be declared in the SC Table.

In the event that peripheral interrupts are used, and depending on the SK version, possible expansion units must be connected via AG/P-S expansion units and not via the SK500 system coordinator.

6.14.4 Interrupt Handling Instructions

The central processing control unit (ZS) internally assigns one interrupt mask each to all interrupt groups, time interrupts (TI) and peripheral interrupts (PI). The TIM and LIM instructions are used to read from and write to these masks. The mask associated with a given interrupt group contains one bit for each interrupt. A bit that is set to HIGH means that the respective interrupt is enabled; a bit that is LOW denotes that the corresponding interrupt is disabled.

To effectively enable the interrupts assigned in the mask, the additional EAI (Enable All Interrupts) instruction is required!

To generally disable an interrupt group without influencing the mask entries, the DAI (Disable All Interrupts) instruction is required.

Incoming interrupt signals cause an entry in the corresponding interrupt register, even though the corresponding interrupts are masked. Again, each interrupt is assigned one bit.

Again, each interrupt is assigned one bit.

If the interrupt is executable, i.e., enabled, the bit in the interrupt register will be automatically cancelled by the call-up of the interrupt OM.

If the interrupt is disabled, the bit will remain in the interrupt register while the interrupt is waiting to be enabled.

The interrupt register can be read with the LAI (Load All Interrupts) instruction, and waiting interrupts can be cancelled with the RAI (Reset All Interrupts) instruction.

During a change of operating mode with the use of STOP /RUN and/or Power-Off/On, all waiting interrupts are cancelled.

The masks for PI and SI are set to zero, and both system and peripheral interrupts disabled. Any required interrupts must be enabled be the user with the TIM and EI instructions!

Time interrupts are enabled by default.

During startup, i.e., while processing OM5 and OM7, all interrupts remain disabled.

Due to xconsiderations of compatibility, and beyond the interrupt handling instructions discussed so far, the CL300-specific EI, DI and RI instructions continue to be usable.

Special features when using the ZS35x and ZS40x:

In the interrupt mask for peripheral interrupts, the ms timer is assigned Bit 8. In the PLC program, the activation of the timer is effected with the Set MS Timer instruction, as follows: SMST R (R = register A, B, C or D). Parameterizaation is handled via the contents of register R.

6.15 Application Stack

The application stack (AST) comprises a pushdown-pop-up memory stack with a storage depth of 128 words, using FILO (first-in-last-out) processing.

The PUSH and POP instructions facilitate a word-by-word data transfer between the registers and the contents of the application stack.

Example:

```
PUSH W  A   ; Shift contents of register A to applic. stack
PUSH W  B   ; Shift contents of register B to applic. stack
PUSH W  C   ; Shift contents of register C to applic. stack
PUSH W  D   ; Shift contents of register D to applic. stack

POP  W  D   ; Load contents of applic. stack into Register D
POP  W  C   ; Load contents of applic. stack into Register C
POP  W  B   ; Load contents of applic. stack into Register B
POP  W  A   ; Load contents of applic. stack into Register A
```

In the event of an application stack underflow, special marker SM28.4 will be set to HIGH. In the case of an application stack overflow, special marker SM28.5 will be set to HIGH.

Both application stack (AST) underflow and overflow conditions will cause the central processing module to enter Stop mode.

The application stack is flushed after each EP!

7 Addressing Conventions for CL350 / CL400 / CL500

7.1 Operand & Module Identifiers, Module Lists

7.1.1 Operand & Module Identifiers

Abbr.	Operand	Peripheral access / data width	Image update
I	Input with image	Image/ bit, byte, word	in I/O state
II	Interface inputs I physically equal to II	direct/ byte, word	during PGM execution
EI	Extended input	direct/ byte, word	./.
O	Output	Image/ bit, byte, word	during PGM execution
IO	Interface outputs O physically equal to IO	direct/ byte, word	during PGM execution
EO	Extended output	direct/ byte, word	./.
M	Marker		
T	Timer		
C	Counter		
D	Data word, 1st curr. DM		
DX	Data word, 2nd curr. DM		
DF	Data field		
DB	Data buffer		
S	System area		
K	Constant		
DM	Data module	CM DMnn ; calls 1st active DM BX DMnn ; calls 2nd active DM	
PM	Program module		

7.1.2 Module Lists

The ZS35x, ZS40x, and each ZS50x manage the following program modules:

Name	Function	Comment
OM1	Cyclical program execution	
OM2	Initialization table	Refer to Section "OM2 Initialization Module"
OM5	Startup module after Power-ON	
OM7	Startup module after STOP/RUN	
OM9	Error module	ONLY ZS35x / ZS40x / ZS501
OM10	Peripheral interrupt module	assigned to EI1.0, highest priority
OM11	Peripheral interrupt module	assigned to EI1.1
OM12	Peripheral interrupt module	assigned to EI1.2
OM13	Peripheral interrupt module	assigned to EI1.3
OM14	Peripheral interrupt module	assigned to EI1.4
OM15	Peripheral interrupt module	assigned to EI1.5
OM16	Peripheral interrupt module	assigned to EI1.6
OM17	Peripheral interrupt module	assigned to EI1.7 lowest priority

Name	Function	Comment
OM18	Time-controlled module	Matrix declaration in OM2 or S18, highest priority
OM19	Time-controlled module	Matrix declaration in OM2 or S20
OM20	Time-controlled module	Matrix declaration in OM2 or S22
OM21	Time-controlled module	Matrix declaration in OM2 or S24
OM22	Time-controlled module	Matrix declaration in OM2 or S26 – availability is version-dependent
OM23	Time-controlled module	Matrix declaration in OM2 or S28 – availability is version-dependent
OM24	Time-controlled module	Matrix declaration in OM2 or S30 – availability is version-dependent
OM25	Time-controlled module	Matrix declaration in OM2 or S32 – availability is version-dependent, lowest priority
OM26	System interrupt module	
OM27	System interrupt module	
OM28	System interrupt module	
OM29	System interrupt module	
OM30	System interrupt module	
OM31	System interrupt module	
OM32	System interrupt module	
OM33	System interrupt module	
OM34	System interrupt module	ONLY ZS35x / ZS40x / ZS501
OM35	System interrupt module	
OM36	System interrupt module	
OM37	System interrupt module	
OM38	System interrupt module	
OM39	System interrupt module	
OM40	System interrupt module	
OM41	System interrupt module	
OM42	MST interrupt module	Time-controlled via ms timer, parameterization in MST start instruction. **ONLY CL350 and CL400**
PM0 through PM511	Program modules	
PM512 through PM1023	Program modules	In upper 64 k word segment of 128 k word PGM memory.
DM 0 through DM255	Data modules	
DM 256 through DM511	Data modules	Programming possible with WinSPS v2.10 and up. **NOTE: Data modules cannot be backed up to the MemoryCard.**

7.2 Special Marker Assignment

Each ZS35x, ZS40x, and ZS500 / ZS501 features a special marker area of 16-word size, i.e., SM0 through SM30.

It contains essential system flags and PLC cycle time information.

The unused addresses are reserved for internal system functions, and may not be changed.

The following list describes the extent of special markers applicable to the ZS35x, ZS40x, and ZS501.

Addr.	Contents	Comment
SM14	Special marker word for ms timer SM14.1-SM15.1 : Actual value SM15.2-SM15.6 : Overflow counter SM15.7 : Timer timeout bit	ONLY CL350 and CL400
SM16	Map of 16 CFS coordination marker bits (for one-time transmission enable) Enable / Disable via instructions: L Knnn,Reg TIM Reg,CFS	Write-only, ZS35x, ZS40x, ZS510 ZS501 not through v103 ZS500 from v203
SM18	Map of 16 CFP coordination markers (for permanent transmission enable) Enable / Disable via instructions: L Knnn,Reg TIM Reg,CFP	Read-only, ZS35x, ZS40x, ZS510, ZS501 ZS500 from v203
SM20	Bit field SM20.0 :Trigger pulse upon each startup SM20.1 : LOW BATTERY warning SM20.2 : Flashing marker SM20.3 : Outputs disabled SM20.4 : Fixiation enabled SM20.5 SM20.6 : Background buffer for system instructions is full SM20.7 : Trigger pulse after Power-On or Loading	Entire bit field is read-only ZS500 through v105 and ZS501 through v103: "BATTERIE" PM from MADAP required at program start.
SM21.0-SM21.7	Mailbox special marker	Read-only, ONLY ZS35x, ZS40x, ZS501
SM22	Actual cycle time for last complete cycle	Read-only
SM24	Maximum measured cycle time	
SM26	Minimum measured cycle time	

Addr.	Contents	Comment
SM28	Error word 1	ONLY ZS35x, ZS40x, ZS501; all errors read-only
	SM28.0 : Addressing error SM28.1 : Parameter error SM28.2 : Nonexistent module called SM28.3 : Module stack error SM28.4 : Application stack underflow SM28.5 : Application stack overflow SM28.6 : Attempted EPROM write-access SM28.7 : Disabled direct access	
	SM29.0 : Illegal write access SM29.1 : Opcode error SM29.2 : Error in parameter field System instructions SM29.3 : Field protection buffer full SM29.4 SM29.5 : No DM active SM29.6 SM29.7 : Cycle time error	
SM30	Error word 2	All errors read-only
	SM30.0 SM30.1 SM30.2 SM30.3 : fixed **0** SM30.4 : 64 k w segment flag, PC SM30.5 : 64 k w segment flag, DP1 SM30.6 : 64 k w segment flag, DP2 SM30.7 : 64 k w segment flag, PP	for internal Bosch FM only for internal Bosch FM only for internal Bosch FM only for internal Bosch FM only
	SM31.0 : Arith_gr_Flag SM31.1 : fixed **1** SM31.2 : SM31.3 : **C**arry_Flag SM31.4 : Log_gr_Flag SM31.5 : **O**verflow_Flag SM31.6 : Negative_Flag = **M**inus SM31.7 : **Z**ero_Flag	

7.3 System Area Assignment

Each ZS35x, ZS40x, and/or ZS500 / ZS501 features a system area of 256-word size, i.e., S0 through S510. It contains the system configuration data for the respective controller.

Essential specifications defined in the OM2 are copied into the system area, where they can be read by the PLC program.

To the extent deemed useful, the system declarations may be changed upon runtime. This also includes the time intervals of time-controlled organization modules.

In addition to the local ZS configuration data, the system area also contains configuration data of all intelligent modules within the CL400 / CL500 system. These are the modules that are entered in the SC Table and, in the case of the CL500, the SK500 itself.

Segments of the system area are used by default function modules which make data available that is also used by other PLC program parts. Examples are date and system clock.

The unassigned addresses in the system area are reserved for internal purposes, and may not be modified

Adr.	Contents	Comment
S0	Initialization flags, e.g., OM2_DW2	Writing in OM5 / OM7 *
S2	System bus handling, as in OM2_DW3	Read-only
S4	Error response, as in OM2_DW4	Writing in OM5 / OM7, ONLY ZS35x, ZS40x, ZS501
S6	Maximum cycle time, as in OM2_DW5	Writing in OM5 / OM7, ONLY ZS40x, ZS501
S8	DM to be copied, as in OM2_DW6	Read-only
S10	First remanent timer, as in OM2_DW7	Writing in OM5 / OM7
S12	First remanent counter, as in OM2_DW8	Writing in OM5 / OM7
S14	First remanent marker address, as in OM2_DW9	Writing in OM5 / OM7
S16	First remanent data buffer address, as in OM2_DW10	Writing in OM5 / OM7
S18	OM18 time interval, as in OM2_DW11	Transfer during startup and EP, possible active timer must expire befire new matrix will be activated.
S20	OM19 time interval, as in OM2_DW12	
S22	OM20 time interval, as in OM2_DW13	
S24	OM21 time interval, as in OM2_DW14	
S26	OM22 time interval, as in OM2_DW15	
S28	OM23 time interval, as in OM2_DW16	
S30	OM24 time interval, as in OM2_DW17	
S32	OM25 time interval, as in OM2_DW18	
S34		
S36		
S38	Module-specific deletion	ONLY ZS40x, writing in OM5 and OM7.
S40	of back ground instructions in STOP	
S42	Marked area: Startup / Load information	Byte-wise information, ONLY ZS501:
S44	Marked area: Startup / Load information	0=Power-On,
S46	Marked area: Startup / Load information	1=Restart,
S48	Marked area: Startup / Load information	2=Loading *

Addr.	Contents	Comment
S50 S52 S54 S56	Realtime: Minutes / Seconds Day / Hours Year / Month ----- / Day of Week	ONLY ZS35x and ZS40x, Read-only Entered by operating system 0=Sun, 1=Mon ... 6=Sat
S58 S112		
S114	Software switch RAM / MemoryCard RAM: 0000H MemoryCard: 0001H	ONLY ZS35x and ZS40x write / read
S116	Program memory module: Size / Type	ONLY ZS501, Read-only *
S118	Hardware / software version Hex format ZBHE Z = HW ver. auxiliary board (only ZS40x) B = HW version, basic board H = SW version, hundreds-digit E = SW version, units digit	ONLY ZS35x, ZS40x, ZS501 Read-only
S120	Local module number ZS40x: always 0	Read-only Row number in SC Table
S122	Initialization values for module 0	refer to itemization *
S124		
S126		
S128		
S130		
S132		
S134	Initialization values for module 1	refer to itemization
S136		
S138		
S140		
S142		
S144		
S146	Initialization values for module 2	refer to itemization
S148		
S150		
S152		
S154		
S156		
S158	Initialization values for module 3	refer to itemization
S160		
S162		
S164		
S166		
S168		
S170	Initialization values for module 4	refer to itemization
S172		
S174		
S176		
S178		
S180		
S182	Initialization values for module 5	refer to itemization
S184		
S186		
S188		
S190		
S192		

Addr.	Contents	Comment
S194	Initialization values for module 6	refer to itemization
S196		
S198		
S200		
S202		
S204		
S206	Initialization values for module 7	refer to itemization
S208		
S210		
S212		
S214		
S216		
S218	Initialization values for module 8	refer to itemization
S220		
S222		
S224		
S226		
S228		
S230	Initialization values for module 9	refer to itemization
S232		
S234		
S236		
S238		
S240		
S242	Initialization values for module 10	refer to itemization
S244		
S246		
S248		
S250		
S252		
S254 - S372		reserved
S374 S376 S378 S380	Time of last STOP switchover or Power-Off. Format as current realtime in S50-S56	ONLY ZS35x and ZS40x
S382- S394		
S396	System data field 1: Start address	Read-only, sizes in bytes
S398	System data field 1: Size	
S400	System data field 2: Start address	
S402	System data field 2: Size	
S404	System data field 3: Start address	
S406	System data field 3: Size	
S408	System data field 4: Start address	
S410	System data field 4: Size	
S412	System data field 5: Start address	
S414	System data field 5: Size	
S416	System data field 6: Start address	
S418	System data field 6: Size	
S420	System data field 7: Start address	
S422	System data field 7: Size	
S424	System data field 8: Start address	
S426	System data field 8: Size	
S428	System data field 9: Start address	
S430	System data field 9: Size	
S432	System data field 10: Start address	

Addr.	Contents	Comment
S434	System data field 10: Size	
S436	System data field 11: Start address	
S438	System data field 11: Size	
S440	System data field 12: Start address	
S442	System data field 12: Size	
S444	System data field 13: Start address	Read-only
S446	System data field 13: Size	
S448	System data field 14: Start address	
S450	System data field 14: Size	
S452	System data field 15: Start address	
S454	System data field 15: Size	
S456	System data field 16: Start address	
S458	System data field 16: Size	
S460	Mailbox 1: Start address	Read-only,
S462	Mailbox 1: Size	ONLY ZS35x, ZS40x, ZS501
S464	Mailbox 2: Start address	
S466	Mailbox 2: Size	
S468	Mailbox 3: Start address	
S470	Mailbox 3: Size	
S472	Mailbox 4: Start address	
S474	Mailbox 4: Size	
S476	Mailbox 5: Start address	
S478	Mailbox 5: Size	
S480	Mailbox 6: Start address	
S482	Mailbox 6: Size	
S484	Mailbox 7: Start address	
S486	Mailbox 7: Size	
S488	Mailbox 8: Start address	
S490	Mailbox 8: Size	
S492		
S494		
S496		
S498		
S500		
S502		
S504	Minutes / Seconds	Value Xfer from MADAP SW
S506	Day / Hours	Value Xfer from MADAP SW
S508	Year / Month	Value Xfer from MADAP SW
S510	... / Day of Week	Value Xfer from MADAP SW

⇒ **Supplementary notes on system area assignment:**

S0: Writing in OM5 / OM7:
- Bit1 = Check specified cycle time
- Bit2 = Assignment list-dependent peripheral operation, as per spec'd / actual assignment. ONLY ZS40x, see OM2 DW2
- Bit3 = Cycle time monit. during startup Yes / No ONLY ZS40x, see OM2 DW2
- Bit8 = Copy DM into data buffer

S42 through S49: marked area: Start / Load information:

The ZS501 writes a startup ID into each of these 8 bytes:

0=Power-On
1=Restart
2=Load

The referred ID bytes can then be used and also overwritten by the application for any purpose, e.g., as an extended trigger pulse.

ZS500 / ZS501:

Detailed itemization for **Byte** S116,

Size and type of program memory module:

Code	Module ID
80H	32k RAM
40H	64k RAM
30H	128k RAM
98H	32K EPROM
58	64K EPROM
38H	128K EPROM

Itemization of initialization values:

The SK500 and all modules listed in the SC Table are described by six words each in the system area of each ZS central processing unit.

In the case of the CL350, CL400, and CL500, the initialization values start with the module 0 that corresponds to the ZS35x, ZS40x and/or the SK500.

In the CL400 and CL500, this is then followed by the remaining modules, in the order in which they are listed in the SC Table.

The description of a module has the following structure:

- 1 word for module type ID
- 1 word for I/O information
- 1 word for module block address
- 1 word for hardware / software version
- 1 word for the number of assigned blocks on the system bus
- 1 word is reserved (spare)

Detailed itemization of I/O information:

Bit	15-14	Maximum number of permitted interrupt groups
Bit	13-11	Declared I/O size: 000 = 0 bytes 001 = 8 bytes 010 = 16 bytes 011 = 32 bytes 100 = 64 bytes 101 = 128 bytes 110 = 256 bytes
Bit	10-8	Declared EI/EO size, as in Bit 13-11
Bit	7-3	Peripheral start address in 8-byte matrix
Bit	2-1	currently 0
Bit	0	1 = Correctly entered module was recognized 0 = Correctly entered module was not recognized

7.4 System Data Fields

System data fields comprise data areas that are specifically designated for communication tasks.

In response to a system command, communication partner modules throughout the CL400 / CL500 sytem are capable of transferring data into the system data fields of various ZS40x or a given ZS50x.

In addition, external communication partners can use the BUEP19E protocol to write data into the system data fields of a ZS central processing unit.

If required, the system data fields must be set up within the data field (DF) onboard the ZS itself.

By declaring a system data field inside the data field, a relative addressing scheme, referenced to the system data field start address, is achieved. For communication partners, system command or BUEP19E protocol, the lowest system data field always has the address offset 0, regardless of which data field address represents the actual start of the system data field.

The referred declaration of system data fieldsoccurs in the OM2.

In the OM2, up to 16 system data fields can be defined. Their respective sizes are also declared at the same location; they range between a minimum of 128 and a maximum of 4096 bytes.

The field type for the data area within the system data field remains 43H.

As a field index, corresponding to the 16 possible system data fields, the values 0 through FH are permitted.

The system data fields are not protected by their own security mechanisms, such as the protective mailbox flags for the mailboxes, for example.

7.5 Mailboxes

Mailboxes comprise data areas that are specifically designated for communication tasks.

In response to a system command, communication partner modules throughout the CL400 / CL500 sytem are capable of transferring data into the mailboxes of various ZS40x or a given ZS50x.

In addition, external communication partners can use the BUEP19E protocol to write data into the mailboxes of a ZS central processing unit.

The transfer of data into the mailbox areas, when compared to a transfer into other destination data areas, provides the following special advantage: Once a communication partner module has written data into a mailbox onboard a ZS central processing module, this mailbox will be automatically protected against another write access. This condition is indicated to the PLC program by a HIGH mailbox protection flag in special marker SM21. A new write access will again be possible only if the PLC program explicitly removes the write protection. This is accomplished with the use of the FR (Field Release) instruction is used; (for more information about this subject, refer also to: Order no. 1070 072 137, "CL500 System Commands").

If required, the mailboxes must be set up within the local data field (DF) onboard the ZS central processing unit.

By declaring a mailbox field inside the data field, a relative addressing scheme, referenced to the mailbox start address, is achieved. For communication partners, system command or BUEP19E protocol, the lowest mailbox byte always has the address offset 0, regardless of which data field address represents the actual start of the mailbox.

The referred declaration of mailboxes occurs in the OM2.

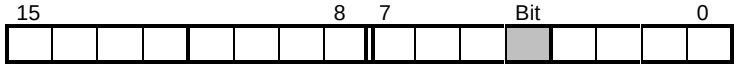
In the OM2, up to 8 mailboxes can be defined. Their respective sizes are also declared at the same location; they range between a minimum of 128 and a maximum of 4096 bytes.

The field type for the mailbox data area is 0H. As a field index, corresponding to the eight possible system data fields, the values 0 through 7 are permitted.

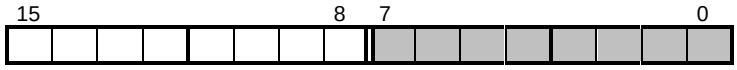
In the case of the CL500, the precondition for mailbox utilization is the use of the ZS501.

7.6 Data Formats

Bit = B



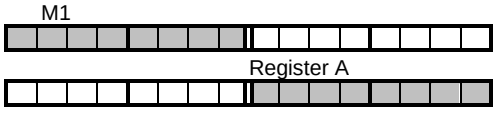
Byte = BY



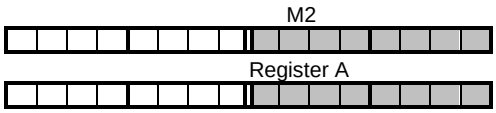
This addressing mode differentiates between load and transfer instructions:

Load instruction: The source operand may be either the even-numbered (LOW) byte or the odd-numbered (HIGH) byte.
In the case of the destination operand (register), the LOW byte is always addressed.

