

Typ3 osa

Software PLC Programming manual



Edition

102



BOSCH
Automation

Typ3 osa

Software PLC Programming manual

1070 073 792-102 (00.03) GB



© 1998-2000

by Robert Bosch GmbH, Erbach / Germany
All rights reserved, including applications for protective rights.
Reproduction or distribution by any means subject to our prior written permission.

Discretionary charge 10.00 DM

Contents

	Page
1	Safety Instructions 1–1
1.1	Proper use 1–1
1.2	Qualified personnel 1–3
1.3	Safety markings on products 1–4
1.4	Safety instructions in this manual 1–5
1.5	Safety instructions concerning the product described 1–6
1.6	Documentation, software release and trademarks 1–8
2	Overview 2–1
2.1	Development environment 2–1
2.2	Architecture of the software PLC 2–1
2.3	Kns_AnwenderInit 2–3
2.4	Kns_AnwenderMain 2–3
3	PLC interface 3–1
3.1	NC interface: Inputs 3–2
3.1.1	Byte number table, NC inputs 3–2
3.1.2	Byte number table, machine (Profibus DP) 3–3
3.1.3	Byte number table, future NC inputs 3–3
3.1.4	Overview of axis-related interface signals, PLC → NC 3–4
3.1.5	Overview of spindle-related interface signals PLC → NC 3–8
3.1.6	Overview of channel-related interface signals PLC → NC 3–12
3.2	NC interface: Outputs 3–16
3.2.1	Byte number table, NC outputs 3–16
3.2.2	Byte number table, machine (Profibus DP) 3–17
3.2.3	Byte number table, future NC outputs 3–17
3.2.4	Overview of axis-related interface signals, NC → PLC 3–18
3.2.5	Overview of spindle-related interface signals NC → PLC 3–20
3.2.6	Overview of channel-related interface signals NC → PLC 3–22
3.3	Global interface 3–25
3.3.1	Overview of inputs of global interface 3–25
3.3.2	Overview of outputs of global interface 3–26
3.4	Machine interface 3–27
3.4.1	Signal assignment area 2: Inputs 3–28
3.4.2	Signal assignment area 2: Outputs 3–29

4	Signal accessing	4-1
4.1	Functions for accessing the NC interface	4-2
4.1.1	Function for accessing the input image	4-2
4.1.2	Functions for accessing the output image	4-2
4.2	Functions for accessing markers (flags)	4-4
4.2.1	Access to non-existent markers (flags)	4-5
4.3	Functions for accessing data modules	4-6
4.3.1	General functions for accessing data modules	4-6
4.3.2	Functions for read and write access on a data module	4-8
4.4	Functions for accessing timer modules	4-9
4.4.1	Timer module types	4-9
4.4.2	Timer functions	4-11
4.5	Functions for accessing counters	4-13
4.6	Analog I/O	4-14
4.7	Access to the extended interface	4-15
4.7.1	Read and write access functions to the extended interface ...	4-15
4.8	Accessing actual axis values	4-16
4.9	Access to the global interface	4-17
4.9.1	Read and write access functions to the global interface	4-17
5	Auxiliary functions	5-1
6	Machine error and status display (MSD)	6-1
7	Ncs functions	7-1
8	Diagnostics	8-1
8.1	Watchdog function	8-1
8.2	PROFIBUS DP errors and warnings	8-2
A	Appendix	A-1
A.1	Index	A-1

1 Safety Instructions

Please read this manual before commissioning the Typ3 osa. Store this documentation in a place to which all users have access at all times

1.1 Proper use

This manual contains information required for the proper use of the product. For reasons of clarity, however, it cannot contain all details about all possible combinations of functions. Likewise, it is impossible to consider every conceivable case of integration or operation.


The Typ3 osa is used to

- activate feed drives, spindles and auxiliary axes of a machine tool via SERCOS interface for the purpose of guiding a processing tool along a programmed path to machine a workpiece (CNC). Furthermore, a PLC is required with appropriate I/O components which – in communication with the actual CNC – controls the machine processing cycles holistically and acts as a technical safety monitor.
- program contours and the machining technology (path feedrate, spindle speed, tool change) of a workpiece.

Any other application is deemed improper use!

The products described

- have been developed, manufactured, tested and documented in compliance with the safety standards. These products pose no danger to persons or property if they are used in accordance with the handling regulations and safety notes prescribed for their configuration, installation and proper operation.
- comply with the requirements of
 - the EMC Directives (89/336/EEC, 93/68/EEC and 93/44/EEC)
 - the Low-Voltage Directive (73/23/EEC)
 - the harmonized standards EN 50081-2 and EN 50082-2
- are designed for operation in industrial environments (emission class A), i.e.
 - no direct connection to public low-voltage power supply,
 - connection to the medium- or high-voltage system via a transformer.In residential environments, in trade and commerce as well as small enterprises class A equipment may only be used if it does not inadmissibly interfere with other equipment.

 **This is a class A device which may cause radio interference in residential environments. In this case, the operator may be required to take suitable countermeasures and to bear the cost of the same.**

The faultless, safe functioning of the product presupposes proper transport, storage, erection and installation as well as careful operation.

**CAUTION**

This product was examined by us for "Year 2000 Compliance" using our operating, programming, visualization and control software, and passed the relevant tests.

We should like to point out that the "Year 2000 Compliance" may be lost if additional software packages are installed, for which the customer is responsible.

Examples of a possible loss of "Year 2000 Compliance" known to us include:

1. If other software packages are installed, parts of the software supplied by us may be overwritten (e.g. DLL's).
 2. The additionally installed software reads the date directly from the BIOS or even the date/time module. In order to correct the date after the millennium change, it will be sufficient to restart the unit.
 3. Some components of Microsoft products (e.g. the File Manager and the Explorer) are limited as to "Year 2000 Compliance". For more information, please refer to the Microsoft web pages at <http://www.eu.microsoft.com/ithome/topics/year2k/product/product.htm> .
-

1.2 Qualified personnel

The requirements as to qualified personnel depend on the qualification profiles described by ZVEI (central association of the electrical industry) and VDMA (association of German machine and plant builders) in:

Weiterbildung in der Automatisierungstechnik

edited by: ZVEI and VDMA

MaschinenbauVerlag

Postfach 71 08 64

D-60498 Frankfurt.

The present manual is designed for **project engineers and NC specialists.**

These persons need special knowledge of

- programming in C
- the safety standards in electrical and automation engineering
- programming in accordance with the DIN regulations and operating CNC machines.

Programming, start and operation as well as the modification of programs or program parameters may only be performed by properly trained personnel! This personnel must be able to judge potential hazards arising from programming, program changes and in general from the mechanical, electrical, or electronic equipment.

Interventions in the hardware and software of our products, unless described otherwise in this manual, are reserved to our specialized personnel.

Tampering with the hardware or software, ignoring warning signs attached to the components, or non-compliance with the warning notes given in this manual may result in serious bodily injury or material damage.