Examples: L BY M1,A

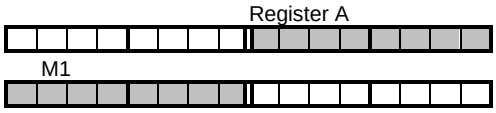


L BY M2,A

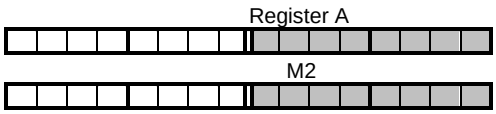


Transfer instruction: The LOW byte in the source operand (SRC_OPD = register) is addressed. The destination operand (DEST_OPD) may be both the even-numbered (LOW) byte and the odd-numbered (HIGH) byte.

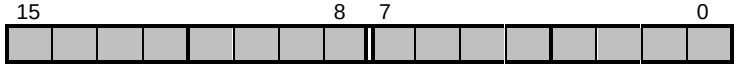
Examples: T BY A,M1



T BY A,M2



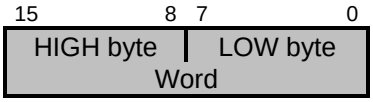
Word = W



7.7 Register Structure

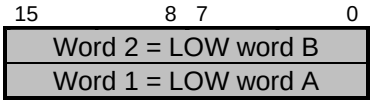
The CL350/400/500 features 4 working registers, which can be addressed in a bit-wise, byte-wise or word-by-word fashion. In this context, it should be noted that byte/word addressing always addresses the LOW-byte word.

Working registers A, B, C and D

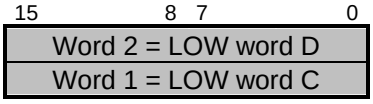


For operations that exceed the 16-bit format, the registers are combined to form permanent register pairs.

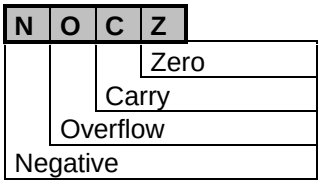
Working register pair A + B



Working register pair C + D



Status bits



⇒ **The negative flag always corresponds to the MSB (most significant bit) of the specified data format. Therefore, for byte operations, this is Bit 7, and for word operations, it is Bit 15.**

7.8 Representing Constants

The representation of constants is contingent upon the programming unit software being utilized. It has no functional bearing on the CL350/400/500.

Data Type		PLC utility programs	
Explanation	Notation	PROFI	WinSPS
UINT (unsigned integer)	Binary / Dual	K00000000 00000000B K11111111 11111111B	2#0000000000000000 2#1111111111111111
	Decimal, word	K00000D - K63535D	00000 - 65535
	Decimal, byte/byte	K000/000 - K255/255	in IEC1131 Teil 3 not defined
	Hexadecimal	K0000H - KFFFFH	16#0000 - 16#FFFF
INT (signed integer)	Decimal, word	K-32768 - K+32767 K-32768D - K+32767D	-32768 - +32767
Text, STRING(2)	ASCII	K'AB'	'AB'
Time value, TVALUE	Time val. (+timebase r) r: 0=10ms, 1=100ms 2=1s, 3=10s	K0.r - K1023.r	T#10ms - T#10230s T#0.r - T#1023.r

7.9 Program Module Calls

	PLC utility programs			
	PROFI		WinSPS	
Program module / function call (IEC1131-3)	CM	PM	CM	PM

7.10 Jump Instructions

	PLC utility programs			
	PROFI		WinSPS	
Jump instruction	JPx	-label	JPx	label
Jump destination	-label		label:	

7.11 Bit & Module Addresses

In the following table, please note that the CL350 features only 64 bytes of I/O and EI/EO.

Operand	Addresses (decimal)	Comment
I	0.0-63.7 64.0-127.7	ONLY ZS40x, ZS501; NOT FOR ZS35x
O	0.0-63.7	
M	0.0-255.7	
SM	0.0-31.7	
T status	0-127	
C status	0-127	
DM	0-511	**Note: DM256 through DM511 cannot be backed up to MemoryCard.**
PM	0-1023	

7.12 Byte / Word Addresses

⇒ **In the following table, please note that the CL350 features only 64 bytes of I/O and EI/EO.**

Operand	Address (decimal)	Comment
I	0-63	I 0-127 are physically identical to II 0-127!
	64-127	For ZS40x, ZS501 only, not for ZS35x
II	0-255	≥ 64 only for ZS40x, ZS501; not for ZS35x
EI	0-255	≥ 64 only for ZS40x, ZS501, not for ZS35x
O	0-63	O 0-63 are physically identical to IO 0-63!
IO	0-255	≥ 64 only for ZS40x, ZS501; not for ZS35x
EO	0-255	≥ 64 only for ZS40x, ZS501; not for ZS35x
T actual value	0-127	Time range, 10 ms-1023 s (matrix resolution 0.01; 0.1; 1; 10 s)
T status	0-127	
C actual value	0-127	Counter range, 0-8191
C status	0-127	
M	0-255	
SM	0-31	
S	0-511	
P	0-62	
DP	0-511	
DF	0-24575	
D	0-511	
DX	0-511	

7.13 Utilization of Extended I/O Range – ZS501 and CL400

(not applicable to ZS500 and CL350)

I/O byte number definition in CL500 SC Table, referenced to the I/O base address

ZS number	Block Address	I/O Base Address	Definable I/O blocks in bytes			
			0-63	64-127	128-191	192-255
ZS0	0	0				
ZS1	8	64				
ZS2	16	128				
ZS3	24	192				

OM2 entries:

Data word 1:	ZS number entry	
Data word 99:	Change entry	K4193H = Extended I/O range enabled
Data word 100:	Area definition	K7F00H = 0 bytes I/O
		K7F01H = 32 bytes I/O
		K7F02H = 64 bytes I/O
		K7F04H = 128 bytes I/O
		K7F08H = 256 bytes I/O

Optional:	Data words 37 - 48	Peripheral assignment list	I	bytes 64 - 255
	Data words 53 - 64	Peripheral assignment list	O	bytes 64 - 255
	Data words 69 - 80	Peripheral assignment list	EI	bytes 64 - 255
	Data words 85 - 96	Peripheral assignment list	EO	bytes 64 - 255

⇒ **The utilization of the extended I/O range of the ZS40x and ZS501 is restricted in the case of address ranges exceeding 1024 I and 512 O.**

Direct programming of bit addresses for
Inputs I0.0 - I127.7
Outputs O0.0 - O63.7

I/O situated above this range can be addressed only indirectly via the II and/or IO addresses.

Indirect addresses in the I/O, II/IO, and EI/EO ranges.

In the following tables, please note that the CL350 features only 64 bytes of I/O and EI/EO.

Bit addresses

Oper-and	Addresses Byte (dec.)	indirekt (hex)	Instructions .. B [Reg]	Comment
I	0.0-63.7	8400-85FF	A/AN/O/ON	
	64.0-127.7	8200-83FF		**Address jump!**
O	0.0-63.7	8600-87FF	A/AN/O/ON S/R/=	

Example of indirect bit addressing

```
L     KiadrH,A
A B   [A]
=     M10.0
```

Word addresses

Oper-and	Address (decimal)	Indirect address (hex)	Comment
I	0.0-63.7	0080-00BF	
	64.0-127.7	0040-007F	**Address jump!**
II	0.0-255.7	0A00-0AFF	
EI	0.0-255.7	0E00-0EFF	
O	0.0-63.7	00C0-00FF	
IO	0.0-255.7	0C00-0CFF	
EO	0.0-255.7	1000-10FF	

7.14 Addressing Modes

7.14.1 Direct Addressing

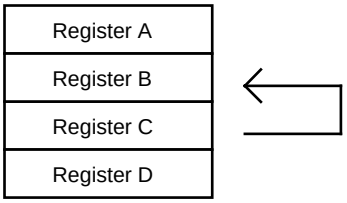
Operands for absolute addressing

Bit-addressable	I, O, M, SM, T and C	for T/C, status applies
Byte/word readable	I, O, M, SM, T and C K, DP, DF, D, DX, S, II, EI, FI	for T/C, actual value applies
Byte/word writable	O, M DB, DF, D, DX, S, IO, EO, FI	

Direct addressing of all absolute-addressable operands

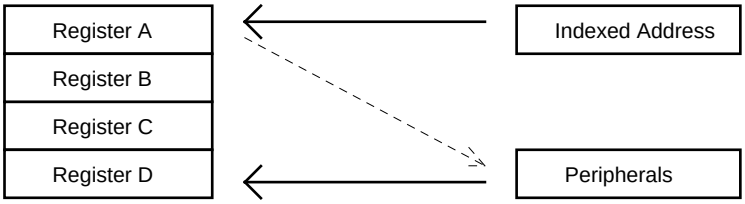

```
Examples: L   BY   I10,B     ; Loads the status of input byte I10 into B.
          L   W    K100,C    ; Loads the value 100 into register C.
```

7.14.2 Register-to-Register Addressing




```
Example:  L   W    C,B       ; Loads the contents of register C into
                             ; register B.
```

7.14.3 Register-indirect Addressing




```
Examples: L   W    &I10,A    ; Loads index address of input byte I10
                             ; into register A.

          L   W    [A],D     ; Loads the status of I10 (address in A)
                             ; into register D.
```

7.14.4 Indirect Addresses

7.14.4.1 Indirect Module Addresses

Operand	Addresses Module no.	indirect (hex)	Address Range	Instructions ... [Reg]	Comment
DM	0-255 256-511	0000-00FF 8000-80FF	0-255 256-511	CMx BXx	**Address jump, effective with WinSPS v2.10 and up!**
PM (FC) **PM**	0-511 512-1023	0100-02FF 8100-82FF	0-511 512-1023	CMx CMx	**Adress jump!**

Example of indirect module call:
DM 0-255:
L Kdb_adrH,A
CM [A]

To address the next module based on a starting address, the address must be incremented by a count of 1.

⇒ **The data modules DM256 - 511 and the program modules FC512 - 1023 are stored in the 2nd memory segment onboard the control unit. There, they are treated as DM0 - 255 and FC0 - 511, respectively.**

Example 1: Data module DM480 ($1E0_H$) is to be called indirectly.

```
L    16#81E0,A   ; Base address 8000H + module address 1E0H
SUB  256,A       ; Subtract DM number offset
CM   [A]         ; Call module DM480 = DM224 in 2nd segment
```

Example 2: PGM module FC980 ($3D4_H$) is to be called indirectly.

```
L    16#84D4,A   ; Base address 8100H + module address 3D4H
SUB  512,A       ; Subtract DM number offset
CM   [A]         ; Call module FC980 = FC468 in 2nd segment
```

7.14.4.2 Indirect Bit Addresses

In the following table, please note that the CL350 features only 64 bytes of I/O and EI/EO.

Operand	Addresses Byte (dec.)	indirect (hex)	Address Range	Instructions .. [Reg]	Comment
I	0.0-63.7 64.0-127.7	8400-85FF 8200-83FF	0-511 512-1024	A/AN/O/ON	**Adress jump!**
O	0.0-63.7	8600-87FF	0-511	A/AN/O/ON S/R/=	
M	0.0-255.7	8800-8FFF	0-511	A/AN/O/ON S/R/=	
SM	0.0-31.7	8100-81FF	0-511	A/AN/O/ON	
T status	0-127	8080-80FF	0-127	A/AN/O/ON	
C status	0-127	8000-807F	0-127	A/AN/O/ON	

Examples of indirect bit addressing:
L KiadrH,A
A B [A]
= M10.0

L KiadrH,A
A M10.0
S B [A]

To address the next bit based on a starting address, the address must be incremented by a count of 1.

7.14.4.3 Indirect Byte / Word Addresses

In the following tables, please note that the CL350 features only 64 bytes of I/O and EI/EO.

L / T instruction (uses the byte addresses)

OPD ID	Address (decimal)	&-OPT	Indirect Byte address (hex)	Load Transfer	Comment
I	0.0-63.7	x	0080-00BF	L + T	
	64.0-127.7	x	0040-007F	L + T	**Address jump!**
II	0.0-255.7	x	0A00-0AFF	***1** L	L of II ... describes the image.
EI	0.0-255.7	x	0E00-0EFF	***1** L	
					***1** Write-access causes ZS to enter STOP mode.
O	0.0-63.7	x	00C0-00FF	L + T	
IO	0.0-255.7		0C00-0CFF	***2** T	T to IO... describes the image.
EO	0.0-255.7		1000-10FF	***2** T	
					***2** Read-access causes ZS to enter STOP mode.
T actual	0-127	x	0700-07FF	***1** L	***1** Write-access causes ZS to enter STOP mode.
C actual	0-127	x	0600-06FF	***1** L	***1** Write-access causes ZS to enter STOP mode.
M	0.0-255.7	x	0100-01FF	L + T	
SM	0.0-31.7	x	0020-003F	L + T	Write-access to SM30/31 disabled.
S	0-511	x	0400-05FF	L + T	
P	0-62		1200-123E	L + T	Writing constants, and write-access to areas marked with "*2", as well as read-access to areas mareked with "*1" will cause ZS to enter STOP mode.
DB	0-511	x	0200-03FF	L + T	
DF	0-24575	x	A000-FFFF	L + T	
D	0-511	x	0800-09FF	L + T	
DX	0-511	x	1800-19FF	L + T	

To address the next byte based on a starting address, the address must be incremented by a count of 1.
To address the next word, the address must be incremented by a count of 2.

The "&" operator is used to facilitate the programming of indirect addresses, and corresponds to the encoded loading of address constants.
Example: L &D0,A ≡ L K0800H,A

LIMR / TIMR instruction (utilizes absolute word addresses)

O ID	Address (decimal)	Absolute Word Addr. (hex)	Load Transfer	Comment
II	0.0-255.7	0500-057F	L + T	
EI	0.0-255.7	0700-077F	L + T	
IO	0.0-255.7	0600-067F	L + T	
EO	0.0-255.7	0800-087F	L + T	
T actual	0-127	0380-03FF	L + T*	Bit 0-9 =value Bits 10+11 = matrix Bit 13 =Run Bit 14 = Halt Bit 15 = Start * writing to flags causes unpredictable timer behaviour → no TIMR
C actual	0-127	0300-037F	L + T	Bit 0-12 = value Bit 13 = CD Bit 14: = CU Bit 15 = set C
M	0.0-255.7	0080-00FF	L + T	
SM	0.0-31.7	0010-001F	L + T	LIMR of SM 30,31, loads undefined range
S	0-511	0200-02FF	L + T	
DB	0-511	0100-01FF	L + T	
DF	0-24575	5000-7FFF	L + T	

To address the next word based on a starting address, the address must be incremented by a count of 1.

Programming Examples

```
;Direct status processing
;Operand defined directly for II,EI,IO,EO
 LIMR W  II0,A
 TIMR W  A,II0

;Operand defined as direct address
;for all operands
LIMR W  $opd-abs,A
TIMR W  A,$opd-abs

;Indirect status processing
;for all operands
;Operand defined as indirect
;byte address
L       W  &IE0,A ; Indirect byte address
SLR   W  A,1    ; :2=absolute addr. IE0
LIMR W  [A],B   ; Load status

;Operand defined as absolute
;word address
L    W  K500H,A ; Load absolute IE0 address
L/MR W  [A],B   ; Load status
```

7.15 Parameterized Modules

When a program module is called up, up to 63 parameter values can be transferred. The number of parameter values to be transferred is stated as part of the module call-up instruction, followed by the actual parameters, starting with the number P0.

All parameters that are to be used as a byte or word in the program module being called up are transferred without operand attribute.

(Depending on the version of the programming unit (PG) being used, the operand attribute **BY** or **W** may be included, depicting that no operand attribute is being used.)

Likewise, program modules (PM) or data modules (DM) that are transferred as a parameter value do not include an operand.

All parameters to be used as bits in the module being called up are transferred with the operand attribute **B**!

Exception:
If times and counters are transferred in the form of parameters without operand attribute, they may be utilized as both a word function, i.e., time/counter value, and/or a bit function, i.e., time/counter status in the module being called up.

Example of parameter transfer:

```
CM      PM100,7    ;Call up PM100 and transfer 7 parameters
P0      43         ;Parameter P0: PM no. as decimal constant K43
P1      4          ;Parameter P1: DM no. as decimal constant K4
P2      O56        ;Parameter P2: Output word with byte address O56
P3      I7.3       ;Parameter P3: Input bit E7.3
P4      T2         ;Parameter P4: Time T2
P5      C13        ;Parameter P5: Counter C13
P6      O10.0      ;Parameter P6: Output bit O10.0
```

Utilization of parameters in called-up module PM100

```
CM      P1         ;Open DM4
BX      DM5

L       P0,A       ;Load PM no. 43
CM      PB[A],2    ;Call up PM43 and transfer two parameter values
P0      D2         ;Parameter P0: D2 of active 1st DM, this being DM4
P1      DX6        ;Parameter P1: DX6 of active 2nd DM, this being DM5

L    W  P2,A       ;Load output word O56

L    W  P4,B       ;Load time value from T2 to B

A    BY  P3         ;E7.3
A    BY  P4         ;Status of T2
A    BY  P5         ;Status of C13
=    BY  P6         ;O10.0
```

7.16 Programming the 1-ms Timer for ZS35x / ZS40x

The ms timer in the CL350 and CL400 runs concurrent with the application program with a basic accuracy of 0.1 ms. To keep the function as flexible as possible, internal transition control of the timer was not used.

Control bits can be used to define whether or not a time matrix is to be used, and which time resolution the actual value is to be transferred into the associated special marker. From that location, it can be queried via the high-speed gate array instructions. To load the accurate current value, the CL350 / CL400 provide a command which facilitates processing at a 0.1 ms resolution.

Mnemonics of the time start / control command

SMST W R ; R = Register A, B, C, D

Mnemonics of the time start / control command

LMST W R ; R = Register A, B, C, D

Explanation of timer handling register bits

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Timer functional principle				SM update		Timer specified value									
				NfB2	NfB1	10 bits in ms-matrix ⇒ MAX timer value = 1023 ms									
				0	0	= none matrix for automatic actual-value updating									
				0	1	= 1 ms in the special marker bytes SM14/15									
				1	0	= 5 ms									
				1	1	= 10 ms									
			Int/Flag = 0 Timeout sets special marker bit SM15.7												
			= 1 Timeout calls up OM42 interrupt module												
		Wrap/Run = 0 Time stands still upon timeout. Depending on Bit 12 = Int/Flag, the OM42 interrupt module is called, or special marker SM15.7 goes HIGH. The remainder of SM14/15 is cleared.													
		= 1 Upon timeout the timer restarts automatically, and the time counter SM15.2 - 15.6 is incremented. The counter stops at a maximum value of 31, until it is again cleared by a Reset. The timer continues to "wrap around"..													
	Reset = 0 Start / Stop cause timer to continue / causes timer to stop.														
	= 1 The actual value and the special marker bytes SM14/15 go LOW.														
Start/Stop = 0 Timer stops															
= 1 Timer restarts and/or continues															

Explanation of SM14/15 special marker bits:

SM15.								SM14.							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
expir'd	Timer counter					Actual timer value, updated as per matrix definition									

⇒ **Previous ZS400 firmware versions through release v1.1 exhibit an error in PG monitoring. As a consequence, the register displays (functioning is assured), upward of displaying the aforementioned commands, are no longer correct. However, the resulting display fault can be easily corrected by scrolling the SMST and/or LMST instruction out of the screen in an upward direction.**
Better still, when using the "MS_ZEIT" function module described below, this problem won't occur in the first place!

7.17 FIFO Instructions

The ZS35x, ZS40x, and ZS50x provide four FIFO buffers, designated FI0 through FI3.

Each FIFO buffer has a size of 512 bytes.

Reading from and writing to the FIFO buffers is accomplished with the LFI and TFI instructions.

A single instruction reads or writes 1 to 32 bytes.

The number of bytes to be handled by means of the LFI / TFI instruction is variable, and is declared in Register C.

Exception: In the event that register contents are written to or read from FIFO buffers, the number of bytes will be defined via the operand attribute W/BY. Accordingly, operand attribute BY = one byte; operand attribute W = two bytes.

When the number of bytes to be handled is variably declared in Register C, each FIFO byte that is read or written causes the value in Register C to be decremented.

In the case of a FIFO buffer overflow or underflow, the value stored in Register C provides information about the number of bytes that could no longer be read or written.

FIFO overflow or underrun will not automatically cause a ZS STOP.

As an indication of a FIFO overflow, carry bit SM31.3 goes HIGH. A FIFO underrun causes zero bit SM31.7 to go HIGH.

The FIFO buffer is flushed with the RFI (Reset FIFO) instruction.

FIFO buffers are always remanent, regardless of the position of the remanence switch.

All FIFO instructions are RES-independent.

7.18 Block Commands

Block commands provide a comfortable means of loading and transferring, as well as comparing and searching, data blocks within a ZS35x, ZS40x or ZS501. The maximum size of these data blocks is 256 words / 512 bytes. The operand attribute indicates whether the block size refers to byte size or word size.

Comm'd	Attribute	Explanation	Note
Block transfer: Shifting of data blocks of defined size, whereby the data blocks may not overlap.			
BLT	W/BY	Block transfer	
Block Compare: Compares two data blocks. When the Compare condition is met, processing is stopped, and the addresses are returned in Registers A and B. Register C contains the number of bytes/words that were not searched. The zero bit goes HIGH when the Compare condition was not met in the entire area.			
		Compare forwd. for:	8 Compare criteria based on the interpretation of flags C, M and Z, plus combinations thereof. The execution of these commands presupposes a flag-influencing instruction (logical Compare oper.).
CFZ	W/BY	equal	
CFN		unequal	
CFAG		arithmetical greater	
CFM		arithmetical less	
CFLG		logical greater	
CFCY		logical less	
CFCN		logical greater than or equal	
CFCZ		logical less than or equal	
		Compare backwd. for:	
CBZ	W/BY	equal	
CBN		unequal	
CBAG		arithmetical greater	
CBM		arithmetical less	
CBLG		logical greater	
CBCY		logical less	
CBCN		logical greater than or equal	
CBCZ		logical less than or equal	
Block search: Search for character in a data block. If the character is found, its address will be written to Register A. Register C contains the number of bytes/words that were not searched. The zero bit goes HIGH when the Compare condition was not met in the entire area.			
		Character search forwd. for:	8 Compare criteria based on the interpretation of flags C, M and Z, plus combinations thereof. The execution of these commands presupposes a flag-influencing instruction (logical Compare oper.).
SFZ	W/BY	equal	
SFN		unequal	
SFAG		arithmetical greater	
SFM		arithmetical less	
SFLG		logical greater	
SFCY		logical less	
SFCN		logical greater than or equal	
SFCZ		logical less than or equal	
		Character search backwd. for:	
SBZ	W/BY	equal	
SBN		unequal	
SBAG		arithmetical greater	
SBM		arithmetical less	
SBLG		logical greater	
SBCY		logical less	
SBCN		logical greater than or equal	
SBCZ		logical less than or equal	

For more details on this subject, refer to Section 8.14.2, "Block Commands", and Section 9.3, "Sample Applications of Block Commands".

7.19 Floating-point Arithmetic

Effective with **WinSPS v2.40**, floating-point arithmetic was integrated into the Instruction set. Accordingly, floating-point instructions can now be entered in a manner reminiscent of the pocket calculator.

Up to **WinSPS v2.30**, floating-point arithmetic is handled by function modules.

For the ZS35x, ZS40x, and ZS501, optimized function modules are available. These operate with firmware support and utilize the IEEE format.

The following floating-point modules are available for the ZS35x, ZS40x, and ZS501:

Module name	Fixed-point format, processing times			Floating-point format
GLPI16S *)	Integer 16, 16 bits including sign			single precision, 32Bit
	FG:	Fixed/floating-point	100 µs	
	GF:	Floating/fixed-point	90 µs	
	AD:	Addition	150 µs	
	SU:	Subtraction	150 µs	
	MU:	Multiplication	140 µs	
	DI:	Division	230 µs	
	VG:	Compare	90 µs	
GLPI32S *)	Integer 32, 32 bits, including sign			
	FG:	Fixed/floating-point	130 µs	
	GF:	Floating/fixed-point	130 µs	
	AD:	Addition	150 µs	
	SU:	Subtraction	150 µs	
	MU:	Multiplication	140 µs	
	DI:	Division	230 µs	
	VG:	Compare	90 µs	
GLPU16S *)	unsigned 16, 16 bits, excluding sign			
	FG:	Fixed/floating-point	100 µs	
	GF:	Floating/fixed-point	90 µs	
	AD:	Addition	150 µs	
	SU:	Subtraction	150 µs	
	MU:	Multiplication	140 µs	
	DI:	Division	230 µs	
	VG:	Compare	90 µs	
GLPU32S *)	unsigned 32, 32 bits, excluding sign			
	FG:	Fixed/floating-point	135 µs	
	GF:	Floating/fixed-point	130 µs	
	AD:	Addition	150 µs	
	SU:	Subtraction	150 µs	
	MU:	Multiplication	140 µs	
	DI:	Division	230 µs	
	VG:	Compare	90 µs	

*):
As these modules are self-manipulating at runtime, they are recognized by the PG programming unit as "not identical".

This effect is unavoidable, and may be safely neglected.

The referred modules can be utilized in the ZS501 only in conjunction with the program memory modules.

7.20 Internal Operating Functions – CL350 and CL400

A new instruction was introduced for the CL350 / CL400 controllers. It facilitates access to IBF modules, a type of module that is stored in the firmware of the central processing unit, and which are addresses with the CF function call.

The option of a firmware update via the MemoryCard facilitates a fast and easy extension of the IBF modules.

Availability

Minimum version requirements:

Pogramming unit:	PROFI software, WinSPS software,	v4.10 v1.10
ZS350 / ZS351	Firmware,	v1.01
ZS400 / ZS401	Firmware,	v2.01

IBF module list

IBF 1 Writing data modules copied from PLC program memory to MemoryCard.

IBF 2 Writing data modules from MemoryCard into the PLC program memory.

7.20.1 Backing Up & Loading Data Modules

The "**Backup / Load Data Modules**" command is used to save current system data in a MemoryCVard, and to restore such data subsequent to a system STOP.

Other data areas, such as markers or data buffer, cannot be backed up. Therefore, all data representing the system condition must be stored in data modules.

In the past, a controller downtime due to buffer failure (battery failure), or a hardware defect in the central processing unit and/or power supply module always necessitated a complete reload of the PLC program once the hardware was replaced. Heretofore, this process also caused the data modules to be overwritten with the default settings, i.e., current data was lost.

Now, with the "**Back Up / Load Data Modules**" command, the user has the option to facilitate a continuous backup of the data modules and, in the case of a failure, to restore the data that was current at the time the failure occurred. The storage medium used by the CL350 / CL400 for this purpose is a battery-backed SRAM MemoryCard. A 512 kB SRAM MemoryCard can be ordered with Part no. 1070 918483.

7.20.2 Using the MemoryCard for Data Backup

7.20.2.1 Preparing the MemoryCard

An SRAM MemoryCard to which the current contents of data modules are to be backed up must contain a valid directory of the following type:

- "Complete PLC Program with SC Table"
- "Complete PLC Program without SC Table"
- "PLC Program Module (0-128 kbytes) with Data Modules
- "PLC Data Modules Program Module"

These directories are generated upon backing up the PLC program via pushbutton commands on the ZS35x / ZS40x.

7.20.2.2 Backing Up PLC Programs to MemoryCard

CAUTION – Risk of Data Loss!
When backing up PLC programs to a MemoryCard, the entire card contents will be deleted! The user must therefore ensure that it is permissible to delete the original contents!

⇒ **The "Save PLC Program To MemoryCard" will be permitted only if there is no simultaneous access to the program memory via the system bus by the programming unit (Monitor service program) or another module!**

⇒ **A MemoryCard swap is not permitted during RUN operation of the PLC!**

Pushbutton operations on the ZS35x / ZS40x

★ Using the slide switch on the front panel, switch the central processing unit to **STOP**. The 7-segment display will indicate **0 F**.

★ Insert MemoryCard into central processing unit.

★ Press **SCROLL** key until display indicates **.S 0**.

★ Actuate the **DISABLE OUTPUT** switch. For this purpose, the actual switch position is of no consequence.

In the event that the **DISABLE OUTPUT** switch is not actuated within 5 seconds, the central processing unit will return to STOP mode.

★ Press the **SCROLL** key until the desired backup command appears in the 7-segment display:

0 **Entire PLC program with data modules and SC Table.**

1 **PLC program module (0-128 kbytes), including all organization modules, program modules 0-511 (system program modules), and all data modules.**

2 **PLC program module (128-256 kbytes), including program modules 512-1023 (default program modules).**

3 **PLC Data Modules Program Module.**

4 **Entire PLC program with data modules but without SC Table.**

★ Actuate the **DISABLE OUTPUT** switch. For this purpose, the actual switch position is of no consequence.

The backup process is started and may take between 5 and 10 seconds. Upon conclusion of the backup procedure, the central processing unit enters into **STOP** mode.

The red **Stop** LED illuminates. The 7-segment display indicates **0 F**.

★ Using the slide switch on the front panel, return the central processing unit to **RUN** mode.

Error message during backup

If the backup process cannot be completed properly, the 7-segment display will indicate **.5 F**.

Although this message display will remain even when the central processing unit has returned to **RUN** mode, it can be cancelled by pressing the **SCROLL** key.

7.20.2.3 Loading the PLC Program from a MemoryCard

Loading the PLC program from a MemoryCard will be possible only if there is no simultaneous access to the PLC program memory via the system bus or by the PG programming unit (Monitor mode). If that is the case, the execution of the command will be rejected, and the 7-segment display will indicate **.L F**.

CAUTION – Risk of Data Loss!
When loading a PLC program from a MemoryCard, the original contents of the PLC program memory will be deleted. The user must therefore ensure that it is permissible to delete PLC program! Therefore, it is good practice to back up the PLC program by copying it to an additional MemoryCard or onto a PG programming unit!

Pushbutton operations on the ZS35x / ZS40x

★ Using the slide switch on the front panel, switch the central processing unit to **STOP**. The 7-segment display will indicate **0 F.**

★ Insert MemoryCard into central processing unit.

★ Press **SCROLL** key until display indicates **.L 0**.

★ Press the **SCROLL** key until the desired load command appears in the 7-segment display:

0 **Entire PLC program with data modules and SC Table.**

1 **PLC program module (0-128 kbytes), including all organization modules, program modules 0-511 (system program modules), and all data modules.**

2 **PLC program module (128-256 kbytes), including program modules 512-1023 (default program modules).**

3 **PLC Data Modules Program Module. To execute this command, the MemoryCard must be of the Data Module type.** The content of all data modules is overwritten. The reference list and the data module headers remain unchanged. In the event that the data modules on the MemoryCard and in the PLC program differ in size, the loading process will be aborted, and the display will indicate **0 I**. Any data modules that are present on the MemoryCard but not in the PLC program memory will not be loaded. Data modules that are not present on the MemoryCard but are contained in the PLC program memory remain unchanged.

4 **Entire PLC program with data modules but without SC Table.**

★ Actuate the **DISABLE OUTPUT** switch. For this purpose, the actual switch position is of no consequence.

The loading process is started and may take between 5 and 10 seconds. Upon conclusion of the loading procedure, the central processing unit enters into **STOP** mode.

The red **Stop** LED illuminates. The 7-segment display indicates **0 F**.

★ Using the slide switch on the front panel, return the central processing unit to **RUN** mode.

Error message during loading

If the backup process cannot be completed properly, the 7-segment display will indicate **L F**.

Although this message display will remain even when the central processing unit has returned to **RUN** mode, it can be cancelled by pressing the **SCROLL** key.

★ Actuate the **DISABLE OUTPUT** switch. For this purpose, the actual switch position is of no consequence.

In the event that the **DISABLE OUTPUT** switch is not actuated within 5 seconds, the central processing unit will return to STOP mode.

7.20.2.4 Examples of Using the Backup / Load Data Modules Command

Backing up data modules into a MemoryCard containing the entire PLC program

The objective is to back up the "**Entire PLC program with data modules and SC Table**" to an SRAM MemoryCard, as described above. In response to the "**Back Up Data Modules**" function call, the PLC program cyclically copies the changeable data to the referred MemoryCard.

Subsequent to a buffer failure accompanied by loss of program, the system status can be fully restored with the use of this MemoryCard. This is accomplished by initiating the "**Entire PLC program with data modules and SC Table**" copy option.

Backing up data modules to a MemoryCard containing only data modules

The backup and load procedure can also be spread over two MCs, whereby one MemoryCard accommodates the entire PLC program with the default values, and the other MemoryCard contains data modules holding the current system data.

This is accomplished by backing up the "**Entire PLC program with data modules and SC Table**" to a MemoryCard (this may also be a FLASH Memory card), as described above. The second MemoryCard is then inserted and, using the pushbutton commands on the front panel of the central processing unit, the "**PLC Data Modules Program Module**" backup option is executed. In response to the "**Back Up Data Modules**" function call, the PLC program cyclically copies the changeable data to the second MemoryCard.

Subsequent to a buffer failure accompanied by loss of program, the system status can be fully restored with the use of the default values from the first MemoryCard. This is accomplished by initiating, via the central processing unit, the "**Entire PLC program with data modules and SC Table**" option. This can then be followed by using the second MemoryCard to restore the previous system status with the use of the "**Load Data Modules Only**" command.

Backing up data modules to several MCs containing data modules holding a variety of different data records (formulation/batch data)

The previously described backup procedure can also be used to create differing formulation/batch data MCs. This is accomplished by setting up various MCs with different formulation/batch data module contents. The formulation data is later transferred into the PLC in the following manner:

PLC is switched to STOP mode. The desired formulation MemoryCard is inserted, and the PLC is switched to RUN mode. In startup module OM7, the data modules specified for the CF "**Read Data Modules**" command will then be loaded. All other data modules remain unchanged.

Program-controlled loading of initialization or start values into the PLC program

Provided that the data modules containing the initialization values were backed up to a MemoryCard as described above, the CF "**Read Data Modules**" command can then be used for event-controlled loading of these values into the PLC program

⇒ **To accomplish the loading of data modules, the MemoryCard must be installed in the MemoryCard slot in the control unit. In the event that, instead of the Backup Data Module, any other MemoryCard of the "Data Modules Program Module" type is slotted, an error will be detected only if the data module numbers to be loaded fail to match or exhibit different sizes. Otherwise, a situation may occur in which wrong and/or garbled data are read as a system status.**

7.20.3 Programming the Function Call

7.20.3.1 Command Processing

The CF function call command is executed immediately, i.e., directly during program processing, and without precondition.

The command provides the option to back up up to 64 data modules per function call, whereby the PLC program processing time is extended, depending on the number and size of data modules to be transferred. For this reason, the number and size of data modules should be limited to the bare minimum.

7.20.3.2 Command Mnemonics

To back up data modules to the MemoryCard, the following applies:

```
CF 1,»Adr«
```

To load data modules from the MemoryCard, the following applies:

```
CF 2,»Adr«
```

7.20.3.3 Addressing Modes

The **»Adr«** option is used to define the addressing mode to be used by the command.

Parameter field trailing the command

»Adr« indicates the number of parameters to follow.

The data module list must then be defined as a constant, as a´n absolute data module address, or as a symbolic name.

Example:

```
CF    1,3          ; CF 1 = Copy data modules to MemoryCard
P0    K1           ; Back up DM1,
P1    DM3          ; DM3
P3    -DMNr7       ; and -DMNr7.
```

Base address of parameter field in register

»Adr« indicates the address of the parameter field in which the data module numbers are stored, with each DM number occupying one word. The number of parameters is returned in Register C.

Example:

```
CM    DM10
L     K1,D0        ; Back up DM1,
L     17,D2        ; DM17
L  W  K22,D4       ; and DM22

L  W  K3,C         ; Number of parameters = 3
L  W  &D0,A        ; indirect address of parameter field
                   ; in DM10, starting at D0

CF    1,[A]        ; CF 1 = Copy data modules into MC.
```

Direct base address of parameter field

»Adr« indicates the first operand of the parameter field in which the data module numbers are stored in the form of values. The number of parameters is returned in Register C.

Example:

```
L     W      K7,M10       ; DM7
L     W      K11,M12      ; DM11
L     W      K33,M14      ; DM33
L     W      K44,M16      ; DM44
L     W      K55,M18      ; DM55

L     W      K5,C         ; Number of parameters = 5

CF    1,M10               ; CF 1 = Copy DMs into MC
```

7.20.3.4 Number of Parameters

The number of parameters is limited to 64. Depending on the addressing mode, it is returned directly or in Register C.

7.20.4 Processing Times

The following processing times apply to the "**Back Up / Load Data Modules**" function call:

Basic interval: approx. 0.1 ms per data module

Offset interval: approx. 2.4 µs per byte

This processing interval ascends in a linear progression with the number of data modules to be backed up and/or loaded, and their respective sizes.

Example: Approx. 1.3 ms elapse when the following are transferred:

1 data module with a size of 512 bytes, OR
5 data modules with a size of 64 bytes each.

7.20.5 Status Messages & Error Messages

The status messages and error messages listed below are always returned in Register C, and can be evaluated immediately after issuing the command representing the function call. In the event that an error is detected during command processing, processing will be interrupted, and the load and/or backup procedure rejected. An occurrence of this type has no additional effect on the processing of the PLC program. Only the respective status and/or error message is returned in Register C.

Code	Cause of error
00_H	No error detected.
01_H	No MemoryCard inserted.
02_H	MemoryCard is write-protected! Write protection switch?
03_H	MemoryCard is not a RAM memory card.
04_H	MemoryCard does not contain a valid PLC program. Reasons: • Unformatted or empty MemoryCard • Is MemoryCard formatted for another PLC, e.g., CL200? • Is MemoryCard a firmware-type MemoryCard? • Invalid checksum of PLC program stored on MemoryCard.
10_H	One or more data modules are not available. The number of one or more nonexistent data modules was defined in the parameter list. This means that a data module that is to be transferred is missing in either the program on the MemoryCard or in the program onboard the PLC.
11_H	Unequal size of one or more data modules. The parameter list defines data modules whose size differs between the PLC program and the program on the MemoryCard.

8 Instruction List

8.1 Structure of Controller Instructions

Control Instruction			
Operation part	Operand attribute	Source operand	Destination operand
OPP	OPA	SRC ,	DEST

Examples:

A	B	I0.0		
A	W	-name	,	O
L	BY	O0	,	B
T	W	C	,	M10
MUL	W	K1234D	,	D

8.2 Status Bits (Flags) & Associated Special Markers

The status bits are influenced by the following instruction groups, and can be equally applied to program processing instructions, as well as to logical links (flag queries):

- Compare
- Convert
- Swap
- Increment
- Decrement
- Shift
- Rotate
- Add
- Subtract
- Multiply
- Divide

Status bits			Special marker	Explanation
Flags	PG display readout	JP... CM... EM...	Query	
C=1 C=0	C	...C ...CN	A SM31.3 AN SM31.3	Carry Carry Not
O=1 O=0	O	...O ...ON	A SM31.5 AN SM31.5	Overflow Overflow Not
Z=1 Z=0	Z	...Z ...N	A SM31.7 AN SM31.7	Zero Not Zero
N=1 N=0	N	...M ...P	A SM31.6 AN SM31,6	Negative / Minus Positive
AG=1 AG=0	N+Z	...AG ...MZ	A SM31.0 AN SM31.0	Arithmetical greater Minus / Zero
LG=1 LG=0	C+Z	...LG ...CZ	A SM31.4 AN SM31.4	Logical greater CarryZero

8.3 Key to Abbreviations

OPP		Operation / operator
OPA		Operand attribute
	B	Bit
	BY	Byte
	W	Word
	R	REAL (Floating Point 32 Bit)
	L	LREAL (Floating Point 64 Bit)
SRC		Source operand
DEST		Destination operand
	I	Input
	O	Output
	M	Marker
	SM	Special marker
	T	Timer
	C	Counter
	S	System area
	K	Constant
	&opd-ind	Indirect operand byte address (constant)
	$opd-abs	Absolute operand word address
	SYM	Symbolic (max. 8 characters)
	R.bit	Register bit with R = A, B, C, D, and bits = 0 - 15
	[R]	Register indirect (A, B, C or D)
	TI	Timed interrupt (time-controlled processing)
	PI	Peripheral interrupt
	SI	System interrupt
RG		Program rung
	A	Permitted operation at RG start
	E	Operation concluding RG
AddrMode		Addressing mode
	D	Direct
	R	Register (A, B, C or D)
	[R]	Register indirect (A, B, C or D)

Status bits / Special markers

V		link result RES
C	SM31.3	Carry
O	SM31.5	Overflow
Z	SM31.7	Zero
M	SM31.6	Negative
AG	SM31.0	Arithmetical greater
LG	SM31.4	Logical greater

Times (µs)

35x ,40x	Command processing times of
501/500	ZS40x, ZS501 and/or ZS500

8.4 Binary Links

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x / 501 / 500				
A	B	I/O/M/SM		•		•			•							**0.25** / 0.4	A	B	E0.0	AND link,
		T/C/SYM		•		•			•							**0.25** / 0.4	A	B	T0	query for status '1'
		R.bit		•			•		•							**0.25** / 0.4	A	B	A.0	
		[R]		•				•	•							**0.33** / 0.5	A	B	[A]	
		P		•		•			•							**0.75** / 1.2	A	B	P0	
AN	B	I/O/M/SM		•		•			•							**0.25** / 0.4	AN	B	O0.0	AND link,
		T/C/SYM		•		•			•							**0.25** / 0.4	AN	B	Z0	query for status '0'
		R.bit		•			•		•							**0.25** / 0.4	AN	B	B0,0	
		[R]		•				•	•							**0.33** / 0.5	AN	B	[B]	
		P		•		•			•							**0.75** / 1.2	AN	B	P1	
O	B	I/O/M/SM		•		•			•							**0.25** / 0.4	O	B	M0.0	OR link,
		T/C/SYM		•		•			•							**0.25** / 0.4	O	B	-SYMBOL	query for status '1'
		R.bit		•			•		•							**0.25** / 0.4	O	B	C0,0	
		[R]		•				•	•							**0.33** / 0.5	O	B	[C]	
		P		•		•			•							**0.75** / 1.2	O	B	P10	
ON	B	I/O/M/SM		•		•			•							**0.25** / 0.4	ON	B	SM31.7	OR link,
		T/C/SYM		•		•			•							**0.25** / 0.4	ON	B	-name	query for status '0'
		R.bit		•			•		•							**0.25** / 0.4	ON	B	D.0	
		[R]		•				•	•							**0.33** / 0.5	ON	B	[D]	
		P		•		•			•							**0.75** / 1.2	ON	B	P62	
=	B	A/M/SYM			•	•										**0.33** / 0.5	=	B	A0.0	Assign result when
																0.42 / 0.7[1]				RES = 1
		P			•	•										**0.83** / 1.3	=	B	P0	
																0.92 / 1.5[1]				[1] upon status change
		[R]			•			•								**0.42** / 0.6	=	B	[A]	
																0.50 / 0.8[1]				
=OM*	B	R.bit			•		•									**0.25** / 0.4	=OM	B	A.0	
S	B	A/M/SYM			•	•										**0.25** / 0.4[1]	S	B	M0.0	Set bit HIGH when RES = 1
																0.42 / 0.7[2]				
		P			•	•										**0.75** / 1.2[1]	S	B	P1	[1] when RES = 0
																0.92 / 1.5[2]				[2] upon status change
		[R]			•			•								**0.33** / 0.5[1]	S	B	[B]	
																0.50 / 0.8[2]				
SWM*	B	R.bit			•		•									**0.25** / 0.4[1]	SWM	B	B0,0	
R	B	A/M/SYM			•	•										**0.25** / 0.4[1]	R	B	-SYMBOL	Set bit LOW when RES = 1
																0.42 / 0.7[2]				
		P			•	•										**0.75** / 1.2[1]	R	B	P62	[1] when RES = 0
																0.92 / 1.5[2]				[2] when RES = 1
		[R]			•			•								**0.33** / 0.5[1]	R	B	[C]	
																0.50 / 0.8[2]				
RWM*	B	R.bit			•		•									**0.25** / 0.4[1]	RWM	B	C0,0	
TST	B	R.bit					•			•						**85** / 150	TST	B	A.0	Check register bit for status = 1 when met: C = 1
TSTZ	B	R.bit					•			•						**85** / 150	TSTZ	B	A.15	Check register bit for status = 0 when met: C = 1

* No status display in the Monitor program or the PG programming unit.