Only electrotechnicians as recognized under IEV 826-09-01 (modified) who are familiar with the contents of this manual may install and service the products described.

Such personnel are

- those who, being well trained and experienced in their field and familiar with the relevant standards, are able to analyze the work to be carried out and recognize any hazards.
- those who have acquired the same amount of expert knowledge through years of experience that would normally be acquired through formal technical training.

Please note our comprehensive range of training courses. Our training center will be pleased to provide you with further information, telephone: +49 (0) 6062 78-258.

1.3 Safety markings on products



Warning of dangerous electrical voltage!



Warning of danger caused by batteries!



Components sensitive to electrostatic discharge!



Warning of hazardous light emissions (optical fiber cable emissions)



Disconnect from mains before opening!



Pin for connecting PE conductor only!



Connection of shield conductor only

1.4 Safety instructions in this manual



DANGEROUS ELECTRICAL VOLTAGE

This symbol is used to warn of a **dangerous electrical voltage**. The failure to observe the instructions in this manual in whole or in part may result in **personal injuries**.




DANGER

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



CAUTION

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

 This symbol is used to draw the user's attention to special circumstances.

★ This symbol is used if user activities are required.

1.5 Safety instructions concerning the product described

**DANGER**

Danger of life through inadequate EMERGENCY-STOP devices!
EMERGENCY-STOP devices must be active and within reach in all system modes. Releasing an EMERGENCY-STOP device must not result in an uncontrolled restart of the system!
First check the EMERGENCY-STOP circuit, then switch the system on!

**DANGER**

Incorrect or undesired axis movement!
First, new programs should be tested carefully without axis movement! For this purpose, the control unit offers the possibility of inhibiting axis movements and/or auxiliary function outputs by appropriate softkeys in the 'automatic' group operating mode.

**DANGER**

Incorrect or undesired control unit response!
Bosch accepts no liability for damage resulting from the execution of an NC program, an individual NC block or the manual movement of axes!

Furthermore, Bosch accepts no liability for consequential damage which could have been avoided by programming the PLC appropriately!

**DANGER**

Retrofits or modifications may adversely affect the safety of the products described!
The consequences may include severe injuries, damage to equipment, or environmental hazards. Possible retrofits or modifications to the system using third-party equipment therefore have to be approved by Bosch.

**DANGEROUS ELECTRICAL VOLTAGE**

Unless described otherwise, maintenance works must be performed on inactive systems! The system must be protected against unauthorized or accidental reclosing.

Measuring or test activities on the live system are reserved to qualified electrical personnel!

**DANGER****Tool or axis movements!**

Feed and spindle motors generate very powerful mechanical forces and can accelerate very quickly due to their high dynamics.

- Always stay outside the danger area of the machine when it is running!
 - Do not ever deactivate the safety-relevant functions of the unit!
 - Report any malfunction of the unit to your servicing and repairs department immediately!
-

**CAUTION**

Use only spare parts approved by Bosch!

**CAUTION****Danger to the module!**

All ESD protection measures must be observed when using the module! Prevent electrostatic discharges!

The following protective measures must be observed for modules and components sensitive to electrostatic discharge (ESD)!

- Personnel responsible for storage, transport, and handling must have training in ESD protection.
- ESD-sensitive components must be stored and transported in the prescribed protective packaging.
- ESD-sensitive components may only be handled at special ESD-workplaces.
- Personnel, working surfaces, as well as all equipment and tools which may come into contact with ESD-sensitive components must have the same potential (e.g. by grounding).
- Wear an approved grounding bracelet. The grounding bracelet must be connected with the working surface through a cable with an integrated 1 M Ω resistor.
- ESD-sensitive components may by no means come into contact with chargeable objects, including most plastic materials.
- When ESD-sensitive components are installed in or removed from equipment, the equipment must be de-energized.

1.6 Documentation, software release and trademarks

Documentation

The present manual provides information on the installation and handling of the development environment of the software PLC

Overview of available documentation	Part no.	
	German	English
Interface conditions for project engineering and maintenance	1070 073 704	1070 073 736
Operating instructions Standard operator interface	1070 073 726	1070 073 739
Operating instructions – Diagnostics Tools	1070 073 779	1070 073 780
DIN programming instructions for programming to DIN 66025	1070 073 725	1070 073 738
CPL programming instructions	1070 073 727	1070 073 740
ICL700 system description, Program structure of the integrated PLC	1070 073 706	1070 073 737
ICL700 project planning manual, software interfaces and CNC interface signals of the integrated PLC	1070 073 728	1070 073 741
MACODA Operation and configuration of the machine parameters	1070 073 705	1070 073 742
Tool Management – Parameterization	1070 073 782	1070 073 793
Software PLC Development environment for Windows NT	1070 073 783	1070 073 792
Measuring cycles for touch-trigger switching probes	1070 073 788	1070 073 789
Universal Milling Cycles	–	1070 073 795


 **In this manual the floppy disk drive is always drive 'A' and the hard disk drive is always drive 'C'.**


Special keys or combinations of keys are represented by pointed brackets:

- Special keys: e.g. <enter>, <pgup>,
- Key combinations (pressed simultaneously): e.g. <ctrl> + <pgup>

Release

 **This manual refers to the following version:
Software: V 5.1.x**

 **The current release number of the individual software modules can be viewed by selecting the 'Control-Diagnostics' softkey in the 'Diagnostics' group operating mode.**

-  **The software version of Windows95 or WindowsNT may be displayed as follows:**
- 1. Click with right mouse key on the "My Computer" icon on your desktop**
 - 2. Select menu item "Properties".**

Trademarks

All trademarks of software installed on Bosch products upon delivery are the property of the respective manufacturer.

Upon delivery, all installed software is copyright-protected. The software may only be reproduced with the approval of Bosch or in accordance with the license agreement of the respective manufacturer.

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

PROFIBUS® is a registered trademark of PROFIBUS Nutzerorganisation e. V.

SERCOS interface® is a registered trademark of Interessengemeinschaft SERCOS interface e.V.

2 Overview

This programming manual contains information on the functions provided by the Typ3 osa **software PLC**.

If you wish to configure the software PLC of the Typ3 osa control system so as to meet your specific requirements, you need to know how to install and handle the "Development environment for Windows NT".

2.1 Development environment

Installation and handling of the development environment of the software PLC

You need a PC with WindowsNT as an operating system for this development environment, which consists of the following components:

- the "Bosch development environment for WindowsNT" and
- the GNU tools (compiler, debugger, etc.)

The related user and programming documentation covers

- the development environment for WindowsNT
- the GNU development tools.

This user and programming documentation is available from Bosch on request. It is also distributed in our user training programs on the development environment. For more information, please contact your Bosch customer service consultant.

2.2 Architecture of the software PLC

The software PLC is an integrated component of the Typ3 osa control. It runs in the form of software subsystems in the overall Typ3 osa system.