8.5 Timer Programming

The ZS35x, ZS40x, and the ZS50x provide 128 timer circuits, T0 through T127.

These can be utilized in the following modes:

SP
SPE
SR
SF
SRE

Starting a timer requires a positive transition of the timer start condition.

A timer start condition that is satisfied immediately upon Power-On does not qualify as a transition!

Presetting a default value of "**0**" for the timer start condition is possible already in the startup OM, provided the information regarding remanence characteristics are observed. For more information about this subject, refer also to Section 6.12, "Remanence Characteristics".

Without exception, any preselection of the timer start condition must utilize the correct timer start mode!

Example:
Right:

```
A    B  -log0     ;Timer start condition 0
SR      A,T5

A    B  -log1     ;Timer start condition 1
SR      A,T5      ;Timer start is correct
```

Wrong:

```
A    B  -log0     ;Timer start condition 0
SR      A,T5

A    B  -log1     ;Timer start condition 1
SPE     A,T5      ;Different timer modes = incorrect timer start!!
```

The timers are decremented in the I/O state. This means that a timeout will be recognized only during the I/O state, and not during the program cycle!

Because during each I/O state, a timer is decremented by multiples of the defined time matrix, it is good practice to select a resolution that is as fine as possible.

A timer start occurs immediately upon a positive transition of the timer start condition.

8.5.1 Timer Instructions

Timer starts are activated only when the RES signal undergoes a transition from 0↑1. In advance of the timer start, the time value is loaded into the register being used. Reset and stop functions of timers are always RES signal-dependent. The timer status for logical links is instruction-dependent, and may be taken from the timer diagrams.

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example		Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500			
SP		R	, T		•	•										**0.58** / 0.9[1]	SP	A,T0	Start time as pulse
			, SYM			•										**0.83** / 1.4[2]	SP	A,-Symbol	
			, [R]					•								**0.42** / 0.6[1]	SP	A,[B]	
																0.66 / 1.1[2]			[1] when RES = stable
			, P			•										**1.10** / 1.7[1]	SP	A,P0	[2] when RES = ↑ ↓
																1.33 / 2.2[2]			
SPE		R	, T		•	•										**0.58** / 0.9[1]	SPE	A,T0	Start pulse extended
			, SYM			•										**0.83** / 1.4[2]	SPE	A,-Symbol	
			, [R]					•								**0.42** / 0.6[1]	SPE	A,[B]	
																0.66 / 1.1[2]			[1] when RES = stable
			, P			•										**1.10** / 1.7[1]	SPE	A,P0	[2] when RES = ↑ ↓
																1.33 / 2.2[2]			
SR		R	, T		•	•										**0.58** / 0.9[1]	SR	A,T0	Start time as raising delay
			, SYM			•										**0.83** / 1.4[2]	SR	A,-Symbol	
			, [R]					•								**0.42** / 0.6[1]	SR	A,[B]	
																0.66 / 1.1[2]			[1] when RES = stable
			, P			•										**1.10** / 1.7[1]	SR	A,P0	[2] when RES = ↑ ↓
																1.33 / 2.2[2]			
SF		R	, T		•	•										**0.58** / 0.9[1]	SF	A,T0	Start time as falling delay
			, SYM			•										**0.83** / 1.4[2]	SF	A,-Symbol	
			, [R]					•								**0.42** / 0.6[1]	SF	A,[B]	
																0.66 / 1.1[2]			[1] when RES = stable
			, P			•										**1.10** / 1.7[1]	SF	A,P0	[2] when RES = ↑ ↓
																1.33 / 2.2[2]			
SRE		R	, T		•	•										**0.58** / 0.9[1]	SRE	A,T0	Start time as raising delay
			, SYM			•										**0.83** / 1.4[2]	SRE	A,-Symbol	extended
			, [R]					•								**0.42** / 0.6[1]	SRE	A,[B]	
																0.66 / 1.1[2]			[1] when RES = stable
			, P			•										**1.10** / 1.7[1]	SRE	A,P0	[2] when RES = ↑ ↓
																1.33 / 2.2[2]			
RT		T			•	•										**0.58** / 0.9[1]	RT	T0	Reset time when RES = 1
		SYM				•										**0.83** / 1.4[2]	RT	-Symbol	
		[R]						•								**0.42** / 0.6[1]	RT	[B]	[1] when RES = 0
																0.66 / 1.1[2]			[2] when RES = 1
		P				•										**1.10** / 1.7[1]	RT	P0	
																1.33 / 2.2[2]			
TH		T			•	•										**0.58** / 0.9[1]	TH	T0	Timer halt when RES = 1,
		SYM				•										**0.83** / 1.4[2]	TH	-Symbol	Time continues when RES = 1
		[R]						•								**0.42** / 0.6[1]	TH	[B]	
																0.66 / 1.1[2]			[1] when RES = 0
		P				•										**1.10** / 1.7[1]	TH	P0	[2] when RES = 1
																1.33 / 2.2[2]			

8.5.2 Time Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x	x	x	x	R	R	W	W	W	W	W	W	W	W	W	W
				Time matrix		Time value 1 - 1023									

L	-Bef.:	x= 0	0	0	0: 10 ms
LIMR	-Bef.:	x= Flags	0	1	1: 100 ms
			1	0	2: 1 s
			1	1	3: 10 s

:

Program entry of time constant:
Kw.r with time value w = 1 -1023
and time matrix r = 0 - 3

Bits 0 through 9 contain the **timer value**. The permitted value range is between 1 and 1023.

The timer value is multiplied by the **time matrix** value declared in **Bits 10 and 11**.

Example:

Timer T100 shall be started at 15 sec:

```
L    W  K080FH,A  ;15s declaration in 1-sec time matrix of CL500
A    B  -start
SPE     A,T100
```

Same function with higher matrix resolution, i.e., higher accuracy:

```
L    W  K0496H,A  ;15s declaration in 1oo-ms time matrix of CL500
A    B  -start
SPE      A,T100
```

Timed start with the assistance of the PG time matrix:

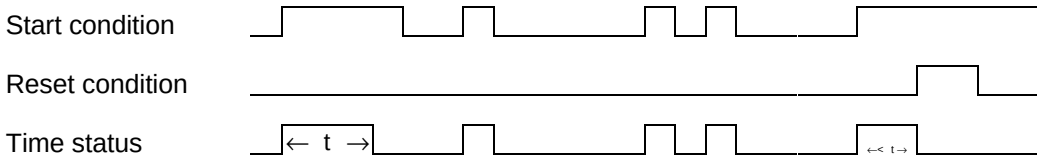
```
L    W  K15.2,A   ;15s declaration in 1-ms time matrix of PG
A    B  -start
SPE     A,T100
```

Same function with higher matrix resolution, i.e., higher accuracy:

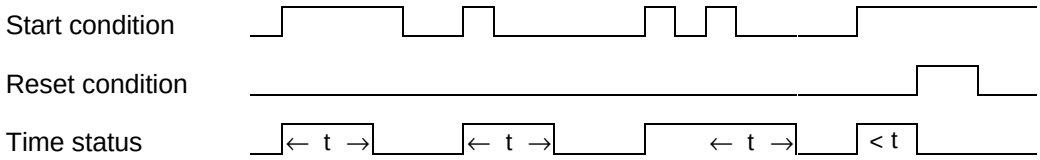
```
L    W  K150.1,A  ;15s declaration in 1oo-ms time matrix of PG
A    B  -start
SPE     A,T100
```

8.5.3 Timer Diagrams

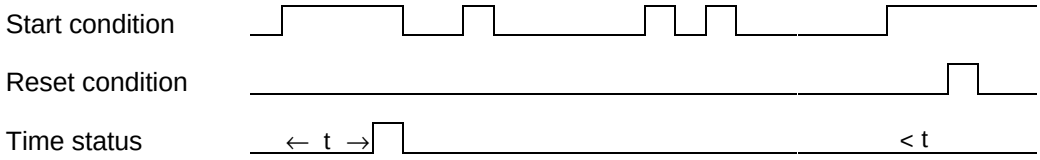
SP – Start time as pulse



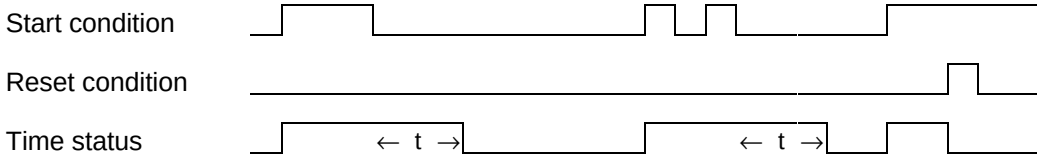
SPE – Start pulse extended



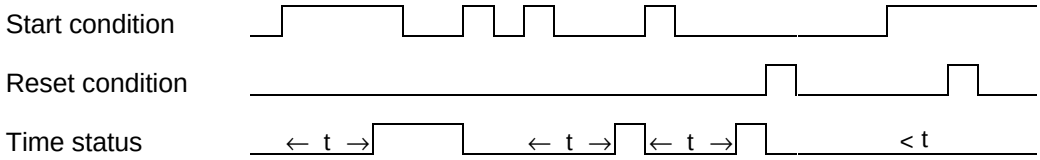
SR – Start time as raising delay



SF – Start time as falling delay



SRE – Start time as raising delay extended



8.5.4 ms Timer for CL350 / CL400

The ms timer in the CL350 and CL400 runs concurrent with application program processing, and with a basic accuracy of 0.1 ms. To keep the function as flexible as possible, internal transition control of the timer was not used.

Control bits can be used to define whether or not a time matrix is to be used, and at which time resolution the actual value is to be transferred into the associated special marker. From that location, it can be queried via the high-speed gate array instructions. To load the accurate current value, the CL350 / CL400 provide a µP command which facilitates processing at a 0.1 ms resolution.

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	(µs)		
SMST	W	R					•									approx. 60	SMST W O	Start & control instruction for ms timer. See below for explanations of functions and control bits.
LMST	W	R					•									approx. 60	LMST W B	Timer read instruction. µP for reading actual value with 0.1 ms accuracy!

Explanation of timer handling register bits

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Timer functional principle				SM update		Timer specified value									
				NfB2	NfB1	10 bits in ms-matrix ⇒ MAX timer value = 1023 ms									
				0	0	= none matrix for automatic actual-value updating									
				0	1	= 1 ms in the special marker bytes SM14/15									
				1	0	= 5 ms									
				1	1	= 10 ms									
			Int/Flag	= 0		Timeout sets special marker bit SM15.7									
				= 1		Timeout calls up OM42 interrupt module									
		Wrap/Run	= 0	Time stands still upon timeout. Depending on Bit 12 = Int/Flag, the OM42 interrupt module is called, or special marker SM15.7 goes HIGH. The remainder of SM14/15 is cleared.											
			= 1	Upon timeout the timer restarts automatically, and the time counter SM15.2 - 15.6 is incremented. The counter stops at a maximum value of 31, until it is again cleared by a Reset. The timer continues to "wrap around"..											
	Reset		= 0	Start / Stop cause timer to continue / causes timer to stop.											
			= 1	The actual value and the special marker bytes SM14/15 go LOW.											
Start/Stop			= 0	Timer stops											
			= 1	Timer restarts and/or continues											

Erklärung der Sondermerkerbits des SM14/15:

SM15.								SM14.							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
expir'd	Timer counter					Actual timer value, updated as per matrix definition									

8.6 Counter Instructions

Timer resets and changes of direction of count (up/down) are activated only when the RES signal undergoes a transition from $_0\uparrow^1$.

In advance of the reset, the required counter content is loaded into the register being used.

Counter reset functions are statically RES signal-dependent.

The counter status for logical links depends on the counter content. For counter contents > 0, the status is = 1 (HIGH); counter content = 0 will have status = 0 (LOW).

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example		Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500			
SC		R	, Z		•	•										**0.58** / 0.9[1]	SC	A,Z0	Set counter
			, SYM			•										**1.00** / 1.7[2]	SC	A,-Symbol	
			, [R]					•								**0.42** / 0.6[1]	SC	A,[B]	[1] when RES = stable
																0.83 / 1.4[2]			[2] when RES = $_0\uparrow^1$
			, P			•										**1.10** / 1.7[1]	SC	A,P0	
																1.50 / 2.5[2]			
CU		Z			•	•										**0.58** / 0.9[1]	CU	Z0	Count up
		SYM				•										**1.00** / 1.7[2]	CU	-Symbol	
		[R]						•								**0.42** / 0.6[1]	CU	[B]	[1] when RES = stable
																0.83 / 1.4[2]			[2] when RES = $_0\uparrow^1$
		P				•										**1.10** / 1.7[1]	CU	P0	
																1.50 / 2.5[2]			
CD		Z			•	•										**0.58** / 0.9[1]	CD	Z0	Count down
		SYM				•										**1.00** / 1.7[2]	CD	-Symbol	
		[R]						•								**0.42** / 0.6[1]	CD	[B]	[1] when RES = stable
																0.83 / 1.4[2]			[2] when RES = $_0\uparrow^1$
		P				•										**1.10** / 1.7[1]	CD	P0	
																1.50 / 2.5[2]			
RC		Z			•	•										**0.58** / 0.9[1]	RC	Z0	when RES = 1, reset counter
		SYM				•										**1.00** / 1.7[2]	RC	-Symbol	
		[R]						•											[1] when RES = 0
		P				•										**0.42** / 0.6[1]	RC	[B]	[2] when RES = 1
																0.83 / 1.4[2]			
																1.10 / 1.7[1]	RC	P0	
																1.50 / 2.5[2]			

8.7 Digital Links

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500				
A	W	I/O/M/SM	, R			•				0	0	•	•			**0.5** / 0.8	A	BY	I0,A	Digital AND link between
	BY	T/C/K/SYM				•				0	0	•	•			**0.5** / 0.8	A	W	T0,B	source and destination.
		S/DP/DF				•				0	0	•	•			**0.5** / 0.8	A	BY	S0,C	The result is written to destination.
		D/DX				•				0	0	•	•			**0.75** / 1.2	A	W	D0,D	
		II/EI				•				0	0	•	•			**3.0**[1] **5.8**[2] / 9[1] 11[2]	A	BY	II0,A	[1] when OPA = BY
																				[2] when OPA = W
		R					•			0	0					**0.25** / 0.4	A	W	A,B	
		[R]						•		0	0	•	•			**0.33** / 0.5	A	BY	[B],C	
		P				•				0	0	•	•			**1.00** / 1.6	A	W	P0,D	
AN	W	I/O/M/SM	, R			•				0	0	•	•			**0.5** / 0.8	AN	BY	I127,A	Digital AND NOT link between
	BY	T/C/K/SYM				•				0	0	•	•			**0.5** / 0.8	AN	W	T127,B	source and destination.
		S/DP/DF				•				0	0	•	•			**0.5** / 0.8	AN	BY	S511,C	The result is written to destination.
		D/DX				•				0	0	•	•			**0.75** / 1.2	AN	W	D510,D	
		II/EI				•				0	0	•	•			**3.0**[1] **5.8**[2] / 9[1] 11[2]	AN	BY	IE127,A	[1] when OPA = BY
																				[2] when OPA = W
		R					•			0	0					**0.25** / 0.4	AN	W	A,B	
		[R]						•		0	0	•	•			**0.33** / 0.5	AN	BY	[B],C	
		P				•				0	0	•	•			**1.00** / 1.6	AN	W	P62,D	
O	W	I/O/M/SM	, R			•				0	0	•	•			**0.5** / 0.8	O	BY	I0,A	Digital OR link between
	BY	T/C/K/SYM				•				0	0	•	•			**0.5** / 0.8	O	W	T0,B	source and destination.
		S/DP/DF				•				0	0	•	•			**0.5** / 0.8	O	BY	S0,C	The result is written to destination.
		D/DX				•				0	0	•	•			**0.75** / 1.2	O	W	D0,D	
		II/EI				•				0	0	•	•			**3.0**[1] **5.8**[2] / 9[1] 11[2]	O	BY	II0,A	[1] when OPA = BY
																				[2] when OPA = W
		R					•			0	0					**0.25** / 0.4	O	W	A,B	
		[R]						•		0	0	•	•			**0.33** / 0.5	O	BY	[B],C	
		P				•				0	0	•	•			**1.00** / 1.6	O	W	P0,D	
ON	W	I/O/M/SM	, R			•				0	0	•	•			**0.5** / 0.8	ON	BY	I127,A	Digital OR -NOT link between
	BY	T/C/K/SYM				•				0	0	•	•			**0.5** / 0.8	ON	W	T127,B	source and destination.
		S/DP/DF				•				0	0	•	•			**0.5** / 0.8	ON	BY	S511,C	The result is written to destination.
		D/DX				•				0	0	•	•			**0.75** / 1.2	ON	W	D510,D	
		II/EI				•				0	0	•	•			**3.0**[1] **5.8**[2] / 9[1] 11[2]	ON	BY	IE127,A	[1] when OPA = BY
																				[2] when OPA = W
		R					•			0	0					**0.25** / 0.4	ON	W	A,B	
		[R]						•		0	0	•	•			**0.33** / 0.5	ON	BY	[B],C	
		P				•				0	0	•	•			**1.00** / 1.6	ON	W	P62,D	
XO	W	I/O/M/SM	, R			•				0	0	•	•			**0.5** / 0.8	XO	BY	I0,A	EXCLUSIVE OR link between
	BY	T/C/K/SYM				•				0	0	•	•			**0.5** / 0.8	XO	W	T0,B	source and destination.
		S/DP/DF				•				0	0	•	•			**0.5** / 0.8	XO	BY	S0,C	The result is written to destination.
		D/DX				•				0	0	•	•			**0.75** / 1.2	XO	W	D0,D	
		II/EI				•				0	0	•	•			**3.0**[1] **5.8**[2] / 9[1] 11[2]	XO	BY	II0,A	[1] when OPA = BY
																				[2] when OPA = W
		R					•			0	0					**0.25** / 0.4	XO	W	A,B	
		[R]						•		0	0	•	•			**0.33** / 0.5	XO	BY	[B],C	
		P				•				0	0	•	•			**1.00** / 1.6	XO	W	P0,D	
XON	W	I/O/M/SM	, R			•				0	0	•	•			**0.5** / 0.8	XON	BY	I127,A	EXCLUSIVE OR NOT link between
	BY	T/C/K/SYM				•				0	0	•	•			**0.5** / 0.8	XON	W	T127,B	source and destination.
		S/DP/DF				•				0	0	•	•			**0.5** / 0.8	XON	BY	S511,C	The result is written to destination.
		D/DX				•				0	0	•	•			**0.75** / 1.2	XON	W	D510,D	
		II/EI				•				0	0	•	•			**3.0**[1] **5.8**[2] / 9[1] 11[2]	XON	BY	IE127,A	[1] when OPA = BY
																				[2] when OPA = W
		R					•			0	0					**0.25** / 0.4	XON	W	A,B	
		[R]						•		0	0	•	•			**0.33** / 0.5	XON	BY	[B],C	
		P				•				0	0	•	•			**1.00** / 1.6	XON	W	P62,D	

8.8 Compare Instructions

The CPLA (Compare Logical and Arithmetical) instruction is available for Compare operations.

As the name implies, the instruction facilitates both logical and arithmetical compare operations.

The logical compare operation regards the bytes and/or words to be compared as unsigned integers, i.e., as "unsigned 8" or "unsigned 16".

The arithmetical compare operation regards the bytes and/or words to be compared as signed integers, i.e., as "integer 8" or "integer 16".

Subsequent to a Compare operation, the flags provide information about the Compare result.

As required for CL300 compatibility, the CPL instruction is also available.

This commands facilitates only logical Compare operations.

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500				
CPL	W	I/O/M/SM	, R			•						•	•	•	•	**0.5** / 0.8	CPL	W	I0,A	Logical Compare operation.
	BY	T/C/K/SYM				•						•	•	•	•	**0.5** / 0.8	CPL	BY	K0,B	The result may be used for logical
		S/DP/DF				•						•	•	•	•	**0.5** / 0.8	CPL	W	DP0,C	purposes only, i.e., the values will be
		D/DX				•						•	•	•	•	**0.75** / 1.2	CPL	BY	D0,D	treated as positive integers.
		II/EI				•						•	•	•	•	**3.0**[1] **5.8**[2] / 9[1] 11[2]	CPL	W	II0,A	
		R					•									**0.25** / 0.4	CPL	BY	B,C	[1] when OPA = BY
		[R]						•				•	•	•	•	**0.33** / 0.5	CPL	W	[C],D	[2] when OPA = W
		P				•						•	•	•	•	**1.00** / 1.6	CPL	BY	P0,A	
CPLA	W	I/O/M/SM	, R			•				•	•	•	•	•	•	**0.5** / 0.8	CPLA	W	I62,A	Arithmetical Compare function. The
	BY	T/C/K/SYM				•				•	•	•	•	•	•	**0.5** / 0.8	CPLA	BY	K255,B	result may be used for logical and
		S/DP/DF				•				•	•	•	•	•	•	**0.5** / 0.8	CPLA	W	DP510,C	arithmetical purposes.
		D/DX				•				•	•	•	•	•	•	**0.75** / 1.2	CPLA	BY	D511,D	Compare values
		II/EI				•				•	•	•	•	•	•	**3.0**[1] **5.8**[2] / 9[1] 11[2]	CPLA	W	IE62,A	logical: positive, integer
																				arithmet.: two's complement, integer
		R					•									**0.25** / 0.4	CPLA	BY	B,C	
		[R]						•		•	•	•	•	•	•	**0.33** / 0.5	CPLA	W	[C],D	[1] when OPA = BY
		P				•				•	•	•	•	•	•	**1.00** / 1.6	CPLA	BY	P62,A	[2] when OPA = W

Binary result evaluation of compare results occurs by means of jump functions dependent upon status bits, and/or through status bit query.

Examples:

Compare		**CPL**			**CPLA**					
CPL	B,A	Logical			Logical			Arithmetical		
CPLA	B,A	Flag	SM...		Flag	SM...		Flag	SM...	
Equal	A=B	JPZ	A	SM31.7	JPZ	A	SM31.7	JPZ	A	SM31.7
Unequal	A≠B	JPN	AN	SM31.7	JPN	AN	SM31.7	JPN	AN	SM31.7
Less than	A<B	JPM	A	SM31.6	JPCY	A	SM31.3	JPM	A	SM31.6
Less than or equal	A≤B		A	SM31.6	JPCZ	AN	SM31.4	JPMZ	AN	SM31.0
			O	SM31.7						
Greater	A>B		AN	SM31.6	JPLG	A	SM31.4	JPAG	A	SM31.0
			AN	SM31.7						
Greater than or equal	A≥B	JPP	AN	SM31.6	JPCN	AN	SM31.3	JPP	AN	SM31.6

8.9 Load & Transfer Operations

8.9.1 Load Instructions

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500				
L	W BY	I/O/M/SM T/C/K/SYM DP/DF D/DX II/EI R [R] P &opd-ind	, R			• • • • • • • •	 • 	 • 								**0.5** / 0.8 **0.5** / 0.8 **0.5** / 0.8 **0.75** / 1.2 **3.0**[1] **5.8**[2] / 9[1] 11[2] **0.25** / 0.4 **0.33** / 0.5 **1.00** / 1.6 **0.5** / 0.8	L W I0,A L BY K0,B L W DP0,C L BY D0,D L W II0,A L BY B,C L W [C],D L BY P0,A L W &E10,A			Transfer contents of SRC operand Load DEST operand. Read value [1] when OPA = BY [2] when OPA = W
LIMR	W	IE,IA,EZ,AZ [R] $opd-abs	, R			• •	 	 • 								**0.58** / 0.9	LIMR W II0,A LIMR W [A],B LIMR W $opd-abs,B			Load SRC operand contents. Load contents of absolute peripheral **word** address (Load Register A and/or opd-abs) to B.
LMB	W	[R] $opd-abs	, R			 •	 	• 								**0.58** / 0.8	LMB W [A],B LMB W $opd-abs,B			Load contents of program memory **word** address (Register A and/or. opd-abs) to B. LMB addresses the lower 64 k of program memory.
LMBX	W	[R] $opd-abs	, R			 •	 	• 								**0.58** / 0.9	LMBX W [A],B LMBX W $opd-abs,B			Load contents of program memory **word** address (Register A and/or. opd-abs) to B. LMBX addresses the upper 64 k of the 128 k perogram memory.

8.9.2 Transfer Instructions

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500				
T	W BY	R	, O/M/SYM , S/DP/DF , D/DX , IO/EO , R , [R] , P			• • • • •	 • 	 • 								**0.5** / 0.8 **0.5** / 0.8 **0.75** / 1.2 **3.0**[1] **5.8**[2] / 9[1] 11[2] **0.25** / 0.4 **0.33** / 0.5 **1.00** / 1.6	T W A,M0 T BY B,DF0 T W C,D0 T BY D,IA0 T W A,B T BY B,[C] TT W D,P0			Transfer contents of SRC operand into DEST operand. Write value
TIMR	W	R	, IE,IA,EZ,AZ , [R] , $opd-abs			• •	 	 • 								**0.58** / 0.9	TIMR W A,IA0 TIMR W B,[A] TIMR W B,$opd-abs			Transfer contents into DEST operand. Transfer contents of B to absolute peripheral **word** address (Register A and/or opd-abs).
TMB	W	R	, [R] , $opd-abs			 •	 	• 								**0.58** / 0.9	TMB W A,[B] TMB W A,$opd-abs			Transfer Register A to the program memory **word** address (Reg. A and/or. opd-abs). TMB addresses the lower 64 k of program memory.
TMBX	W	R	, [R] , $opd-abs			 •	 	• 								**0.58** / 0.9	TMBX W A,[B] TMBX W A,$opd-abs			Transfer Register A to the program memory **word** address (Reg. A and/or. opd-abs). TMBX addresses the upper 64 k op program memory.
CAUTION: The **TMB and TMBX** are used only to **manipulate program memory** in standard BOSCH modules. **Under no circumstances may these instructions be used for write-access to the reference list** because this would result in severe errors that could be remedied only by reloading the entire PLC program!																				

8.10 Convert, Swap & Backup Operations

8.10.1 Convert Instructions

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500		
BID	W BY	R					•			0	•	0	•			**62 /** 115 **70 /** 125	BID W O BID BY B	Binary → BCD (decimal) Result > 9999 sets the overflow bit.
DEB	W BY	R					•			0	•	0	•			**62 /** 115 **70 /** 125	DEB W C DEB BY D	BCD (decimal) → binary Incorrect BCD encoding sets the overflow bit.
TC	W BY	R					•			•	•	•	•			**0.25 /** 0.4	TC W O TC BY B	Converts the register contents to the two's complement.
N	W BY	R					•			0	•	0	•			**0.25 /** 0.4	N W C N BY D	Negates the register contents (one's complement).

Notation of positive and negative numbers

A negative number corresponds to the two's complement of the positive number.

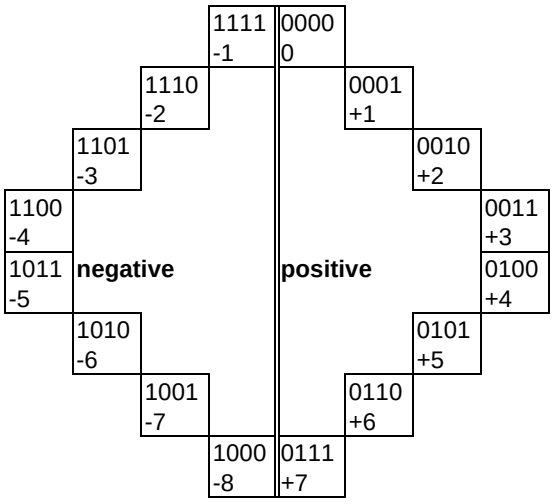
Example: 0 1 1 0 positive integer 6
1 0 0 1 Negation and/or one's complement
\+ 1
1 0 1 0 two's complement = negative integer 6

In the case of word operations, the differentiation between positive and negative integers is determined by Bit 15. In the case of byte operations, this is Bit 7.

Word: Bit 15 = 0 Byte: Bit 7 = 0 positive integer
Bit 15 = 1 Bit 7 = 1 negative integer

Integer value range

Positive integers Word: 0 - 32767 Byte: 0 - 127
Negative integers 0 - 32768 0 - 128



8.10.2 Swap Instructions

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500				
SWAP	W BY	R					•									**0.25** / 0.4	SWAP	W	O	Byte swap in register High-Byte ↔ Low-Byte
EXC	W	O/M/SYM S/DP/DF D P [R]	, R			• • •		•		0	•	0	•			**0.67** / 1.1 **0.67** / 1.1 **1.00** / 1.6 **1.17** / 1.9[1] **1.5** / 2.4[2] **0.5** / 0.8[1]	EXC EXC EXC EXC EXC	W W W W W	A0,A DP0,B D0,C P0,D [A],B	Swaps the contents of SRC and DEST. For register-to-register addressing, only A and B are permitted! [1] directly adressable OPD
																0.83 / 1.3[2] **1.33** / 2.1[3]				[2] for data module [3] for parameters
	W BY	R					•									**0.25** / 0.4	EXC EXC	W BY	A,B A,B	

8.10.3 Stack Instructions

⇒ **The available stack size comprises 128 words. In the event of underflow, bit SM28.4 in the system area is set; overflow sets system area bit SM28.5. The I/O state resets/deletes the entire application stack.**

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501/500				
PUSH	W	R					•									**0.25** / 0.4	PUSH	W	O	Saves the register contents to application stack, and lowers the stack address.
POP	W	R					•									**0.25** / 0.4	POP	W	B	Raises the application stack address and reads the saved contents from the stack.

8.11 Increment, Decrement, Shift & Rotate Operations

8.11.1 Increment / Decrement Instructions

Increment / decrement the source operand (SRC_OPD, 1 to 15 in number):

- by the number n
- when n=0, and when [C], by the number stored in C

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x,40x 501 / 500		
INC	W BY	R	, n , 0 , [C]			•	•	•		•	•	•	•			**0.25** / 0.4	INC BY A,5 INC W A,0 INC W B,[C]	Raise (increment) the contents of the SRC_OPD.
DEC	W BY	R	, n , 0 , [C]			•	•	•		•	•	•	•			**0.25** / 0.4	DEC BY A,5 DEC W A,0 DEC W B,[C]	Lower (decrement) the contents of the SRC_OPD.

8.11.2 Shift Instructions

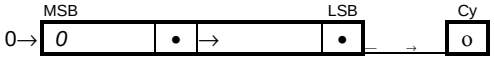
Shift the contents of the source operand:

- by the number n
- when n=0, and when [C], by the number stored in C
- when OPA = W n: 1-15
- when OPA = BY n: 1-7

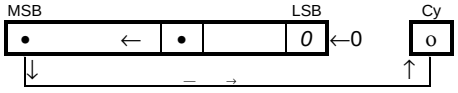
Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x,40x 501 / 500		
SLR	W BY	R	, n , 0 , [C]			•	•	•		•	0 0 0	0 0 0	•			**0.25** / 0.4	SLR W A,7 SLR BY B,[C]	SHIFT logical RIGHT
SLL	W BY	R	, n , 0 , [C]			•	•	•		•	0 0 0	0 0 0	•			**0.25** / 0.4	SLL W A,7 SLL BY B,[C]	SHIFT logical LEFT
SAR	W BY	R	, n , 0 , [C]			•	•	•		•	0 0 0	0 0 0	•			**0.25** / 0.4	SLL W A,7 SLL BY B,[C]	SHIFT arithmetical RIGHT

Logical SHIFT

SLR B, n

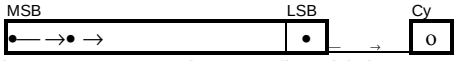


SLL B, n



Arithmetical SHIFT (All bits being vacated are filled up with the contents of the MSB)

SAR B, n



In the case of shift operations exceeding one space, the overflow bit is set after a "1" was shifted through Cy.

8.11.3 Rotate Instructions

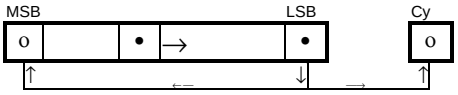
Shift the contents of the source operand:

- by the number n
- when n=0, and when [C], by the number stored in C
  when OPA = W n: 1-15
- when OPA = BY n: 1-7

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x **0.25** / 501 / 500		
ROR	W BY	R	, n , 0 , [C]			•	•	•		• • •	0 0 0	0 0 0	• • •			**0.25** / 0.4	ROR BY A,7 ROR W A,0 ROR W B,[C]	Rotate RIGHT
ROL	W BY	R	, n , 0 , [C]			•	•	•		• • •	0 0 0	0 0 0	• • •			**0.25** / 0.4	ROR BY A,7 ROR W A,0 ROR W B,[C]	Rotate LEFT
RCR	W BY	R	, n , 0 [C]			•	•	•		• • •	0 0 0	0 0 0	• • •			**0.25** / 0.4	ROR BY A,7 ROR W A,0 ROR W B,[C]	Rotate RIGHT through CARRY
RCL	W BY	R	, n , 0 , [C]			•	•	•		• • •	0 0 0	0 0 0	• • •			**0.25** / 0.4	ROR BY A,7 ROR W A,0 ROR W B,[C]	Rotate LEFT through CARRY

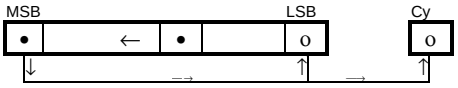
Rotate RIGHT

ROR B, n



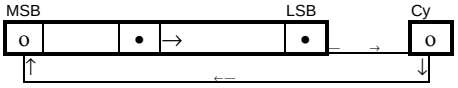
Rotate LEFT

ROL B, n



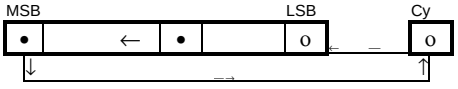
Rotate RIGHT through CARRY

RCR B, n



Rotate LEFT through CARRY

RCL B, n



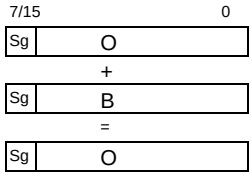
8.12 Arithmetic

8.12.1 Add Instructions

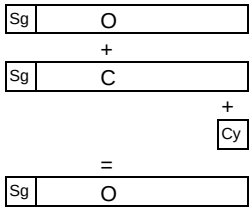
Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501/500				
ADD	W BY	I/O/M/SM T/C/K/SYM S/DP/DF D/DX II/EI R [R] P	, R			• • • • • •	 • 	 • 		• • • • • • • •	• • • • • • • •	• • • • • • • •	• • • • • • • •			**0.5** / 0.8 **0.5** / 0.8 **0.5** / 0.8 **0.75** / 1.2 **3.0**[1] **5.8**[2] / 9[1] 11[2] **0.25** / 0.4 **0.33** / 0.5 **1.00** / 1.6	ADD W I0,A ADD BY K0,B ADD W DP0,C ADD BY D0,D ADD W II0,A ADD BY B,C ADD W [C],D ADD BY P0,A		Fixed-point addition of signed integers Source + destination = DEST [1] when OPA = BY [2] when OPA = W
ADC	W BY	I/O/M/SM T/C/K/SYM S/DP/DF D/DX II/EI R [R] P	, R			• • • • • •	 • 	 • 		• • • • • • • •	• • • • • • • •	• • • • • • • •	• • • • • • • •			**0.5** / 0.8 **0.5** / 0.8 **0.5** / 0.8 **0.75** / 1.2 **3.0**[1] **5.8**[2] / 9[1] 11[2] **0.25** / 0.4 **0.33** / 0.5 **1.00** / 1.6	ADC W I0,A ADC BY K0,B ADC W DP0,C ADC BY D0,D ADC W II0,A ADC BY B,C ADC W [C],D ADC BY P0,A		Fixed-point addition of signed integers with consideration Übertrages (Carry) Source + destination + Cy = DEST [1] when OPA = BY [2] when OPA = W

Byte or Word addition

ADD B/W B , O



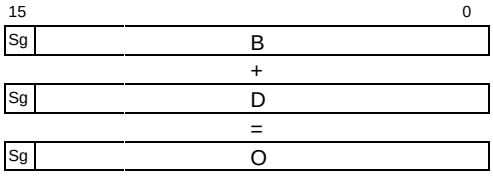
ADC B/W C , O



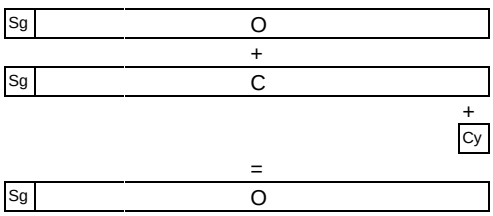
Double-word addition: value 1 + value 2

Value 1: LOW word in B, HIGH word in A
Value 2: LOW word in D, HIGH word in C

LOW word
ADD W D , B



HIGH word
ADC W C , O



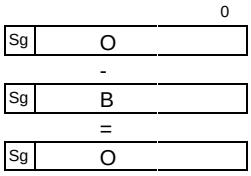
8.12.2 Subtract Instructions

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500				
SUB	W	I/O/M/SM	, R			•				•	•	•	•			**0.5** / 0.8	SUB	W	I0,A	Fixed-point subtraction
	BY	T/C/K/SYM				•				•	•	•	•			**0.5** / 0.8	SUB	BY	K0,B	of signed integers.
		S/DP/DF				•				•	•	•	•			**0.5** / 0.8	SUB	W	DP0,C	
		D/DX				•				•	•	•	•			**0.75** / 1.2	SUB	BY	D0,D	Destination - source = DEST
		II/EI				•				•	•	•	•			**3.0**[1] **5.8**[2] / 9[1] 11[2]	SUB	W	II0,A	[1] when OPA = BY
		R					•									**0.25** / 0.4	SUB	BY	B,C	[2] when OPA = W
		[R]						•		•	•	•	•			**0.33** / 0.5	SUB	W	[C],D	
		P				•				•	•	•	•			**1.00** / 1.6	SUB	BY	P0,A	
SBB	W	I/O/M/SM	, R			•				•	•	•	•			**0.5** / 0.8	SBB	W	I0,A	Fixed-point subtraction
	BY	T/C/K/SYM				•				•	•	•	•			**0.5** / 0.8	SBB	BY	K0,B	of signed integers
		S/DP/DF				•				•	•	•	•			**0.5** / 0.8	SBB	W	DP0,C	with consideration
		D/DX				•				•	•	•	•			**0.75** / 1.2	SBB	BY	D0,D	of negative CARRY. (- Carry =
		II/EI				•				•	•	•	•			**3.0**[1] **5.8**[2] / 9[1] 11[2]	SBB	W	II0,A	Borrow)
		R					•									**0.25** / 0.4	SBB	BY	B,C	Destination - source - Cy = DEST
		[R]						•		•	•	•	•			**0.33** / 0.5	SBB	W	[C],D	[1] when OPA = BY
		P				•				•	•	•	•			**1.00** / 1.6	SBB	BY	P0,A	[2] when OPA = W

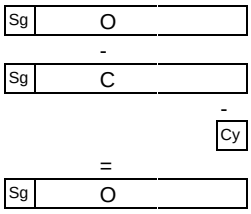
Byte or Word subtraction

Bit 7 when OPA = BY
MSB Bit 15 when OPA = W

SUB *OPA* B , O



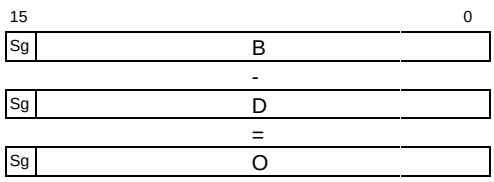
SBB *OPA* C , O



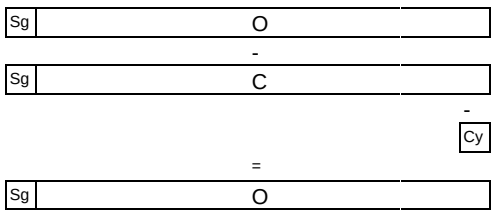
Double-word subtraction: value 1 – value 2

Value 1: LOW word in B, HIGH word in A
Value 2: LOW word in D, HIGH word in C

LOW word
SUB D D , B



HIGH word
SBB W C , O



8.12.3 Multiply Instructions

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x,40x 501 / 500		
MUL	W	K	, R			•				0	0	•	•			**80** / 140	MUL BY K100,A	Fixed-point multiplication of signed integers.
	BY	R					•			0	0	•	•			**65** / 120	MUL W BA	Byte instruction: SRC BY x DEST BY = DEST W Word instruction: SRC W x DEST W = DEST W and DEST W +1. Refer to functional description below.

The product of all Multiply operations occupies twice the width of the starting operands.

Byte multiplication

```
MUL  B   B , O   ; SRC BY          7 ... 0
                                   Sg | B
                                      x
                 ; DEST BY         Sg | O
                                      =
                 ; DEST W      15 ... 0
                               Sg | O
```

Word multiplication

```
MUL  W   B , O   ; SRC W       15 ... 0
                               Sg | B
                                  x
                 ; DEST W      Sg | O

                                  =
                 ; DEST W          | O
                                  u n d
                 ; DEST        Sg | B
                 W+1
```

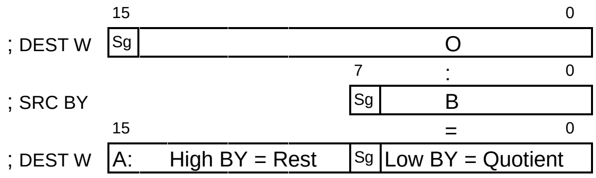
8.12.4 Divide Instructions

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x,40x 501 / 500		
DIV	W BY	K R	, R			•	 •			0 0	• •	• •	• •			85 / 150 70 / 130	DIV BYW K100,A DIV B,A	Fixed-point division of signed integers. Byte instruction: DEST W : SRC BY = DEST W (REST/remainder) Word instruction: DEST DW : SRC W = DEST W and DEST W+1 REST remainder Refer to functional description below.

The dividend of all Divide operations occupies twice the width of the divisor.

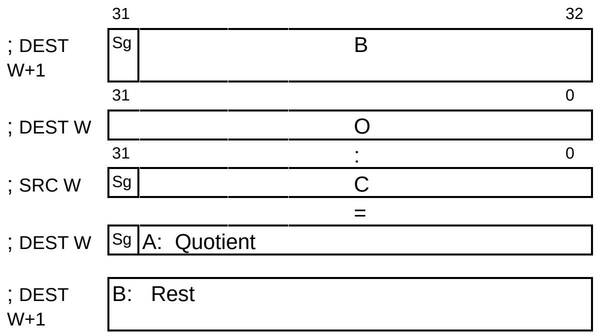
Byte division

DIV B B , O



Word division

DIV W C , O



⇒ **In the case of a Divide operation by 0, the Divide instruction will not be carried out, and the overflow bit will be set.**

8.13 Floating-point Arithmetic (not available on ZS500)

Effective with WinSPS v2.40, instructions handling floating-point arithmetic operations are available for the CL350, CL400 and CL500 controllers (except ZS500). These instructions support data formats defined in the IEEE 754 and IEE 854 standards.

8.13.1 Data Formats

Pursuant to IEC 1131, two data formats, i.e., REAL and LREAL, are defined.