These subsystems are called:

- APS
- KNS

APS subsystem

The **APS** subsystem is the actual PLC. It is a software package developed by Bosch.

The APS has been designed to make data exchange with the Typ3 osa subsystems easier for PLC project planners.

It includes:

- communication routines for interface accessing
- timers
- server functions for the Typ3 osa logic analyser and screen

The APS transfers NC input signals, MSD signals, and logic analyser signals to the corresponding function areas in the Typ3 osa control.



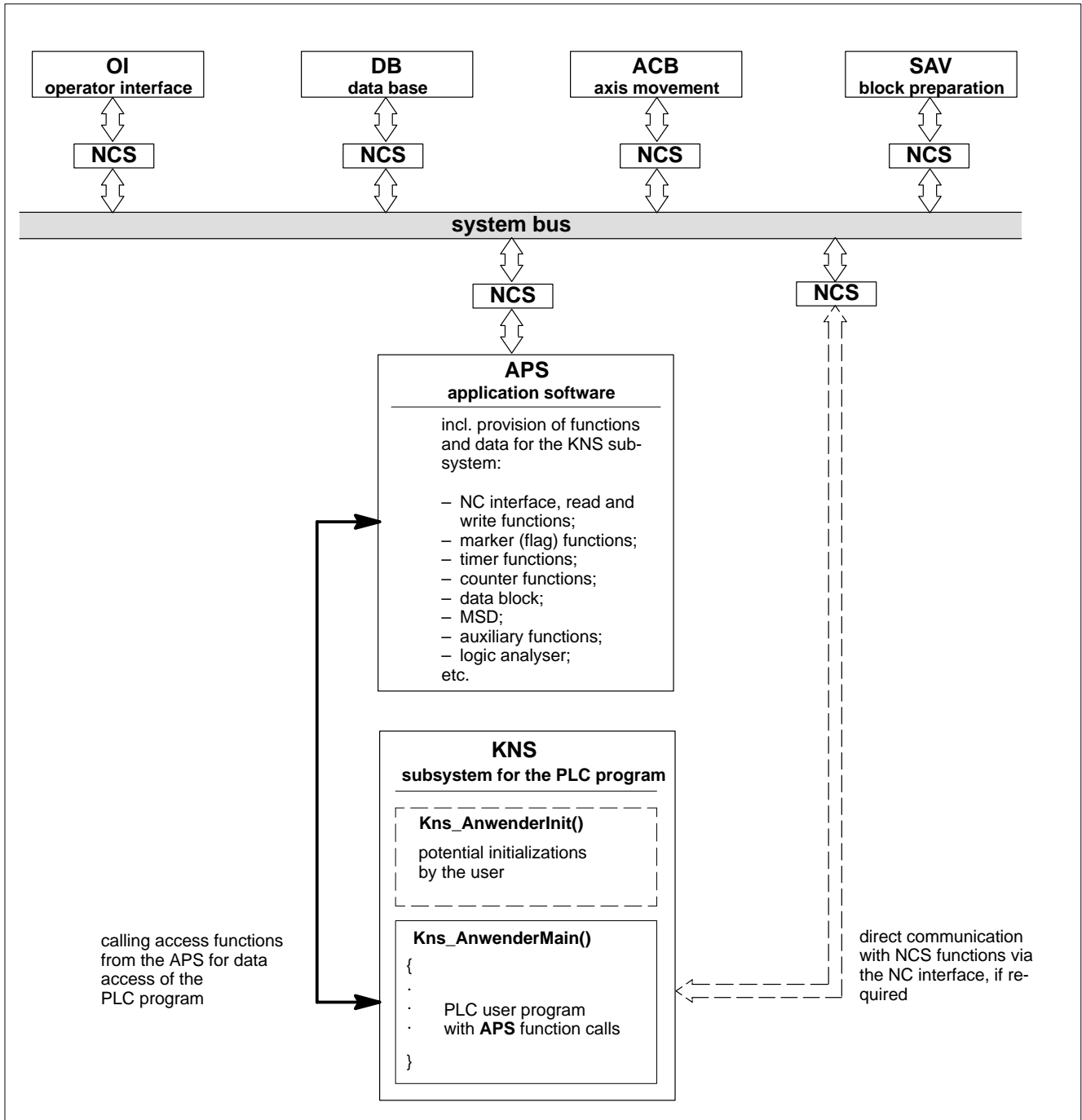
For more information, see the APS description in Project Planning Manual ICL 700.

KNS subsystem

The **software PLC** written by the user is installed in the **KNS** subsystem. The KNS subsystem exists as a file called **kns_main.c**. This file also contains the two main functions **Kns_AnwenderInit** and **Kns_AnwenderMain** in which the user can integrate his PLC program.

The following description of the software PLC refers to those elements of the software PLC that can be programmed in the user PLC program of the current software version.

Software PLC communication structure and interfaces



☞ See also information on the NC interface (NCS) in Project Planning Manual ICL 700.

2.3 Kns_AnwenderInit

All initialization routines created by the user are stored in the **Kns_AnwenderInit** function.

Function call

The **Kns_AnwenderInit** function is called just once, when the Typ3 osa is started before the actual PLC task is called.

Syntax

```
Kns_AnwenderInit(void)
{
    /* Here, the user can enter his initialization function,
       if created*/
    AnwenderInit1(); /*e.g. marker (flag) initializations
    AnwenderInit2(); /*e.g. creation of user-specific data modules
    .
    .
    .
}
```

 **This function corresponds to the former organization module OM27.**

2.4 Kns_AnwenderMain

The **Kns_AnwenderMain** function contains the PLC program written by the user. This function is called cyclically.

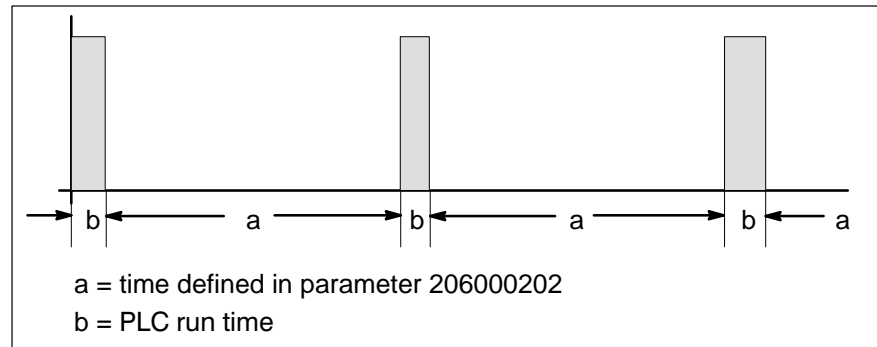
Cycle control

Cycle control is the most widely used method of running a PLC program. No special measures need to be taken for developing a PLC program. The user must only try to keep the program as short as possible to minimize its run time.

Activating the cycle control

The cycle control is activated by setting the machine parameter 206000201 (1 = cyclic). Additionally, parameter 206000202 is used to define the time interval after which the PLC is called up.

It may vary between 10 and 500 ms. This time interval must have elapsed after **EP** before the PLC is reactivated. The time the PLC needs to complete one cycle should be approx. 10% of the value of parameter 206000202. If the PLC runtime is longer than that, this will be at the expense of the overall system.



The intervals (a) between two PLC call-ups (b) are always the same, their length depends on the time set in machine parameter 206000202.

Example:

```

Kns_AnwenderMain()
{
.
.
Aps_EBit_R(p1,p2)
/* read inputs, statuses, links
and write them to outputs
.
.
Aps_EBit_W(p1,p2,p3)
/* define outputs as desired
.
.
Aps_Erzeuge_DB(*p1,p2,p3,p4)
/* create data module
.
.
Eigene Funktion ()
/* e.g. machine functions
.
.
}

```

} PLC program

3 PLC interface

The PLC interface includes the data areas:

- **the NC interface: inputs** for axes, spindles and channels
- **the NC interface: outputs** for axes, spindles and channels
- **the global interface**
- **the interface: machine**

The NC interface provides communication with the NC and includes:

- 16 axis data areas with 8 bytes for inputs and 8 bytes for outputs, ea.
- 8 spindle data areas with 8 bytes for inputs and 8 bytes for outputs, ea.
- 14 channel data areas with 8 bytes for inputs and 12 bytes for outputs, ea.

The individual data areas are distinguished by different byte numbers. As regards **interface access functions**, no distinction is made between axis-, spindle-, or channel-related functions.

For the byte numbers of the individual data areas, see the tables under 3.1 and 3.2.

Channel 0 is an internal channel not available to the user.

The interface is accessed through functions available at the subsystem interface. Accessing functions are available both for inputs and for outputs.

 **For a description of the signals listed in the tables under 3.1 and 3.2 ff., please see the Typ3 Project Planning Manual ICL 700.**

3.1 NC interface: Inputs

3.1.1 Byte number table, NC inputs

Byte number range, NC inputs	
axis number	axis byte numbers
1	0-7
2	8-15
3	16-23
4	24-31
5	32-39
6	40-47
7	48-55
8	56-63
9	64-71
10	72-79
11	80-87
12	88-95
13	96-103
14	104-111
15	112-119
16	120-127
spindle number	spindle byte numbers
1	256-263
2	264-271
3	272-279
4	280-287
5	288-295
6	296-303
7	304-311
8	312-319
channel number	channel byte number
0	384-391
1	392-399
2	400-407
3	408-415
4	416-423
5	424-431
6	432-439
7	440-447
8	448-455
9	456-463
10	464-471
11	472-479
12	480-487
13	488-495

3.1.2 Byte number table, machine (Profibus DP)

Byte number range, machine (Profibus DP)	
machine	machine byte numbers
machine	600-1111

3.1.3 Byte number table, future NC inputs

The following table shows the byte numbers for a possible increase in the number of axes available in the Typ3.

Future byte number range, NC inputs	
axis number	axis byte numbers
17	128-135
18	136-143
19	144-151
20	152-159
21	160-167
22	168-175
23	176-183
24	184-191
25	192-199
26	200-207
27	208-215
28	216-223
29	224-231
30	232-239
31	240-247
32	248-255

3.1.4 Overview of axis-related interface signals, PLC → NC

In the following overviews of interface signals, the byte numbers shown are examples relating to the 1st axis, 1st spindle, or 1st channel. As regards the other axes, spindles, and channels, the byte numbers are listed in the tables of section 3.1.1 through 3.1.3.

PLC → NC (1st axis)	
Interface signal	Designation
0.0	Manual +
0.1	Manual –
0.2	Axis override 100%
0.3	Inch incr. step
0.4	res.
0.5	Handwheel, direction of rotation
0.6*	Safe manual operation mode
0.7	res.
1.0	Manual feed/incremental step 2 ⁰
1.1	Manual feed/incremental step 2 ¹
1.2	Manual feed/incremental step 2 ²
1.3	Manual feed/incremental step 2 ³
1.4	Travel limit range 2 ⁰
1.5	Travel limit range 2 ¹
1.6	Mode selection bit 0
1.7	Mode selection bit 1
2.0	Handwheel assignment bit 0
2.1	Handwheel assignment bit 1
2.2	res.
2.3	res.
2.4	res.
2.5*	Travel to next grid position (Hirth axis)
2.6	res.
2.7	res.
3.0	Axis is discharged
3.1	Control reset (auxiliary axis)
3.2	Suppress software limit switches
3.3	res.
3.4	res.
3.5	res.
3.6	res.
3.7	Torque reduction

* not yet available

PLC → NC (1st axis)	
Interface signal	Designation
4.0	Override bit 0
4.1	Override bit 1
4.2	Override bit 2
4.3	Override bit 3
4.4	Override bit 4
4.5	Couple lag: NC error off
4.6	Standstill torque: NC error off
4.7	res.
5.0	res.
5.1	res.
5.2	res.
5.3	res.
5.4	res.
5.5	res.
5.6	res.
5.7	res.
6.0	Customer input 1
6.1	Customer input 2
6.2	Customer input 3
6.3	Customer input 4
6.4	Customer input 5
6.5	Customer input 6
6.6	Customer input 7
6.7	Customer input 8
7.0	Hold command position
7.1	res.
7.2	res.
7.3	res.
7.4	res.
7.5	Σ Drive inhibit
7.6	Σ Drive off
7.7	Σ Feed inhibit

Extended interface signals can be accessed by special functions.

PLC → NC (1st axis extension)	
Interface signal	Designation
8.0	Drive inhibit 1
8.1	Drive inhibit 2
8.2	Drive inhibit 3
8.3	Drive inhibit 4
8.4	Drive inhibit 5
8.5	Drive inhibit 6
8.6	Drive inhibit 7
8.7	Drive inhibit 8
9.0	Drive inhibit 9
9.1	Drive inhibit 10
9.2	Drive inhibit 11
9.3	Drive inhibit 12
9.4	Drive inhibit 13
9.5	Drive inhibit 14
9.6	Drive inhibit 15
9.7	Drive inhibit 16
10.0	Drive off 1
10.1	Drive off 2
10.2	Drive off 3
10.3	Drive off 4
10.4	Drive off 5
10.5	Drive off 6
10.6	Drive off 7
10.7	Drive off 8
11.0	Drive off 9
11.1	Drive off 10
11.2	Drive off 11
11.3	Drive off 12
11.4	Drive off 13
11.5	Drive off 14
11.6	Drive off 15
11.7	Drive off 16

Extended interface signals can be accessed by special functions.