Data format	Data width	Mantissa	Exponent	Range
REAL: Short real floating-pt. number Single Precision	32 bits	24 bits	8 bits	$10^{\pm 38}$
LREAL: Long real floating-pt. number Double Precision	64 bits	53 bits	11 bits	$10^{\pm 308}$

The data format "R" (short real floating-point number) always uses the Register pairs AB and CD.

The data format "L" (long real floating-point number) utilizes all Registers (ABCD).

8.13.2 Floating-point Operands

Depending on the instruction, the following floating-point operands can be used:

- M, DB, DF, D, DX, with either direct or indirect [ ] addressing.
  The specified operand address must be divisible as follows:
  - for REAL by 4, and
  - for LREAL by 8.
- K, Register

8.13.3 Indirect Addressing

In the Real format, when Register A (C) are defined as source registers, then Register C (A) must be specified as the destination register.

8.13.4 Floating-point Instructions

The floating-point formats and operands discussed in the foregoing can be used in the collowing instructions:

- Load
- Transfer
- Convert
- Compare
- Arithmetic (Add, Subtract, Multiply, Divide)

8.13.5 Value Range and Resolution

In the floating-point formats, not all numbers can be represented in any desired resolution. For example, in the REAL data format, if one wishes to work with a unit of measure that is quite common in machine tool engineering, such as **µm**, each individual **µm** can be represented up to the limit value of 16.0 **m**. By contrast, using the LREAL format instead facilitates the representation of numbers up to 17,179,869,184.0 **m**.

Resolution		Limit value	
Floating-point notation	Expotential notation	REAL	LREAL
1.0	E^{0}	16,777,228.0	18,014,398,509,481,984.0
0.1	E^{-1}	1,048,576.0	1,125,899,906,842,624.0
0.01	E^{-2}	131,072.0	140,737,488,355,328.0
0.001	E^{-3} milli (m)	16,384.0	17,592,186,044,416.0
0.0001	E^{-4}	1,024.0	1,099,511,627,776.0
0.00001	E^{-5}	128.0	137,438,953,472.0
0.000001	E^{-6} micro (µ)	16.0	17,179,869,184.0
0.0000001	E^{-7}	1.0	1,073,741,824.0
0.00000001	E^{-8}	0.125	134,217,728.0
0.000000001	E^{-9} nano (n)	0.015625	16,777,216.0
0.0000000001	E^{-10}	0.000976563	1,048,576.0
0.00000000001	E^{-11}		131,072.0
0.000000000001	E^{-12} pico (p)		16,384.0
0.0000000000001	E^{-13}		1,024.0
0.00000000000001	E^{-14}		128.0
0.000000000000001	E^{-15} femto (f)		16.0
0.0000000000000001	E^{-16}		1.0
0.00000000000000001	E^{-17}		0.125
0.000000000000000001	E^{-18} atto (a)		0.015625

8.13.6 Programming Unit Display in Monitor Mode

REAL:

In the REAL format, eight significant digits are displayed in IL – Monitor mode. Starting at a numerical value of 10,000,000, the display automatically switches to exponential notation.

LREAL:

In the LREAL format, eleven significant digits are displayed in IL – Monitor mode. Starting at a numerical value of 100,000,000,000, the display automatically switches to exponential notation.

⇒ **Due to the execution of Compare, Load, Transfer, and some Divide instructions in macro mode (one Load instruction consisting of n number of IL instructions), a situation may soon occur where the monitor information required to write the display can no longer be supplied within a single cycle. This condition will be confirmed by an appropriate comment in the status bar.**

8.13.7 Error Displays & Range Violations

In Monitor mode, calculation errors and over-range violations are identified on the display of the programming unit.

Over-range violation of the largest possible number, and division by 0:

```
++++++.++++++
```

Falling short (under-range violation) of smallest possible number:

```
------.------
```

Non-displayable number and calculation error:

```
******.******
```

8.13.8 Loading Floating-point Values

Control Instruction				RG		Addr Mode			Instruct. Length	Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	Words	35x, 40x 501		
L	R	K R M/DP/DF/D/DX [R]	, R			• •	 • 	 •	7 5 7 6		L R 12.321,A L R A,C L R M8,C L R [A],C	REAL constant → Reg. AB REAL Reg. AB → CD REAL M8-M11 → Reg. CD REAL contents of operand addressed via Reg. A → Reg.CD
	L	K M/DP/DF/D/DX [R]	, R			• •	 	 •	11 11 11		L L 12.321,A L L M8,A L L [A],A	LREAL constant → Reg. ABCD LREAL M8-M15 → Reg.ABCD LREAL REAL contents of operand addressed via Reg. A → Reg. ABCD

For these instructions, instruction sequences comprising existing opcodes are assembled. As the Load instruction for indirect addressing contains INC and DEC instructions, modification of the flags may occur.

⇒ **In the LREAL format, the index register (operand address) is always overwritten. A possible remedy in this case is the rescue of the register with the use of PUSH and/or POP instructions.**

8.13.9 Transferring Floating-point Values

Control Instruction				RG		Addr Mode			Instruct. Length	Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	Byte	35x,40x 501		
T	R	R	, M/DP/DF/D/DX [R]			•		•	7 6		T R A,M8 T R C,[A]	REAL Reg. CD → M8-M11 REAL Reg. contents of CD → the operand addressed by Reg. A
	L	R	, M/DP/DF/D/DX			•			11		T L A,M8	REAL Reg. CD → M8-M15

For these instructions, instruction sequences comprising existing opcodes are assembled. As the Load instruction for indirect addressing contains INC and DEC instructions, modification of the flags may occur.

8.13.10 Converting Number Formats (Floating-point ←→ Integer)

Converting 32-bit integer values to floating-point REAL/LREAL (ITF)

Converting 16-bit integer values to floating-point REAL/LREAL (ITFW)

Control Instruction				RG		Addr Mode			Instruct. Length	Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	Words	35x,40x 501		
ITF	R	R					•		1		ITF R O ITF R C	Coverts 32-bit integer value from Register AB to REAL floating-point (result in AB). Coverts 32-bit integer value from Register CD to REAL floating-point (result in CD).
	L	R					•		1		ITF L O	Coverts 64-bit integer value from Register ABCD to LREAL floating-point (result in ABCD).
ITFW	R	R					•		1		ITFW R O ITFW R C	Coverts 16-bit integer value from Register A to REAL floating-point (result in Register AB). Coverts 16-bit integer value from Register C to REAL floating-point (result in Register CD).
	L	R					•		1		ITFW L O	Coverts 16-bit integer value from Register A to LREAL floating-point (result in Register ABCD).

Converting floating-point REAL/LREAL to 32-bit integer values (FTI)

Converting floating-point REAL/LREAL to 16-bit integer values (FTIW)

Control Instruction				RG		Addr Mode			Instruct. Length	Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	Worte	35x, 40x 501		
FTI	R	R					•		1		FTI R A FTI R C	Converts REAL floating-point from Reg. AB to 32-bit integer value (result in AB). Converts REAL floating-point from Reg. CD to 32-bit integer value (result in CD).
	L	R					•		1		FTI L A	Converts LREAL floating-point from Reg. ABCD to 64-bit integer value (result in ABCD).
FTIW	R	R					•		1		FTIW R A FTIW R C	Converts REAL floating-point from Reg. AB to 16-bit integer value (result in A). Converts REAL floating-point from Reg. CD to 16-bit integer value (result in C).
	L	R					•		1		FTIW L A	Converts LREAL floating-point from Reg. ABCD to 16-bit integer value (result in ABCD).

Notes on handling Convert instructions:

When converting from double word to REAL or LREAL, the LOW word is in Register A or C, and the corresponding HIGH word in Register B or D.

```
L  W7,C      ;C = 00007
L  W1,D      ;D = 00001 (65536)
ITF R   C    ;C = 65543.00000
```

In the case of integer Convert instructions with a 16-bit destination register, over-range violations cause the overflow flag to go HIGH. Negative Carry and Zero are always LOW.

The overflow flag will also go HIGH with the numerical value of 0.0.

8.13.11 Comparing Floating-point Values

Control Instruction				RG		Addr Mode			Instruct. Length	Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	Words	35x, 40x 501		
CPLA	R	K	, R			•			3		CPLA R 12.321,A	REAL constant with Reg. AB
		R					•		1		CPLA R A,C	REAL Reg. AB with Reg. CD
		M/DP/DF/D/DX				•			2		CPLA R M8,C	REAL M8-M11 with Reg. CD
		[R]						•	1		CPLA R [A],C	REAL contents of operand addressed with Reg. A, with Reg. CD.
	L	K	, R			•			5		CPLA L 12.321,A	LREAL conastant with Reg. ABCD
		M/DP/DF/D/DX				•			2		CPLA L M8,A	LREAL M8-M15 with Reg. ABCD

When comparing in the REAL and LREAL data formats, the negative and zero status bits (flags) must be interpreted arithmetically. As these flags are also interpreted in the special markers, they, too, may be utilized for interpretation.

The easiest means of interpretation is by means of Jump instructions.

Examples:

Compare Destination (A) w/ Source (B)		**CPLA B,A** Flag
Equal	A=B	JPZ
Unequal	A≠B	JPN
Less than	A<B	JPM
Less than or equal	A≤B	JPMZ
Greater	A>B	JPAG
Greater than or equal	A≥B	JPP

⇒ **The interpretation of Compare results should always be programmed to occur immediately after the Compare instruction itself.**

⇒ **In the case of different resolutions in the REAL data format, the Compare operation will return correct results only within specific limit values (refer to Section 8.13.5, "Value Range & Resolution").**

Example:

```
L    R2048.00000,A
CPLA R2048.00009,A      ;the differential will not be found,
                        ;and the numbers recognized as being
                        ;equal, Z=1
```

For large numbers with high resolutions, the LREAL data format must be used.

8.13.12 Calculating with Floating-point Values

For floating-point processing, the basic arithmetical operations of addition, subtraction, multiplication and division are available.

The instructions for the four basic arithmetic operations calculate the contents of the destination register and/or register pair to the contents of the source operand. The obtained results are then always written into the destination register and/or register pair.

Control Instruction				RG		Addr Mode			Instruct. Length	Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	Words	35x, 40x 501				
ADD	R	K	, R			•			3		ADD	R	12.321,A	REAL Reg. AB + constant
		R					•		1		ADD	R	A,C	REAL Reg. CD + Reg. AB
		M/DP/DF/D/DX				•			2		ADD	R	M8,C	REAL Reg. CD + M8-M11
		[R]						•	1		ADD	R	[A],C	REAL Reg.CD + Contents of operand addressed by Reg. A.
	L	K	, R			•			5		ADD	L	12.321,A	LREAL Reg. ABCD + constant
		M/DP/DF/D/DX				•			2		ADD	L	M8,A	LREAL Reg.ABCD + M8-M15
SUB	R	K	, R			•			3		SUB	R	12.321,A	REAL Reg. AB - constant
		R					•		1		SUB	R	A,C	REAL Reg. CD - Reg. AB
		M/DP/DF/D/DX				•			2		SUB	R	M8,C	REAL Reg. CD - M8-M11
		[R]						•	1		SUB	R	[A],C	REAL Reg.CD - Contents of operand addressed by Reg. A.
	L	K	, R			•			5		SUB	L	12.321,A	LREAL Reg. ABCD - constant
		M/DP/DF/D/DX				•			2		SUB	L	M8,A	LREAL Reg.ABCD - M8-M15
MUL	R	K	, R			•			3		MUL	R	12.321,A	REAL Reg. AB x constant
		R					•		1		MUL	R	A,C	REAL Reg. CD x Reg. AB
		M/DP/DF/D/DX				•			2		MUL	R	M8,C	REAL Reg. CD x M8-M11
		[R]						•	1		MUL	R	[A],C	REAL Reg.CD x Contents of operand addressed by Reg. A.
	L	K	, R			•			5		MUL	L	12.321,A	LREAL Reg. ABCD x constant
		M/DP/DF/D/DX				•			2		MUL	L	M8,A	LREAL Reg.ABCD x M8-M15
DIV	R	K	, R			•			3		DIV	R	12.321,A	REAL Reg. AB : constant
		R					•		1		DIV	R	A,C	REAL Reg. CD : Reg. AB
		M/DP/DF/D/DX				•			2		DIV	R	M8,C	REAL Reg. CD : M8-M11
		[R]						•	1		DIV	R	[A],C	REAL Reg.CD : Contents of operand addressed by Reg. A.
	L	K	, R			•			5		DIV	L	12.321,A	LREAL Reg. ABCD constant
		M/DP/DF/D/DX				•			2		DIV	L	M8,A	LREAL Reg.ABCD : M8-M15

When using floating-point arithmetical operatiojns, only the negative and zero status bits (flags) are formed. As these flags are also interpreted in the special markers, they, too, may be utilized for interpretation.

⇒ **There is no provision for over-range violation monitoring.**

8.14 Multiple Word Instructions

8.14.1 FIFO Instructions

Each ZS5xx provides four FIFO buffers (FI0 through FI3), with a capacity of 512 bytes each. The number of bytes to be processed is first loaded into Register C. In the case of register addressing, the number is defined by OPA BY (1 byte) or W (2 bytes).

A register underflow or overflow causes the status bits C and/or Z to go HIGH.

When using indirect addressing, only OPD addresses available for direct addressing may be used.

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501 / 500				
LFI	BY	FIn	, M/S/SYM			•				Ü:•			U:•							Read from FIFO buffer
			, D/DX			•				Ü:•			U:•							
			, DP/DF			•				Ü:•			U:•			155 / 245	LFI	BY	FI2,DP30	Number of bytes in C
			, [R]					•		Ü:•			U:•			+11 / 19 per	LFI	BY	FI3,[A]	'' '' ''
	BY		, R				•			Ü:•			U:•			BY	LFI	BY	FI0,A	One byte from FIFO into Reg. A
	W																LFI	W	FI0,A	Two bytes from FIFO into Reg. A
																135 / 235				
TFI	BY	M/S/SYM	, FIn			•				Ü:•			U:•							Write to FIFO buffer.
		D/DX				•				Ü:•			U:•							
		DP/DF				•				Ü:•			U:•			180 / 300	TFI	BY	DP30,FI2	Number of bytes in C
		[R]						•		Ü:•			U:•			+3 / 5 per	TFI	BY	[A],FI3	'' '' ''
	BY	R					•			Ü:•			U:•			BY	TFI	BY	A,FI0	One byte from Reg. A into FIFO
	W																TFI	W	A,FI0	Two bytes from Reg. A into FIFO
																135 / 235				
RFI		FIn														115 / 200	RFI		FI0	Reset / Clear FIFO buffer

8.14.2 Block Commands (not available for ZS500)

The number of words (0-256) or bytes (1-512) to be processed is first loaded into Register C.

When using indirect addressing, only OPD addresses available for direct addressing may be used.

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	35x, 40x 501/500		
BLT	BY W	M/S/SYM D/DX DP/DF [B]	, M/S/SYM , D/DX , DP/DF , [A]			• • •		•								45 +0.9 per BY	BLT BY M0,D0 BLT W [B],[A]	Block transfer from Source address → destination addr. Block size in Register C.
VVxx VRxx	BY W	[B]	, C				•	•								45 +0.9 per BY	CFZ W [B],C VRN BY [B],C	Compare forwrd.within block Compare backwrd. within block DEST block addr. always in Reg. A SRC block address in Reg. B Block size in Register C.
SVxx SRxx	BY W	B K	, C , K			• •	•									45 +0.9 per BY	SVZ W B,C SRLG BY K35D,K100D	Search forwrd.within block Search backwrd. within block Block start address always in Reg. A Z=Block size Q=Search value
xx: C/Z/N/M CN/CZ AG/LG																		8 Compare / Search criteria through interpretation of status bits, and combinations thereof.

For more information about this subject, refer to Section 7.18, "Block Commands" and 9.3, "Sample Applications of Block Commands".

8.15 Definitions

8.15.1 Parameter Assignments

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	(µs)				
Pn	W BY B	I/O/M/T/C/K II/IO/EI/EO S/SM/SYM D/DX /DP/DF PB/DM				• • • • •											P0 P1 P2 P3 P4	B BY W W	E0.0 IE0 S0 D0 PM0	Call up parameter definition for parameterized module. n: 0-62
PSn	W	K															PS0	W	K0000H	Parameter definition for system commands

8.15.2 Local Symbol Names & Auxiliary Flags for Program Tracking

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	(µs)				
DEF		I/O/M/T/C/K II/IO/EI/EO S/SM/SYM D/DX /DP/DF PB/DM	, SYM														DEF DEF		E0.0,-Symbol E0,-Name	Definition of symbolic names that are valid only within the module ("local") in which they have been entered (essential for the creation of library modules).
*		n n = 0-63															*		1	Definition of auxiliary flags for program tracking. Processing of these aux. flags is entered only in the marker buffer, and is interpretable only in the case of an error. *n has no influence on the program.

8.15.3 System Variable

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time	Example			Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	(µs)				
DEFW	W	K															DEFW	W	K0000H	Definition of function for system variable in OM2.

8.15.4 Parenthesized & No-operation Instructions, CARRY Manipulations

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs) 35x, 40x 501 / 500	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4			
( ) O( )N																**0.25** / 0.4 **0.25** / 0.4 **0.25** / 0.4 **0.25** / 0.4	( ) O( )N	AND open parenthesis close parenthesis OR open parenthesis Negation of contents enclosed in parentheses
NOP0 NOP1																**0.25** / 0.4 **0.25** / 0.4	NOP0 NOP1	No-operation with zeroes in memory location No-operation with ones in memory location
SCY RCY										• •						**0.25** / 0.4 **0.25** / 0.4	SCY RCY	Essential to set CARRY bit HIGH Essential to set CARRY bit LOW

8.16 Program Processing Instructions

8.16.1 Jump Instructions

Jump operations may be executed unconditionally, and also in dependence of a binary link and/or mathematical operation. With one exception, Jump operations are programmed symbolically (exception: JP [R]). This dispenses with the necessity to observe the enter point because that is calculated by the PG programming unit in any case.

Control Instruction				RG		Addr Mode			Dependent on Status bit							Time (µs)	Example		Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C	O	M	Z	AG	LG	35x, 40x 501 / 500 exec.;not exec			
JP		SYM [R]				•		•								**0.5** / 0.8	JP JP	-LABEL1 [A]	Unconditional to –LABEL destination. Unconditional, by jump distance (words) in Register A.
JPC		SYM			•	•			1								JPC	-LABEL2	conditional, see status bit
JPCI		SYM			•	•			0								JPCI	-LABEL3	conditional, see status bit
JPCY		SYM				•				1							JPCY	-LABEL4	conditional, see status bit
JPCN		SYM				•				0							JPCN	-LABEL5	conditional, see status bit
JPO		SYM				•					1						JPO	-LABEL6	conditional, see status bit
JPON		SYM				•					0					**0.33;0.5** /	JPON	-LABEL7	conditional, see status bit
JPM		SYM				•						1				0.5;0.8	JPM	-LABEL8	conditional, see status bit
JPP		SYM				•						0					JPP	-LABEL9	conditional, see status bit
JPZ		SYM				•							1				JPZ	-LABEL10	conditional, see status bit
JPN		SYM				•							0				JPN	-LABEL11	conditional, see status bit
JPAG		SYM				•								1			JPAG	-LABEL12	conditional, see status bit
JPMZ		SYM				•								0			JPMZ	-LABEL13	conditional, see status bit
JPLG		SYM				•									1		JPLG	-LABEL14	conditional, see status bit
JPCZ		SYM				•									0		JPCZ	-LABEL15	conditional, see status bit

The JP [R] instruction causes an unconditional jump whose jump distance in words can be calculated in the PLC program. However, because PLC instructions are of different length, this jump should be used only in accordance with the following example. If this is the case, this will will facilitate the risk-free implementation of program changes located outside of this instruction sequence.

Example:

PLC program interlude

```
Jump distance calculation in register A for the following jump sequence:
A may have odd-numbered values only (1, 3, 5, ...).
JP      [A]             ; 1-word instruction        Fixed program sequence
JP      -Ziel1          ; 2-word instruction
JP      -Ziel2          ; 2-word instruction
  :
JP      -Zieln          ; 2-word instruction
```

```
      -Ziel1            ; Partial program 1
PLC program
JP      -Ende
```

```
      -Ziel2            ; Partial program 2
PLC program
JP      -Ende
  :
```

```
      -Zieln            ;Teilprogramm n
PLC program
SP      -Ende
      -Ende
PLC successor program
  :
```

8.16.2 Module Calls

Module calls may be executed unconditionally and also in dependence of a binary link and/or a mathematical operation.

The module nesting depth is 32 program modules.

Two data modules may be kept enabled at the same time. For this purpose the following module calls are available:

CM, CMC, CMCI DMx: enables DMx as 1st DM

CX, CXC, CXCI DMy: enables DMy as 2nd DM

Control Instruction				RG		Addr Mode			Dependent on Status bit							Time	Example		Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C	O	M	Z	AG	LG	(µs)			
CM CX		DM PM PM P P [R]	 , n , n 			• • • • • •		• 								 ↓	CM CM CM CM CM CM	DM0 PM0 PB1,2 P0 P0,2 [A]	Unconditional, direct Parameterized, list to follow as parameter als Parameter, para. Indirect
CMC CXC		DM PM PM P P [R]	 , n , n 		•	• • • • • •		• 	1							 ↓	CMC CMC CMC CMC CMC CMC	DM0 PM0 PB1,2 P0 P0,2 [A]	conditional, see status bit Direct Parameterized, list to follow as parameter als Parameter, para. Indirect
CMCI CXCI		DM PM PM P P [R]	 , n , n 		•	• • • • • •		• 	0							 ↓	CMCI CMCI CMCI CMCI CMCI CMCI	DM0 PM0 PB1,2 P0 P0,2 [A]	conditional, see status bit Direct Parameterized, list to follow as parameter als Parameter, para. Indirect
CMCY		DM PM PM P P [R]	 , n , n 			• • • • • •		• 		1						 ↓	CMCY CMCY CMCY CMCY CMCY CMCY	DM0 PM0 PB1,2 P0 P0,2 [A]	conditional, see status bit Direct Parameterized, list to follow as parameter als Parameter, para. Indirect
CMCN		DM PM PM P P [R]	 , n , n 			• • • • • •		• 		0						 ↓	CMCN CMCN CMCN CMCN CMCN CMCN	DM0 PM0 PB1,2 P0 P0,2 [A]	conditional, see status bit Direct Parameterized, list to follow as parameter als Parameter, para. Indirect
CMO		DM PM PM P P [R]	 , n , n 			• • • • • •		• 			1					 ↓	CMO CMO CMO CMO CMO CMO	DM0 PM0 PB1,2 P0 P0,2 [A]	conditional, see status bit Direct Parameterized, list to follow as parameter als Parameter, para. Indirect
CMON		DM PM PM P P [R]	 , n , n 			• • • • • •		• 			0					 ↓	CMON CMON CMON CMON CMON CMON	DM0 PM0 PB1,2 P0 P0,2 [A]	conditional, see status bit Direct Parameterized, list to follow as parameter als Parameter, para. Indirect

Instr. **35x, 40x, 501** / 500 not executed; executed

DM	**0.50;0.75** / 0.8;1.2	P	**0.50;2.08** / 0.8;3.5: PM
PM	**0.50;1.58** / 0.8 / 2.7		**1.25** / 2.0 : DM
PM,n	**0.58;0.67** / 0.9 / 2.8	P,n	**0.58;2.17** / 0.9;3.6
		[R]	**0.33;1.42** / 0.5;2.4: PM
			0.58 / 0.9 : DM
		[R],n	**0.33;1.50** / 0.5;2.5

Module calls (cont'd)

Control Instruction				RG		Addr Mode			Dependent on Status bit							Time	Example		Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C	O	M	Z	AG	LG	(µs)			
CMM		DM				•											CMM	DM0	conditional, see status bit
		PM				•											CMM	PM0	Direct
		PM	, n			•											CMM	PB1,2	Parameterized, list to follow
		P				•											CMM	P0	as parameter
		P	, n			•											CMM	P0,2	als Parameter, para.
		[R]						•								↓	CMM	[A]	Indirect
CMP		DM				•			1								CMP	DM0	conditional, see status bit
		PM				•											CMP	PM0	Direct
		PM	, n			•											CMP	PB1,2	Parameterized, list to follow
		P				•											CMP	P0	as parameter
		P	, n			•											CMP	P0,2	als Parameter, para.
		[R]						•								↓	CMP	[A]	Indirect
CMZ		DM				•			0								CMZ	DM0	conditional, see status bit
		PM				•											CMZ	PM0	Direct
		PM	, n			•											CMZ	PB1,2	Parameterized, list to follow
		P				•											CMZ	P0	as parameter
		P	, n			•											BAZ	P0,2	als Parameter, para.
		[R]						•								↓	CMZ	[A]	Indirect
CMN		DM				•				1							CMN	DM0	conditional, see status bit
		PM				•											CMN	PM0	Direct
		PM	, n			•											CMN	PB1,2	Parameterized, list to follow
		P				•											CMN	P0	as parameter
		P	, n			•											CMN	P0,2	als Parameter, para.
		[R]						•								↓	CMN	[A]	Indirect
CMAG		DM				•				0							CMAG	DM0	conditional, see status bit
		PM				•											CMAG	PM0	Direct
		PM	, n			•											CMAG	PB1,2	Parameterized, list to follow
		P				•											CMAG	P0	as parameter
		P	, n			•											CMAG	P0,2	als Parameter, para.
		[R]						•								↓	CMAG	[A]	Indirect
CMMZ		DM				•					1						CMMZ	DM0	conditional, see status bit
		PM				•											CMMZ	PM0	Direct
		PM	, n			•											CMMZ	PB1,2	Parameterized, list to follow
		P				•											CMMZ	P0	as parameter
		P	, n			•											CMMZ	P0,2	als Parameter, para.
		[R]						•								↓	CMMZ	[A]	Indirect
CMLG		DM				•					0						CMLG	DM0	conditional, see status bit
		PM				•											CMLG	PM0	Direct
		PM	, n			•											CMLG	PB1,2	Parameterized, list to follow
		P				•											CMLG	P0	as parameter
		P	, n			•											CMLG	P0,2	als Parameter, para.
		[R]						•								↓	CMLG	[A]	Indirect
CMCZ		DM				•					0						CMCZ	DM0	conditional, see status bit
		PM				•											CMCZ	PM0	Direct
		PM	, n			•											CMCZ	PB1,2	Parameterized, list to follow
		P				•											CMCZ	P0	as parameter
		P	, n			•											CMCZ	P0,2	als Parameter, para.
		[R]						•								↓	CMCZ	[A]	Indirect

Instr. **35x, 40x, 501** / 500 not executed;

			executed
DM	**0.50;0.75** / 0.8;1.2	P	**0.50;2.08** / 0.8;3.5: PM
PM	**0.50;1.58** / 0.8 / 2.7		**1.25** / 2.0: DM
PM,n	**0.58;0.67** / 0.9 / 2.8	P,n	**0.58;2.17** / 0.9;3.6
		[R]	**0.33;1.42** / 0.5;2.4: PM
			0.58 / 09 : DM
		[R],n	**0.33;1.50** / 0.5;2.5

8.16.3 End Of Module Instructions

End Of Module instructions may be executed unconditionally, and also in dependence of a binary link and/or mathematical operation.

Control Instruction				RG		Addr Mode			Dependent on Status bit							Time	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C	O	M	Z	AG	LG	(µs)		
EM																↓	EM	Unconditional
EMC									1							↓	EMC	conditional, see status bit
EMI									0							↓	EMI	conditional, see status bit
EMCY										1						↓	EMCY	conditional, see status bit
EMCN										0						↓	EMCN	conditional, see status bit
EMO											1					↓	EMO	conditional, see status bit
EMON											0					↓	EMON	conditional, see status bit
EMM												1				↓	EMM	conditional, see status bit
EMP												0				↓	EMP	conditional, see status bit
EMZ													1			↓	EMZ	conditional, see status bit
EMN													0			↓	EMN	conditional, see status bit
EMAG														1		↓	EMAG	conditional, see status bit
EMMZ														0		↓	EMMZ	conditional, see status bit
EMLG															1	↓	EMLG	conditional, see status bit
EMCZ															0	↓	EMCZ	conditional, see status bit

Instr. **35x, 40x, 501 /** 500 not executed; executed
0.25;1.17 / 0.4;2.0

8.16.4 Mask Instructions Enabling / Disabling Coordination Markers

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	**35x, 40x 501 /** 500		
TIM		R	, KME/KMP			•										**190 /** 320	TIM A,KME	Transfer mask Writes mask for disabling or enabling the coordination markers. The mask was first loaded into a register.

8.16.5 Interrupt Instructions

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time (µs)	Example		Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	**35x, 40x** / **501** / 500			
TIM		R	, TI/PI/SI			•										**190** / 320	TIM	A,TI	Transfer interrupt mask. Writes interrupt mask for disabling or enabling interrupts. The mask was first loaded into a register.
LIM		TI/PI/SI	, R			•										**85** / 150	LIM	PI;B	Load interrupt mask Define interrupt mask
EAI		TI/PI/SI				•										**200-215** / 330-360	EAI	SI	Enable interrupt group (activate)
DAI		TI/PI/SI				•										**90-115** / 160-200	DAI	SI	Disable interrupt group (deactivate)
LAI		TI/PI/SI	, R			•										**85-115** / 150-200	LAI	SI	Load interrupt register (Read statuses)
RAI		R	, TI/PI/SI			•										**85-115** / 150-200	RAI	A,TI	Resetting interrupts in accordance with previously loaded mask.
As the following interrupt instructions were implemented only due to CL300 compatibility considerations, they cover a significantly narrower range of interrupt handling options.																			
EI		n				•										**140** / 240	EI EI	1	Peripheral interrupt n (0-3) is enabled in the interrupt mask. If SRC operand is not defined, all interrupts will be addressed.
DI		n				•										**140** / 240	DI DI	2	Peripheral interrupt n (0-3) is disabled in the interrupt mask. If SRC operand is not defined, all interrupts will be addressed.
RI		n				•										**140** / 240	RI RI	3	Peripheral interrupt n (0-3) is reset in the interrupt mask. If SRC operand is not defined, all interrupts will be addressed.

8.16.6 Program Stop / End

Control Instruction				RG		Addr Mode			Influences status bit & SM... special marker							Time	Example	Comment
OPP	OPA	SRC	DEST	O	I	D	R	[R]	V	C 31.3	O 31.5	M 31.6	Z 31.7	AG 31.0	LG 31.4	(µs)		
HLT																	HLT	HALT instruction. The controller enters STOP mode, the program address is entered in error stack, and outputs are cleared (deleted).
EP																	EP	Program End The I/O state is initialized, and the program cycle starts again at the beginning. At least one EP must be present.

8.17 System Commands

In the CL400 and CL500 systems, the term "system commands" describes all those PLC instructions that can be used to accomplish a data exchange between two system modules, e.g., between two ZS500 or between ZS40x and ZATx, while operating in the same basic unit.

To to the extensive and differentiated addressing options for system commands, these appear only in list form in the table below. For detailed functional descriptions, refer to the separate publication:

CL500 System Commands, order no. 1070 072 068

Overview of CL400 / CL500 system commands:

Comm.	Extension	Explanation	Note
LFD	W/BY	Load field directly	
LFB [1)]	W/BY	Load field in background	
LFS [1)]	W/BY	LFB with system interrupt	Example in above documentation.
TFD	W/BY	Transfer field directly	Example in above documentation. See also 9.4, "Setting the System Clock"
TFB [1)]	W/BY	Transfer field in background	Example in above documentation.
TFS [1)]	W/BY	TFB with system	
FS	none	Field save	For ZS40x / ZS501 only
FR	none	Field release	For ZS40x / ZS501 only
LAD [1)]	W/BY	Load absolute range directly	Recommendation: If possible, use LFD field command.
LAB [1)]	W/BY	Load absolute range in background	Recommendation: If possible, use LFB field command.
LAS [1)]	W/BY	LAB with system interrupt	Recommendation: If possible, use LFS field command.
TAD [1)]	W/BY	Transfer absolute range directly	Recommendation: If possible, use TFD field command.
TAB [1)]	W/BY	Transfer absolute range in background	Recommendation: If possible, use TFB field command.
TAS [1)]	W/BY	TAB with system	Recommendation: If possible, use TFS field command.
LCC	none	Load clock cyclical	Examples provided
LCCS	none	LCC with system interrupt	
AB	none	Alarm bell	
ABS	none	AB with system interrupt	Example provided
ABC	none	Alarm bell request cyclical	
ABCS	none	ABC with system interrupt	Example provided
CLSI	none	Clear system instruction	Example provided
SINT [1)]	none	Send system interrupt	Destination module: For ZS40x / ZS501
SYN [1)]	none	Synchronization point achieved	For ZS501 only
EVA	none	Event achieved	For ZS40x / ZS501 only
EVD [1)]	none	Event instruction directly, U	For ZS40x / ZS501 only
EVB [1)]	none	Event instruction background, H	For ZS40x / ZS501 only
EVS [1)]	none	Event with system interrupt, S	For ZS40x / ZS501 only

[1)] not for CL350

9 Sample Programs

9.1 Indirect Addressing Examples

```
          DEF     O0.0,-Ergebnis
          DEF     I4.0,-count
          DEF     I8.0,-time
          DEF     SM31.1,-log1

          ;Bit-indirect addressing of counter C3 status:
1         L     W  K8003H,A  ;Indirect bit address for status of counter C3
          PZ:   1
2         U     B  [A]       ;Status bit query for C3
3         =     B  -Ergebnis ;0, if counter value C3=0; 1, if counter value C3>0

;Word-indirect addressing of counter C3 status:
4         L     W  K0H,B     ;Indirect address of first 16 counter statuses
5         L     W  [B],C     ;Load statuses C0 through C15 into Register C
          PZ:   2
6         U     B  C.3       ;Status bit query for C3
7         =     B  -Ergebnis ;0, if counter value C3=0; 1, if counter value C3>0

          ;Indirect addressing of C3 counter value:
8         L     W  K0606H,A  ;Indirect address of counter value of counter C3
          ;                   even simpler: L W &Z3,A
9         L     W  [A],D     ;Load counter value from C3 to D

          ;Indirect addressed count :
10        L     W  K0606H,D  ;indirect address of counter value of counter C3
          ;                   even simpler: L W &Z3,D
          PZ:   3
11        U     B  -count    ;Counter event
12        ZV       [D]       ;Count up indirect addressed counter C3


          ;Bit-indirect addressing of timer T3 status:
13        L     W  K8083H,A  ;Indirect bit address for status of timer T3
          PZ:   4
14        U     B  [A]       ;Status bit query for T3
15        =     B  -Ergebnis ;0, if timer value of T3=0; 1, if timer value T3>0

          ;Wort-indirect addressing of status of timer T3:
16        L     W  K10H,B    ;Indirect address of first 16 timer statuses
17        L     W  [B],C     ;Load statuses T0 through T15 into Register C
          PZ:   5
18        U     B  C.3       ;Status bit query for T3
19        =     B  -Ergebnis ;0, if counter value C3=0; 1, if counter value C3>0
```

```
     ;Indirect addressing of actual value of T3 :
20   L    W  K0706H,A  ;Indirect address of actual value of timer T3
     ;                  even simpler: L W &T3,A
21   L    W  [A],D     ;Load actual value from T3 to D


     ;Start timer T3 with indirect addressing:
22   L    W  K080FH,B  ;Timer start value = 15 sec, also permitted: L  W  K15.2,B
23   L    W  K0706H,D  ;Indirect address of timer T3
     ;                  even simpler: L W &T3,D
     PZ:  6
24   U    B  -time     ;Timer start event
25   SV      B,[D]     ;Start indirect addressed timer T3 with value from Reg. B


     ;Bit-indirect addressing of input bit I3.4 :
26   L    W  K841CH,A  ;Indirect bit address for input I3.4
     PZ:  7
27   U    B  [A]       ;Query of input bit I3.4
28   =    B  -Ergebnis


     ;Word-indirect addressing of input byte I3:
29   L    W  K0,B
30   L    W  K83H,B    ;Indirect address of input byte I3 :
     ;                  even simpler: L by &E3,B
31   L    BY [B],C     ;Load I3 input byte into Register C
     PZ:  8
32   U    B  C.4       ;Follow up by query of input bit I3.4
33   =    B  -Ergebnis


     ;Bit-indirect setting of output bit O8.0:
34   L    W  K8640H,A  ;Indirect bit address for output O8.0
     PZ:  9
35   U    B  -log1     ;Fixed 1
36   S    B  [A]       ;Set output O8.0
```

```
     ;Copy 10 words from DM4,D50 to DM5,D10 with the aid of two active
     ;data modules, indirect data word addressing, and indirect module calls:
     ;
37   L    W  K4H,A     ;Indirect address for DM4
38   CM      [A]       ;Open data module DM4
39   L    W  K5H,B     ;Indirect address for DM5
40   CX      [B]       ;Open DM5 as 2nd active data module

41   L    W  K10D,A    ;Number of words to be copied

42   L    W  K0832H,C  ;Indirect address for D50 in 1st active DM
43   L    W  K180AH,D  ;Indirect address for D10 in 2nd active DM

     ;                  even simpler: L W &D50,C
     ;                  even simpler : L W &DX10,D
          -Schleife

44   L    W  [C],B     ;Read address in 1st DM
45   T    W  B,[D]     ;Write address in 2nd DM
46   INC  W  C,2       ;Increment address in 1st DM by one word = 2 bytes
47   INC  W  D,2       ;Increment address in 2nd DM by one word = 2 bytes


48   DEC  W  A,1       ;Decrement loop counter
49   JPN
50   PN     -Schleife ;If loop counter not yet=0, continue copying
50   NOP1
51   NOP1



     ;With the use of block commands, there is a much simpler solution
     ;to the preceding copy example:

52   CM      -db4      ;Open source DM
53   CX      -db5      ;Open destination DM
54   L    W  K10D,C    ;Block size always in C
55   BLT  W  D50,DX10  ;Block transfer from D50, 1st active DM, to D10, 2nd active DM
```