PLC → NC (1st axis extension)	
Interface signal	Designation
12.0	Feed inhibit 1
12.1	Feed inhibit 2
12.2	Feed inhibit 3
12.3	Feed inhibit 4
12.4	Feed inhibit 5
12.5	Feed inhibit 6
12.6	Feed inhibit 7
12.7	Feed inhibit 8
13.0	Feed inhibit 9
13.1	Feed inhibit 10
13.1	Feed inhibit 11
13.3	Feed inhibit 12
13.4	Feed inhibit 13
13.5	Feed inhibit 14
13.6	Feed inhibit 15
13.7	Feed inhibit 16

3.1.5 Overview of spindle-related interface signals PLC → NC

PLC → NC (1st spindle)	
Interface signal	Designation
256.0	Spindle M3 manual
256.1	Spindle M4 manual
256.2	Spindle M5 manual
256.3	res.
256.4	Spindle jog M3
256.5	Spindle jog M4
256.6 *	Safe manual operation mode
256.7 *	In position, activate range 2
257.0	Spindle speed jog 2 ⁰
257.1	Spindle speed jog 2 ¹
257.2	Spindle speed jog 2 ²
257.3	res.
257.4	res.
257.5	res.
257.6	res.
257.7	res.
258.0	Override 2 ⁰
258.1	Override 2 ¹
258.2	Override 2 ²
258.3	Override 2 ³
258.4	Override 2 ⁴
258.5	res.
258.6	res.
258.7	res.
259.0 *	Spindle orientation dir. 2 ⁰
259.1 *	Spindle orientation dir. 2 ¹
259.2	Spindle orientation manual
259.3	Spindle override 100%
259.4	Reset (spindle)
259.5	res.
259.6	res.
259.7	res.

* not yet available

PLC → NC (1st spindle)	
Interface signal	Designation
260.0	res.
260.1	res.
260.2	res.
260.3	res.
260.4	res.
260.5	res.
260.6	res.
260.7	res.
261.0	Gear range 1 acknowledge
261.1	Gear range 2 acknowledge
261.2	Gear range 3 acknowledge
261.3	Gear range 4 acknowledge
261.4	res.
261.5	res.
261.6	res.
261.7	Gear in idle motion acknowledge
262.0	Customer input 1
262.1	Customer input 2
262.2	Customer input 3
262.3	Customer input 4
262.4	Customer input 5
262.5	Customer input 6
262.6	Customer input 7
262.7	Customer input 8
263.0	res.
263.1	res.
263.2	res.
263.3	res.
263.4	res.
263.5	Σ Drive inhibit
263.6	Σ Drive off
263.7	Σ Spindle inhibit

Extended interface signals can be accessed by special functions.

PLC → NC (1st spindle extension)	
Interface signal	Designation
8.0	Drive inhibit 1
8.1	Drive inhibit 2
8.2	Drive inhibit 3
8.3	Drive inhibit 4
8.4	Drive inhibit 5
8.5	Drive inhibit 6
8.6	Drive inhibit 7
8.7	Drive inhibit 8
9.0	Drive inhibit 9
9.1	Drive inhibit 10
9.2	Drive inhibit 11
9.3	Drive inhibit 12
9.4	Drive inhibit 13
9.5	Drive inhibit 14
9.6	Drive inhibit 15
9.7	Drive inhibit 16
10.0	Drive off 1
10.1	Drive off 2
10.2	Drive off 3
10.3	Drive off 4
10.4	Drive off 5
10.5	Drive off 6
10.6	Drive off 7
10.7	Drive off 8
11.0	Drive off 9
11.1	Drive off 10
11.2	Drive off 11
11.3	Drive off 12
11.4	Drive off 13
11.5	Drive off 14
11.6	Drive off 15
11.7	Drive off 16

Extended interface signals can be accessed by special functions.

PLC → NC (1st spindle extension)	
Interface signal	Designation
12.0	Spindle inhibit 1
12.1	Spindle inhibit 2
12.2	Spindle inhibit 3
12.3	Spindle inhibit 4
12.4	Spindle inhibit 5
12.5	Spindle inhibit 6
12.6	Spindle inhibit 7
12.7	Spindle inhibit 8
13.0	Spindle inhibit 9
13.1	Spindle inhibit 10
13.2	Spindle inhibit 11
13.3	Spindle inhibit 12
13.4	Spindle inhibit 13
13.5	Spindle inhibit 14
13.6	Spindle inhibit 15
13.7	Spindle inhibit 16

3.1.6 Overview of channel-related interface signals PLC → NC

PLC → NC (1st channel)	
Interface signal	Designation
384.0	Conditional jump 1
384.1*	Conditional jump 2
384.2*	Conditional jump 3
384.3*	Conditional jump 4
384.4*	Conditional subprogram call 1
384.5*	Conditional subprogram call 2
384.6*	Conditional subprogram call 3
384.7*	Conditional subprogram call 4
385.0	Cycle start
385.1	Control reset
385.2	1st override 100%
385.3*	Rapid without override
385.4*	Automatic restart
385.5*	Contour retrace
385.6	Optional stop
385.7	Limit rapid travel
386.0	Mode selection 2 ⁰
386.1	Mode selection 2 ¹
386.2	Mode selection 2 ²
386.3	Mode selection 2 ³
386.4*	Call function after interrupt 1
386.5*	Call function after interrupt 2
386.6*	Call function after interrupt 4
386.7*	Call function after interrupt 8
387.0	Mode selection by PLC
387.1*	Return to contour
387.2	res.
387.3	Cancel distance to go
387.4	Block skip 1
387.5*	Block skip 2
387.6*	Block skip 3
387.7*	Block skip 4

* not yet available

PLC → NC (1st channel)	
Interface signal	Designation
388.0	1st override 2 ⁰
388.1	1st override 2 ¹
388.2	1st override 2 ²
388.3	1st override 2 ³
388.4	1st override 2 ⁴
388.5	res.
388.6	res.
388.7	res.
389.0	res.
389.1	res.
389.2	res.
389.3	res.
389.4	res.
389.5	res.
389.6	res.
389.7	res.
390.0	Customer input 1
390.1	Customer input 2
390.2	Customer input 3
390.3	Customer input 4
390.4	Customer input 5
390.5	Customer input 6
390.6	Customer input 7
390.7	Customer input 8
391.0	Automatic program reselection inactive
391.1	res.
391.2	res.
391.3	res.
391.4	res.
391.5	Σ Feed inhibit
391.6	Σ Feed hold
391.7	Σ Block transfer inhibit

Extended interface signals can be accessed by special functions.

PLC → NC (1st channel extension)	
Interface signal	Designation
8.0	Feed inhibit general 1
8.1	Feed inhibit general 2
8.2	Feed inhibit general 3
8.3	Feed inhibit general 4
8.4	Feed inhibit general 5
8.5	Feed inhibit general 6
8.6	Feed inhibit general 7
8.7	Feed inhibit general 8
9.0	Feed inhibit general 9
9.1	Feed inhibit general 10
9.2	Feed inhibit general 11
9.3	Feed inhibit general 12
9.4	Feed inhibit general 13
9.5	Feed inhibit general 14
9.6	Feed inhibit general 15
9.7	Feed inhibit general 16
10.0	Feed hold 1
10.1	Feed hold 2
10.2	Feed hold 3
10.3	Feed hold 4
10.4	Feed hold 5
10.5	Feed hold 6
10.6	Feed hold 7
10.7	Feed hold 8
11.0	Feed hold 9
11.1	Feed hold 10
11.2	Feed hold 11
11.3	Feed hold 12
11.4	Feed hold 13
11.5	Feed hold 14
11.6	Feed hold 15
11.7	Feed hold 16

Extended interface signals can be accessed by special functions.

PLC → NC (1st channel extension)	
Interface signal	Designation
12.0	Block transfer inhibit 1
12.1	Block transfer inhibit 2
12.2	Block transfer inhibit 3
12.3	Block transfer inhibit 4
12.4	Block transfer inhibit 5
12.5	Block transfer inhibit 6
12.6	Block transfer inhibit 7
12.7	Block transfer inhibit 8
13.0	Block transfer inhibit 9
13.1	Block transfer inhibit 10
13.2	Block transfer inhibit 11
13.3	Block transfer inhibit 12
13.4	Block transfer inhibit 13
13.5	Block transfer inhibit 14
13.6	Block transfer inhibit 15
13.7	Block transfer inhibit 16

3.2 NC interface: Outputs

3.2.1 Byte number table, NC outputs

Byte number range, NC outputs	
axis number	axis byte numbers
1	0-7
2	8-15
3	16-23
4	24-31
5	32-39
6	40-47
7	48-55
8	56-63
9	64-71
10	72-79
11	80-87
12	88-95
13	96-103
14	104-111
15	112-119
16	120-127
spindle number	spindle byte numbers
1	256-263
2	264-271
3	272-279
4	280-287
5	288-295
6	296-303
7	304-311
8	312-319
channel number	channel byte number
0	384-395
1	396-407
2	408-419
3	420-431
4	432-443
5	444-455
6	456-467
7	468-479
8	480-491
9	492-503
10	504-515
11	516-527
12	528-539
13	540-551

3.2.2 Byte number table, machine (Profibus DP)

Byte number range, machine (Profibus DP)	
machine	machine byte numbers
machine	600-1111

3.2.3 Byte number table, future NC outputs

The following table shows the byte numbers for a possible increase in the number of axes available in the Typ3.

Future byte number range, NC outputs	
axis number	axis byte numbers
17	128-135
18	136-143
19	144-151
20	152-159
21	160-167
22	168-175
23	176-183
24	184-191
25	192-199
26	200-207
27	208-215
28	216-223
29	224-231
30	232-239
31	240-247
32	248-255

3.2.4 Overview of axis-related interface signals, NC → PLC

In the following overviews of interface signals, the byte numbers shown are examples relating to the 1st axis, 1st spindle, or 1st channel. As regards the other axes, spindles, and channels, the byte numbers are listed in the tables of section 3.2.1 through 3.2.3.

NC → PLC (1st axis)	
Interface signal	Designation
0.0	Sercos system ready
0.1	Drive ready
0.2	Drive under control
0.3	Travel command
0.4	Traversing direction -
0.5	Axis running
0.6	Axis in position
0.7 *	In position range 2 activated
1.0	Axis inhibited (test)
1.1	Override 0%
1.2	Override 100%
1.3	Control reset acknowledge
1.4 *	Limit switch range 2 ⁰
1.5 *	Limit switch range 2 ¹
1.6 *	Reference point was reached
1.7	Reference point is known
2.0	Index of the master axis bit 2 ⁰
2.1	Index of the master axis bit 2 ¹
2.2	Index of the master axis bit 2 ³
2.3	Index of the master axis bit 2 ⁴
2.4	Index of the master axis bit 2 ⁵
2.5*	Axis in grid position (Hirth axis)
2.6	Couple lag exceeded
2.7	Standstill torque exceeded
3.0	res.
3.1	res.
3.2	Axis near endpoint
3.3	Error diagnostics class 1
3.4	Class 2 diagnostics change
3.5	Class 3 diagnostics change
3.6	Torque reduced
3.7	Drive-controlled interpolation

* not yet available

NC → PLC (1st axis)	
Interface signal	Designation
4.0	Axis position 1
4.1	Axis position 2
4.2	Axis position 3
4.3	Axis position 4
4.4	Axis position 5
4.5	Axis position 6
4.6	Axis position 7
4.7	Axis position 8
5.0	Channel number bit 0
5.1	Channel number bit 1
5.2	Channel number bit 2
5.3	Channel number bit 3
5.4	res.
5.5	res.
5.6	res.
5.7	res.
6.0	Customer output 1
6.1	Customer output 2
6.2	Customer output 3
6.3	Customer output 4
6.4	Customer output 5
6.5	Customer output 6
6.6	Customer output 7
6.7	Customer output 8
7.0 *	SERCOS signal status word bit 0
7.1 *	SERCOS signal status word bit 1
7.2 *	SERCOS signal status word bit 2
7.3 *	SERCOS signal status word bit 3
7.4 *	SERCOS signal status word bit 4
7.5 *	SERCOS signal status word bit 5
7.6 *	SERCOS signal status word bit 6
7.7 *	SERCOS signal status word bit 7

* not yet available

3.2.5 Overview of spindle-related interface signals NC → PLC

NC → PLC (1st spindle)	
Interface signal	Designation
256.0	Sercos system ready
256.1	Drive ready
256.2	Drive under control
256.3	Spindle command
256.4	Direction of rotation M4
256.5	Gear range switching active
256.6	Spindle in position
256.7 *	Spindle in position range 2 activated
257.0	Spindle orientation active
257.1	Spindle is oriented
257.2	Progr. spindle speed reached
257.3	Override 100%
257.4	Idling speed reached
257.5	Spindle stopped
257.6*	Speed too high for gear range
257.7*	Speed too low for gear range
258.0	Output gear range 1
258.1	Output gear range 2
258.2	Output gear range 3
258.3	Output gear range 4
258.4	res.
258.5	res.
258.6	res.
258.7	Output gear in idle motion
259.0	C axis is active
259.1	Spindle speed limited
259.2	Override 0%
259.3	Error diagnostics class 1
259.4	Class 2 diagnostics change
259.5	Class 3 diagnostics change
259.6	Position control active
259.7	C axis switching active