9.2 Programming Examples – Handling the 1-ms Timer

```
+-------------------------------------------------------------------------------+
| PLC Documentation      Bosch CL400     Version 4.02    Date:    3. Jan. 1995 |
| Project: CL400/ZS0     Filename: OM1.PCO               Page:        1        |
+-------------------------------------------------------------------------------+

     ;NOTES:    - The specified value may not exceed 10 bits (1023D) in size.
     ;             Larger values cause default to maximum value!
     ;          - When using interrupt triggering, OM42 must be remembered!
     ;
     ;CM     -MS_ZEIT,7
     ;                  +---+
     ;P0  W  -Sollwert ; <   ! Spec'd timer value (end value) 10 bits at 1-ms resolution
     ;P1  B  -Strt/Stp ; <   ! Start timer at "1"  / Stop timer at "0"
     ;P2  B  -Reset    ; <   ! Delete current timer value, reset SM14/15
     ;P3  B  -Rund/Abl ; <   ! Timer wraparound (1) / Elapsing (0)
     ;P4  B  -Int/Flag ; <   ! Trigger interrupt (1) / Set process flag (0)
     ;P5  B  -NfB1     ; <   ! Matrix definition for actual-value update in SM14/15
     ;P6  B  -NfB2     ; <   ! NfB1/NfB2: 00=without / 01=1ms / 10=5ms / 11=10ms
     ;                  +---+


     ; Operand definitions for following examples
     ;-------------------------------------------
     ; CL400 System definitions
     DEF     SM14,-ZeitZust
     DEF     SM15.4,-Interr
     DEF     SM15.5,-rundl
     DEF     SM15.6,-reset
     DEF     SM15.7,-abgel
     DEF     SM30.3,-log0
     DEF     SM31.1,-log1
     DEF     SM31.7,-gleich
     ; User definitions
     DEF     I0,-ParamS
     DEF     I0.0,-Strt/Stp
     DEF     I0.1,-Reset
     DEF     I0.2,-NfB1
     DEF     I0.3,-NfB2
     DEF     I2,-Sollwert
     DEF     M0.0,-FlMerk
     DEF     M0.1,-Start
     DEF     M0.2,-ImpMerk
     DEF     O0,-SM_Abb
     DEF     O2,-Wirkung
     DEF     IO2,-Dauer

     ; Sample selection
1    L    BY -ParamS,A
2    SLR  BY A,5       ; only bits 5,6 and 7 are valid
3    CPLA BY K1,A
4    JPZ     -Beisp1
5    CPLA BY K2,A
6    JPZ     -Beisp2
7    CPL  BY K3,A
8    JPZ     -Beisp3
9    CPL  BY K4,A
10   JPZ     -Beisp4
11   CPL  BY K5,A
12   JPZ     -Beisp5
13   CPL  BY K6,A
14   JPZ     -Beisp6
15   JP      -EM
```

```
+-------------------------------------------------------------------------------+
| PLC Documentation      Bosch CL400     Version 4.02    Date:    3. Jan. 1995 |
| Project: CL400/ZS0     Filename: OB1.PCO               Page:        3        |
+-------------------------------------------------------------------------------+

          -Beisp1

     ; 1. Wraparound timer
     ;*********************

     ; 1.1 With Interrupt
     ;-------------------
     ; Example:  The timer is started  o n c e , on each positive transition
     ;           at the start input, and is then always restarted automatically
     ;           whereby the OM42 interrupt OM (response program)is called once after
     ;           each timer cycle.
     ;           This means that prior to each new timer start (module call), the
     ;           specified value and the update matrix in the assigned SM14/15
     ;           can be newly defined.

     ; Enable interrupt (peripheral interrupt 9 = OM42)
16   L    W  K0100H,A          ; Set default for mask bit 8 (PI9)
17   TM      A,PI              ; Write peripheral interrupt mask
18   EAI     PI                ; Enable peripheral interrupts

     ;Start timer with upon each signal transition on start bit
19   AN   B  -Strt/Stp
20   R    B  -FlMerk
21   A    B  -Strt/Stp
22   AN   B  -FlMerk
23   S    B  -FlMerk
     ;
24   CMC     -MS_ZEIT,7
     ;                   +---+
     P0   W  -Sollwert ; <   ! Spec'd timer value (end value) 10 bits at 1-ms resolution
     P1   B  -Strt/Stp ; <   ! Start timer at "1"  / Stop timer at "0"
     P2   B  -log1     ; <   ! Delete current timer value, reset SM14/15
     P3   B  -log1     ; <   ! Timer wraparound (1) / Timer elapsing (0)
     P4   B  -log1     ; <   ! Trigger interrupt (1) / Set process flag (0)
     P5   B  -NfB1     ; <   ! Matrix definition for actual-value update in SM14/15
     P6   B  -NfB2     ; <   ! NfB1/NfB2: 00=without / 01=1ms / 10=5ms / 11=10ms
     ;                   +---+
     ;
     ; Response program for time-out is stored in OM42 interrupt organization module.

25   JP      -EM
```

```
+------------------------------------------------------------------------------+
| PLC Documentation      Bosch CL400     Version 4.02    Date:    3. Jan. 1995 |
| Project: CL400/ZS0     Filename: OM1.PCO               Page:        3        |
+------------------------------------------------------------------------------+

          -Beisp2

     ; 1.2 Without interrupt, using the 5-bit SM counter (SM15.2 through SM15.6)
     ;-----------------------------------------------------------------------
     ;Example:  Simulation of a timer length of 30s with a basic accuracy of 1 ms.
     ;          The special aspect of this timer is that it is always available for
     ;          current queries, even within the PLC cycle.

     ;Start timer with each signal transition on the start bit
26   AN   B  -Strt/Stp
27   R    B  -FlMerk
28   A    B  -Strt/Stp
29   AN   B  -FlMerk
30   S    B  -FlMerk
     ;
31   CMC      -MS_ZEIT,7
     ;                    +---+
     P0   W  K1000     ; <   ! Spec'd timer value (end value) 10 bits at 1-ms resolution
     P1   B  -Strt/Stp ; <   ! At "1" start timer / At "0" stop timer
     P2   B  -log1     ; <   ! Delete current timer value, reset SM14/15
     P3   B  -log1     ; <   ! Timer wraparound (1) / Elapsing (0)
     P4   B  -log0     ; <   ! Trigger interrupt (1) / Set process flag (0)
     P5   B  -NfB1     ; <   ! Matrix definition for actual-value update in SM14/15
     P6   B  -NfB2     ; <   ! NfB1/NfB2: 00=without / 01=1ms / 10=5ms / 11=10ms
     ;                    +---+
     ;
     ; Interpretation of 5-bit SM counter at any point of the PLC program,
     ; including multiples, possible.
32   L    W  -ZeitZust,A       ; Read timer SM, contents as defined in P5/P6
33   SLL  W  A,1               ; Delete timer end bit
34   SLR  W  A,11              ; 5-bit SM counter into lower-value bits
35   CPLA W  K30,A             ; 30s elapsed?
36   JPCY    -30s_inc          ; No!
37   L    W  K0,A
38   L    W  &-ZeitZust,B
39   T    W  A,[B]             ; Delete SM contents
40   L    W  -Wirkung,A
41   INC  W  A,1
42   T    W  A,-Wirkung

     ; Response program to timer end is programmed here.

          -30s_inc

43   JP       -EM
```

```
+------------------------------------------------------------------------------+
| PLC Documentation      Bosch CL400     Version 4.02    Date:    3. Jan. 1995 |
| Project: CL400/ZS0     Filename: OM1.PCO               Page:        4        |
+------------------------------------------------------------------------------+

          -Beisp3

     ; 2. Elapsing timer
     ;******************

     ; 2.1 With Interrupt
     ;--------------------
     ;Example:  The timer is started  o n c e , on each positive transition
     ;          at the start input, and runs for the specified interval.
     ;          When the interval has elapsed, the OM42 interrupt module is called once,
     ;          and the "response program" is processed.

     ; Enable interrupt (Peripheral interrupt 9 = OM42)
44   L    W  K0100H,A          ; Set default for mask bit 8 (PI9)
45   TM      A,PI              ; Write peripheral interrupt mask
46   EIA     PI                ; Enable peripheral interrupts

     ;Start timer with upon each signal transition on start bit
47   AN   B  -Strt/Stp
48   R    B  -FlMerk
49   A    B  -Strt/Stp
50   AN   B  -FlMerk
51   S    B  -FlMerk
     ;
52   BAB     -MS_ZEIT,7
     ;                   +---+
     P0   W  -Sollwert ; <   ! Spec'd timer value (end value) 10 bits at 1-ms resolution
     P1   B  -Strt/Stp ; <   ! Start timer at "1"  / Stop timer at "0"
     P2   B  -log1     ; <   ! Delete current timer value, reset SM14/15
     P3   B  -log0     ; <   ! Timer wraparound (1) / Timer elapsing (0)
     P4   B  -log1     ; <   ! Trigger interrupt (1) / Set process flag (0)
     P5   B  -NfB1     ; <   ! Matrix definition for actual-value update in SM14/15
     P6   B  -NfB2     ; <   ! NfB1/NfB2: 00=without / 01=1ms / 10=5ms / 11=10ms
     ;                   +---+
     ;
     ; Response program for time-out is stored in OM42 interrupt organization module.

53   SP      -EM
```

```
+-------------------------------------------------------------------------------+
| PLC Documentation      Bosch CL400     Version 4.02    Date:    3. Jan. 1995 |
| Project: CL400/ZS0     Filename: OM1.PCO               Page:        5        |
+-------------------------------------------------------------------------------+

          -Beisp4

     ; 2.2 Using the Timer HALT function
     ;-------------------------------------
     ;Example:  The timer is started with the specified value, and elapses.
     ;          After timer start, depending on programmed functions, the timer decrement
     ;          can be halted by the Start/Stop control bit.
     ;          Timer control: - Cyclical processing of timer control command
     ;                         - HIGH state upon RESET clears SM14/15
     ;                         - Start by 1/0 transition on Reset=Bit14
     ;                         - Timer stop / Timer continue via state 0/1 state
     ;                           on start/stop=Bit15. Here, the timer countdown bit is
     ;                           also linked in order to prevent cyclical restarting.

     ; Timer start
54   A    B  -Strt/Stp
55   AN   B  -abgel
56   =    B  -Start    ; Timer Start (1) / Stop (0)

57   CM      -MS_ZEIT,7
     ;                  +---+
     P0   W  -Sollwert ; <   ! Spec'd timer value (end value) 10 bits at 1-ms resolution
     P1   B  -Start    ; <   ! Start timer at "1"  / Stop timer at "0"
     P2   B  -Reset    ; <   ! Delete current timer value, reset SM14/15
     P3   B  -log0     ; <   ! Timer wraparound (1) / Timer elapsing (0)
     P4   B  -log0     ; <   ! Trigger interrupt (1) / Set process flag (0)
     P5   B  -NfB1     ; <   ! Matrix definition for actual-value update in SM14/15
     P6   B  -NfB2     ; <   ! NfB1/NfB2: 00=without / 01=1ms / 10=5ms / 11=10ms
     ;                  +---+
     ;
     ;Response to timer status can be programmed at any point of the PLC program.
58   L    W  -ZeitZust,A       ; Read timer SM, contents as defined in P5/P6

59   JP      -EM
```

```
+------------------------------------------------------------------------------+
| PLC Documentation      Bosch CL400     Version 4.02    Date:    3. Jan. 1995 |
| Project: CL400/ZS0     Filename: OM1.PCO               Page:        6        |
+------------------------------------------------------------------------------+

          -Beisp5

     ; 2.3 Timer-accurate program loop (0.1 ms accuracy)
     ;-----------------------------------------------------
     ;Example:  Measured values (readings) are to be captured during a 10 ms interval.
     ;          The timer is started on each positive transition at the start input,
     ;          and then runs for the specified interval.

     ;Start timer with each signal transition on the start bit
60   AN   B  -Strt/Stp
61   R    B  -FlMerk
62   U    B  -Strt/Stp
63   UN   B  -FlMerk
64   S    B  -FlMerk
65   =    B  -ImpMerk  ; Pulse marker for measuring loop, see below.
     ;
66   BAB     -MS_ZEIT,7
     ;                  +---+
     P0   W  K20       ; <   ! Spec'd timer value (end value) 10 bits at 1-ms resolution
     P1   B  -Strt/Stp ; <   ! Start timer at "1"  / Stop timer at "0"
     P2   B  -log0     ; <   ! Delete current timer value, reset SM14/15
     P3   B  -log0     ; <   ! Timer wraparound (1) / Timer elapsing (0)
     P4   B  -log0     ; <   ! Trigger interrupt (1) / Set process flag (0)
     P5   B  -log0     ; <   ! Matrix definition for actual-value update in SM14/15
     P6   B  -log0     ; <   ! NfB1/NfB2: 00=without / 01=1ms / 10=5ms / 11=10ms
     ;                  +---+

     ; Program loop for measured-value capture

67   AN   B  -ImpMerk  ; Pulse marker for measuring loop, formation, see above.
68   SPB     -noMess   ; Process measuring loop during one PLC cycle only!

          -Messchl

     ; >>>>>>>>>>> The measured-value capture program is processed here >>>>>>>>>>
     ;    :
     ;    :
     ; >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

69   RMST    A         ; Timer read command, 0.1 ms resolution (accuracy)
70   CPLA W  K100,A    ; 10ms elapsed?
71   JPCY    -Messchl  ; No --> continue loop processing

     ; Read and display the measuring loop duration
     ; (Observe output "delay", programming unit communications!)
72   T    W  A,-Dauer  ; Timer loop duration

          -noMess

73   JP      -EM
```

```
+------------------------------------------------------------------------------+
| PLC Documentation      Bosch CL400     Version 4.02    Date:    3. Jan. 1995 |
| Project: CL400/ZS0     Filename: OM1.PCO               Page:        7        |
+------------------------------------------------------------------------------+

          -Beisp6

     ; 2.4 Cycle time measuring in 0.1 ms resolution (max. deviation 0.1 ms)
     ;-----------------------------------------------------------------------
     ;Example:  The timer is cyclically started, with a >> cycle time value immediately
     ;          prior to program end, and does not elapse.
     ;          In the respective successive cycle, the elapsed value is read
     ;          prior to the new timer start.

     ; Timer loop which simulates a PLC program
74   L    W  K1000,A
          -loop
75   DEC  W  A,1
76   JPN      -loop

     ; Read and display the duration of the last program cycle
     ; (Observe output "delay", programming unit communications!)
77   RMST     B          ; Timer read command, 0.1 ms accuracy
78   T    W   B,-Dauer   ; Measured cycle time

79   BA       -MS_ZEIT,7
     ;                    +---+
     P0   W   K1000      ; <   ! Spec'd timer value (end value) 10 bits at 1-ms resolution
     P1   B   -log1      ; <   ! Start timer at "1"  / Stop timer at "0"
     P2   B   -log1      ; <   ! Delete current timer value, reset SM14/15
     P3   B   -log0      ; <   ! Timer wraparound (1) / Timer elapsing (0)
     P4   B   -log0      ; <   ! Trigger interrupt (1) / Set process flag (0)
     P5   B   -log0      ; <   ! Matrix definition for actual-value update in SM14/15
     P6   B   -log0      ; <   ! NfB1/NfB2: 00=without / 01=1ms / 10=5ms / 11=10ms
     ;                    +---+

          -BE

     ; For verification only
     ; ---------------------

     ; Read and display timer SM, contents as per update definition
80   L    W   -ZeitZust,A
81   T    W   A,-SM_Abb

     ; Read output word which is incremented in OM42 interrupt module.
82   L    W   -Wirkung,A

83   EP
```

```
+-------------------------------------------------------------------------------+
| PLC Documentation      Bosch CL400     Version 4.02    Date:    3. Jan. 1995 |
| Project: CL400/ZS0     Filename: MS_ZEIT.PCO           Page:        8        |
+-------------------------------------------------------------------------------+

     ==========
     1-ms Timer
     ==========
     ;NOTES:    - The specified value may not exceed 10 bits (1023D) in size.
     ;            Larger values cause default to maximum value!
     ;          - When using interrupt triggering, OM42 must be remembered!

+-------------------------------------------------------------------------------+
|Para.|RES.|Ind|   Symbol   |< >|Comment                      | Version: 1.0     |
|-----+----+---+------------+---+-----------------------------------------------|
| P0  | W  |   | Sollwert   | < |Spec'd timer (end) value 10 bits in 1-ms matrix |
| P1  | B  |   | Strt/Stp   | < |Start timer at "1"  / Stop timer at "0"         |
| P2  | B  |   | Reset      | < |Delete current timer value, reset SM14/15       |
| P3  | B  |   | Rund/Abl   | < |Timer wraparound (1) / Elapsing (0)              |
| P4  | B  |   | Int/Flag   | < |Trigger interrupt (1) / Set process flag (0)     |
| P5  | B  |   | NfB1       | < |Matrix defin. for actual-value update in SM14/15|
| P6  | B  |   | NfB2       | < |NfB1/NfB 2:00=without/ 01=1ms / 10=5ms / 11=10ms|
+-------------------------------------------------------------------------------+



     ; Load timer specified value (bits 0-9 of control word)
1    L    W  -Sollwert,A
2    CPLA W  K1023,A    ; Maximum value exceeded?
3    JPCY    -WertiO    ; Lesser value OK
4    L    W  K1023,A    ; Set maximum value default
          -WertiO

5    U    B  -Strt/Stp
6    =OM  B  A.15       ; Timer Start (1) / Timer Stop (0)

7    U    B  -Reset
8    =OM  B  A.14       ; Delete current timer value (SM bytes 14+15) (1)

9    U    B  -Rund/Abl
10   =OM  B  A.13       ; Timer elapses (0) / Timer restarts (1)

11   U    B  -Int/Flag
12   =OM  B  A.12       ; Upon time-out, process SM35.7 = 1 (0) / OM42 interrupt module (1)

     ; Timer control (bits 10-15 of control word)
13   U    B  -NfB1
14   =OM  B  A.10       ; Updating current value in SM14/15, bits 0-9.
15   U    B  -NfB2
16   =OM  B  A.11       ; 00=without / 01=1ms / 10=5ms / 11=10ms

17   SMST    A          ; Timer control command (not transition-controlled)

18   EM


+-------------------------------------------------------------------------------+
| PLC Documentation      Bosch CL400     Version 4.02    Date:    3. Jan. 1995 |
| Project: CL400/ZS0     Filename: 1MS_INT.PCO           Page:        9        |
+-------------------------------------------------------------------------------+

     ; Peripheral interrupt 9, invoked by time-out of 1-ms timer

     ; Example: Output word O2 is incremented in the specified time matrix.
1    L    W  A2,A
2   INC   W  A,1
3    T    W  A,A2

4   EM
```

9.3 Sample Applications of Block Commands

```
 1 | PBBLT - Sample module for block transfer


2    CM      -DM10
3    CX      -DM9

     ; Example 1 : all address specifications are direct:
     ; Copy data block of 50-word size from DM10, from Byte20 in DM9 from Byte40 :
     ; Size specification compulsory in Reg. C

4    L    W  K50,C
5    BLT  W  D20,DX40


     ; Example 2 : all address specifications are indirect:
     ; Copy data block of 5-byte length from DM10, from Byte100 in DM9 from Byte150 :
     ; Size specification compulsory in Reg. C

6    L    W  K5,C
7    L    W  K0864H,B  ; DM, byte offset 100
8    L    W  K1896H,A  ; DX, byte offset 150
9    BLT  BY [B],[A]   ; fixed register assignment for source and destination!


10   EM
```

```
 1 | PBVVX - Sample module comparing two block contents

; In this example, a verification is made whether all elements differ
; between blocks. (CFZ)

2    CM      -DM11
3    CX      -DM13

; Example: Compare block in 1st active DM with block in 2nd active DM:

; Destination block address, always from Register A:
4    L    W  K1800H,A  ; 2nd active DM, byte offset 0

; Example of block size, here in Register C:
5    L    W  K256D,C

; Example of source block address, here in Reg. B: 1st active DM, byte offset 0
6    L    W  K800H,B

; Block compare forward for congruence (being equal):
7    CFZ  W  [B],C
; The compare result used as a stop criterion refers to the arithmetical operation:
; Value from destination block minus value from source block.
; In the event that compare criterion is met as a result of this operation, then
; Cancel.
; Registers A and B contain the addresses of the first equal values!
8    L    W  A,A       ;Monitor assistance
9    L    W  B,B       ;Monitor assistance

; Register C contains the number of characters that have not yet been searched:
10   L    W  C,C       ;Monitor assistance

; Failure to meet the compare condition will cause the zero flag SM31.7 to be set.
; In this case, if no two identical characters were found in the entire block,
; indicating that the blocks are completely different!
; In general: ZERO, if the complete block was searched without
; the compare condition having been met at any point.

11   A    B  SM31.7
12   =    B  M255.7    ;Monitor assistance


13   EM
```

```
 1 | PBVRX - Sample module comparing two block contents

; This example is used to check whether all elements are identical
; between blocks. (CBN)

2    CM      -DM11
3    CX      -DM13

; Example: Compare block in 1st active DM with block in 2nd active DM:

; Destination block address, always from Register A:
4    L    W  K19FEH,A  ; 2nd active DM, byte offset 510

; Example of block size, here in Register C:
5    L    W  K256D,C

; Example of source block address, here in Reg. B: 1st active DM, byte offset 510
6    L    W  K09FEH,B

; Block compare backward for incongruence (being unequal):
7    CBN  W  [B],C
; The compare result used as a stop criterion refers to the arithmetical operation:
; Value from destination block minus value from source block.
; In the event that compare criterion is met as a result of this operation, then
; Cancel.
; Registers A and B contain the addresses of the first unequal values!
8    L    W  A,A       ;Monitor assistance
9    L    W  B,B       ;Monitor assistance

; Register C contains the number of characters that have not yet been searched:
10   L    W  C,C       ;Monitor assistance

; Failure to meet the compare condition will cause the zero flag SM31.7 to be set.
; In this case, if no two unidentical characters were found in the entire block,
; indicating that the blocks are completely identical!
; In general: ZERO, if the complete block was searched without
; the compare condition having been met at any point.

11   A    B  SM31.7
12   =    B  M255.7    ;Monitor assistance


13   EM
```

```
 1 | PBSVX - Sample module searching forward for a character in a block

2    CM      -DM11

     ; Block start must always be defined in Register A:
     ; here, 1st active DM from byte offset 10:

3    L    W  &D10,A

     ; For a block size of 100 bytes, search for the character that is log. greater
     ; than 35d:
     ; Compare value and size directly defined as a constant:
4    SFLG BY K35D,K100D

     ; The address of the character found is now in Register A:
     ; If no character is found, the address in Register A will point behind the
     ; data block to be searched.
5    L    W  A,A       ; Monitor assistance

     ; Failure to meet the compare condition will cause the zero flag SM31.7 to be set.
     ; In this case, this means that no character was found.
6    A    B  SM31.7
7    =    B  M255.7    ; Monitor assistance




 2 | 2. Example of searching forward for a character in a block

9    CM      -DM11

     ; Block start must always be defined in Register A:
     ; here, 1st active DM from byte offset 10:

10   L    W  &D10,A

     ; For a block size of 100 bytes, search for the character 35D:
     ; Compare value and size directly defined as a constant:
11   L    W  K35D,B    ; Compare value
12   L    W  K100D,C   ; Block size to be searched
13   SFZ  BY B,C       ; Direct compare value only in Register B
                       ; Direct size information only in Register C
                       ; Immediate and direct specifications can be combined here!

     ; The address of the character found is now in Register A:
     ; If no character is found, the address in Register A will point behind the
     ; data block to be searched.
14   L    W  A,A       ;Monitor assistance

     ; Register C contains the number of characters that have not yet been searched:
15   L    W  C,C

     ; Failure to meet the compare condition will cause the zero flag SM31.7 to be set.
16   A    B  SM31.7
17   =    B  M255.7    ;Monitor assistance

18   EM
```

```
 1 | PBSRX - Sample module searching backward for a character in a block


2    CM      -DM11

     ; Block start must always be defined in Register A:
     ; here, 1st active DM from byte offset 511:

3    L    W  &D511,A

     ; For a block size of 512 bytes, search for the character that is log. less
     ; than 35d:
     ; Compare value and size directly defined as a constant:
4    SRC  BY K35D,K512D

     ; The address of the character found is now in Register A:
     ; If no character is found, the address in Register A will point behind the
     ; data block to be searched.
5    L    W  A,A       ; Monitor assistance

     ; Failure to meet the compare condition will cause the zero flag SM31.7 to be set.
     ; In this case, this means that no character was found.
6    A    B  SM31.7
7    =    B  M255.7    ; Monitor assistance




 2 | 2. Example of searching backward for a character in a block

9    CM      -DM11

     ; Block start must always be defined in Register A:
     ; here, 1st active DM from byte offset 511:

10   L    W  &D511,A

     ; For a block size of 512 bytes, search for the character 35D:
     ; Compare value and size directly defined as a constant:
11   L    W  K35D,B    ; Compare value
12   L    W  K512D,C   ; Block size to be searched
13   SRZ  BY B,C       ; Direct compare value only in Register B
                       ; Direct size information only in Register C
                       ; Immediate and direct specifications can be combined here!

     ; The address of the character found is now in Register A:
     ; If no character is found, the address in Register A will point behind the
     ; data block to be searched.
14   L    W  A,A       ;Monitor assistance

     ;Register C contains the number of characters that have not yet been searched:
15   L    W  C,C

     ; Failure to meet the compare condition will cause the zero flag SM31.7 to be set.
16   A    B  SM31.7
17   =    B  M255.7    ;Monitor assistance

18   EM
```

9.4 Setting the System Clock

⇒ **When setting the system clock, no crosschecks with the calendar month are made for the "Day" entry. As a result, incorrect entries may occur (e.g., April 31). Neither is the "Day of the Week" referenced to the date. Therefore, when setting the system clock, the user is required to ensure that all data information has been entered correctly.**

In CL350, CL400, and CL500 controllers, the system clock is set via the TFD system command (refer also to Section 8.17, "System Commands"). In this procedure, it must be ensured that the command setting the clock is issued only once (pulse interpretation of the select criterion) for the sake of simplification, the trigger pulse special marker "RI_Anl" is used here.

The sample program listed below was generated by the WinSPS utility software. The command mnemonics encountered with the use of PROFI PG software differs only with regard to the PLC label , and in the operand extension (BY instead of B).

```
AN        -RI_Anl    ; Trigger pulse upon each PLC startup
JPC       1Anl       ; Set clock each time upon Power-On.

; Reset clock (13:15:00 28.7.99 Wednesday)
L     B    0,A
T     B    A,M10      ;Second
L     B    15,A
T     B    A,M11      ;Minute
L     B    13,A
T     B    A,M12      ;Hour
L     B    28,A
T     B    A,M13      ;Day
L     B    7,A
T     B    A,M14      ;Month
L     B    99,A
T     B    A,M15      ;Year
L     B    3,A
T     B    A,M16      ;Day of the Week (0=Sun, 1=Mon .... 6=Sat)

;Setting date/time
TFD   B    10         ; Transfer field direct

; PS0: Acknowledgement word address (HB: error code, LB: free user code, see PS2)
PS0   W    &M8        ; The acknowledgement message is returned in marker M8
; PS1: System interrupt/User code
PS1   W    16#FFAA    ; No system interrupt (FF), and the free user code (AA)
; PS2: Coordination
PS2   W    16#FFFF    ; No coordination point, no coordination marker
; PS3: Define time (source data)
PS3   W    &M10       ; Basic source data address
; PS4: DB-Nr.
PS4   W    0          ; Not relevant because no DM is in use
; PS5: Number of bytes to be transferred
PS5   W    7
; PS6: ZS400 block address
PS6   W    16#FF
; PS7: Field type
PS7   W    16#0010    ; ... for date / time
; PS8: Field index
PS8   W    0          ; Not relevant, otherwise field index
; PS9: Address offset of operand defined under field type
PS9   W    0          ; Not relevant, because destination is defined
1Anl:
```

1070 072 127-112 (00.02) GB • HB SP • AT/PLS • Printed in Germany