* not yet available

NC → PLC (1st spindle)	
Interface signal	Designation
260.0	Index of couple bit 0
260.1	Index of couple bit 1
260.2	Index of couple bit 2
260.3	Spindle is master
260.4	Synchronous 1
260.5	Synchronous 2
260.6	Coupling error
260.7	res.
261.0	res.
261.1	res.
261.2	res.
261.3	res.
261.4	Control reset acknowledge
261.5	res.
261.6	res.
261.7	res.
262.0	Customer output 1
262.1	Customer output 2
262.2	Customer output 3
262.3	Customer output 4
262.4	Customer output 5
262.5	Customer output 6
262.6	Customer output 7
262.7	Customer output 8
263.0	SERCOS signal status word bit 0
263.1	SERCOS signal status word bit 1
263.2	SERCOS signal status word bit 2
263.3	SERCOS signal status word bit 3
263.4	SERCOS signal status word bit 4
263.5	SERCOS signal status word bit 5
263.6	SERCOS signal status word bit 6
263.7	SERCOS signal status word bit 7

3.2.6 Overview of channel-related interface signals NC → PLC

NC → PLC (1st channel)	
Interface signal	Designation
384.0	Program running
384.1	Feed hold active
384.2	Control reset acknowledge
384.3	Block transfer inhibit active
384.4*	Test – G0
384.5*	Test – Feed
384.6	Test without motion
384.7*	Test without compensation
385.0	Active mode 2 ⁰
385.1	Active mode 2 ¹
385.2	Active mode 2 ²
385.3	Active mode 2 ³
385.4*	Activate conditional jump 1
385.5*	Activate conditional jump 2
385.6*	Activate conditional jump 3
385.7*	Activate conditional jump 4
386.0	NC ready
386.1	Program stop with M0, M1
386.2	Program end with M2, M30
386.3*	Optional stop activated
386.4	Ready for re-entry
386.5*	Re-entry active
386.6	Remove finished
386.7*	Contour retrace active
387.0*	Activate block skip 1
387.1*	Activate block skip 2
387.2*	Activate block skip 3
387.3*	Activate block skip 4
387.4*	Cond. subprogram call 1 active
387.5*	Cond. subprogram call 2 active
387.6*	Cond. subprogram call 3 active
387.7*	Cond. subprogram call 4 active

* not yet available

NC → PLC (1st channel)	
Interface signal	Designation
388.0	G131 tool rotation
388.1*	Dry run to block
388.2*	Dry run to block without compensation
388.3*	Override 0%
388.4*	Override 100%
388.5	res.
388.6	res.
388.7	Inpos range 2 active
389.0	External tool correction active 2 ⁰
389.1	External tool correction active 2 ¹
389.2	External tool correction active 2 ²
389.3	External tool correction active 2 ³
389.4	External zero offset active 2 ⁰
389.5	External zero offset active 2 ¹
389.6	res.
389.7	res.
390.0	Rapid traverse active
390.1	G70 active
390.2	G32 active
390.3*	G33 active
390.4	res.
390.5	G41 active
390.6	G42 active
390.7	G63 active
391.0	G189 active
391.1	G90 active
391.2	G91 active
391.3	G92 active
391.4	G95 active
391.5	Const. cutting velocity active
391.6	G190 active
391.7	G191 active

* not yet available

NC → PLC (1st channel)	
Interface signal	Designation
392.0	Customer output 1
392.1	Customer output 2
392.2	Customer output 3
392.3	Customer output 4
392.4	Customer output 5
392.5	Customer output 6
392.6	Customer output 7
392.7	Customer output 8
393.0	General tool correction bit 0
393.1	General tool correction bit 1
393.2	General tool correction bit 2
393.3	General tool correction bit 3
393.4	res.
393.5	res.
393.6	res.
393.7	res.
394.0	CPL customer output 1
394.1	CPL customer output 2
394.2	CPL customer output 3
394.3	CPL customer output 4
394.4	CPL customer output 5
394.5	CPL customer output 6
394.6	CPL customer output 7
394.7	CPL customer output 8
395.0	CPL customer output 9
395.1	CPL customer output 10
395.2	CPL customer output 11
395.3	CPL customer output 12
395.4	CPL customer output 13
395.5	CPL customer output 14
395.6	CPL customer output 15
395.7	CPL customer output 16

3.3 Global interface

3.3.1 Overview of inputs of global interface

PLC → NC (global interface)	
Interface signal	Designation
0.0	System control reset
0.1	Edit inhibit
0.2	res.
0.3	res.
0.4	res.
0.5	res.
0.6	res.
0.7	res.
1.0	Stroke inhibit
1.1	Stroke reservation
1.2	Stroke on
1.3	res.
1.4	res.
1.5	res.
1.6	res.
1.7	res.
2.0	res.
2.1	res.
2.2	res.
2.3	res.
2.3	res.
2.4	res.
2.5	res.
2.6	res.
2.7	res.
3.0	res.
3.1	res.
3.2	res.
3.3	res.
3.4	res.
3.4	res.
3.5	res.
3.6	res.
3.7	res.

3.3.2 Overview of outputs of global interface

NC → PLC (global interface)	
Interface signal	Designation
4.0	res.
4.1	res.
4.2	res.
4.3	res.
4.4	res.
4.5	res.
4.6	res.
4.7	res.
5.0	Stroke intended
5.1	Stroke is not running
5.2	res.
5.3	res.
5.4	res.
5.5	res.
5.6	res.
5.7	res.
6.0	res.
6.1	res.
6.2	res.
6.3	res.
6.3	res.
6.4	res.
6.5	res.
6.6	res.
6.7	res.
7.0	res.
7.1	res.
7.2	res.
7.3	res.
7.4	res.
7.4	res.
7.5	res.
7.6	res.
7.7	res.

3.4 Machine interface

The machine interface includes the following:

- **Area 1** for communication with the machine,
including
 - 512 bytes for inputs and 512 bytes for outputs
 - communication via Profibus DP

The range of addresses to be activated includes 4096 bits for signals in the byte number range from 600 to 1111 (see also the tables shown in sections 3.1.2 and 3.2.2).

- **Area 2** for communication with the machine,
including
 - MCP1: 16 bytes for inputs and 4 bytes for outputs
 - MCP2: 16 bytes for inputs and 4 bytes for outputs

Communication is performed via **CAN bus**. Access is made via the following byte number ranges:

Inputs:

- MCP1: 1200 ... 1215
- MCP2: 1216 ... 1231

Outputs:















































































- MCP1: 1200 ... 1203
- MCP2: 1204 ... 1207

For signal assignment, see the following table:

3.4.1 Signal assignment area 2: Inputs

Starting address of MCP1 = 1200

Starting address of MCP2 = 1216

	7	6	5	4	3	2	1	Bit 0
Byte 0	S28 	S27 	S26 	S25 	S24 	S23 	S22 	S21 
1	S28 	S27 	S26 	S25 	S24 	S23 	S22 	S21 
2			S34 	S33 	S32 	S31 	S30 	S29 
3	S2 	S1 	S34 	S33 	S32 	S31 	S30 	S29 
4	S5 	S4 	S3 	SS2 	SS2 	SS2 	SS2 	SS2 
5	S18 	S17 	S16 	S15 	S14 	S13 	S12 	S11 
6	S10 	S9 	S20 	S19 	SS1 	SS1 	SS1 	SS1 
7	S8 	S7 	S6 	SS3 	SS3 	SS3 	SS3 	SS3 
8	ST5/6 	ST5/5 	ST5/4 	ST5/3 	ST5/2 	ST3/3 	ST3/15 	ST3/2 
9	ST3/9 	ST3/21 	ST3/8 	ST3/20 	ST3/7 	ST3/19 	ST3/6 	ST3/18 
10	res.	res.	res.	res.	res.	res.	res.	res.
11	res.	res.	res.	res.	res.	res.	res.	res.
12	res.	res.	res.	res.	res.	res.	res.	res.
13	res.	res.	res.	res.	res.	res.	res.	res.
14	res.	res.	res.	res.	res.	res.	res.	res.
15	res.	res.	res.	res.	res.	res.	res.	res.

SS1 rotary switch, manual feed/increment

SS2 rotary switch, feed potentiometer

SS3 rotary switch, spindle potentiometer

ST3 handwheel

ST5 additional inputs

3.4.2 Signal assignment area 2: Outputs

Starting address of MCP1 = 1200

Starting address of MCP2 = 1204

	7	6	5	4	3	2	1	Bit 0
Byte 0	H16	H15	H14	H13	H12	H11	H10	H9
1	H8	H7	H6	H5	H4	H3	H2	H1
2	H32	H31	H30	H29	H28	H27	H26	H25
3	H24	H23	H22	H21	H20	H19	H18	H17

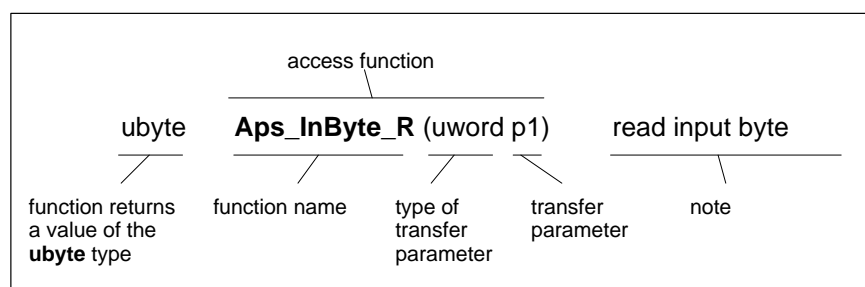
Notes:

4 Signal accessing

The interface is accessed through a number of **functions** which can be accessed from the KNS subsystem. The names of these functions are defined in header file **aps1_if.h** and should be used with their defined designations. Use "Include" to add the header file to the module **kns_main.c**. For details on the header file, see the online help of the development environment (see also the preface of this manual).

☞ **In exceptional cases, application-specific adjustments of function names and function types can be made.**

Explanation of the description of the following access functions:



4.1 Functions for accessing the NC interface

4.1.1 Function for accessing the input image

Read access

ubyte	Aps_InBit_R (uword p1, ubyte p2)	read input bit
ubyte	Aps_InByte_R (uword p1)	read input byte
uword	Aps_InWord_R (uword p1)	read input word
udword	Aps_InDWord_R (uword p1)	read input double word

The parameters to be transferred define the following:

p1: byte number within the interface range

p2: bit number to be read (0–7)

Write access

void	Aps_InBit_W (uword p1, ubyte p2, ubyte p3)	write input bit
void	Aps_InByte_W (uword p1, ubyte p2)	write input byte
void	Aps_InWord_W (uword p1, ubyte p2)	write input word
void	Aps_InDWord_W (uword p1, ubyte p2)	write input double word

The parameters to be transferred define the following:

p1: byte number within the interface range

p2: bit number (0–7) to be written for bits to be accessed, or value to be written to the interface in case of access other than bit access

p3: status of the respective bit (0–1)

4.1.2 Functions for accessing the output image

Read access

ubyte	Aps_OutBit_R (uword p1, ubyte p2)	read output bit
ubyte	Aps_OutByte_R (uword p1)	read output byte
uword	Aps_OutWord_R (uword p1)	read output word
udword	Aps_OutDWord_R (uword p1)	read output double word

The parameters to be transferred define the following:

p1: byte number within the interface range

p2: bit number (0–7) to be read

Write access

```
void Aps_OutBit_W(uword p1, ubyte p2, ubyte p3) write output bit
void Aps_OutByte_W(uword p1, ubyte p2) write output byte
void Aps_OutWord_W(uword p1, ubyte p2) write output word
void Aps_OutDWord_W(uword p1, ubyte p2) write output double
word
```

The parameters to be transferred define the following:

p1: byte number within the interface range
p2: bit number (0–7) to be written for bits to be accessed, or value to be written to the interface in case of access other than bit access
p3: status of the respective bit (0–1)

**CAUTION**

Ranges 0–599 are accessed directly on the internal interface of the Typ3 osa (no image).

Therefore, the editing of signals > 1 bit (operating mode, e.g.) should be made using the byte, word, or double word functions.

4.2 Functions for accessing markers (flags)

In the software PLC, there is an area for **global variables**. These variables are equivalent to the former **markers (flags)**.

There is a total of 51,200 bit markers available, of which

- 49,152 are volatile and
- 2,048 are retentive.

The **volatile markers** are created whenever the Typ3 osa is started. They are initialized at value 0.

The **retentive markers** are stored in a buffered memory area. Unless a system error occurs, this area is never deleted and its data is retained when the control is switched off.

In order to delete on purpose the retentive marker area, it is necessary to start the Typ3 osa control in switch position 1 (osa master).



CAUTION

When starting the Typ3 osa control in switch position 1, non-volatile data modules, too, will be deleted!

When the retentive marker area has been deleted, this is indicated by an error message on the status line of the Typ3 osa control panel.

Markers can be accessed in the read mode or the write mode at bit, byte, word or double word level. Access is possible via 8 functions available to the user at the system interface.

Read access

ubyte	Aps_MBit_R (uword p1, ubyte p2)	read bit marker
ubyte	Aps_MByte_R (uword p1)	read byte marker
uword	Aps_MWord_R (uword p1)	read word marker
udword	Aps_MDWord_R (uword p1)	read double word marker

The parameters to be transferred define the following:

- p1:** byte number within the marker range:
- volatile markers (0...6143 bytes)
 - retentive markers (6144...6399 bytes)
- p2:** bit number within a byte (0–7)

Write access

```
void Aps_MBit_W(uword p1, ubyte p2, ubyte p3) write bit marker
void Aps_MByte_W(uword p1, ubyte p2)       write byte marker
void Aps_MWord_W(uword p1, ubyte p2)      write word marker
void Aps_MDWord_W(uword p1, ubyte p2)    write double word
                                             marker
```

The parameters to be transferred define the following:

- p1:** byte number within the marker range:
 - volatile markers (0...6143 bytes)
 - retentive markers (6144...6399 bytes)
- p2:** bit number (0–7) to be written for bits to be accessed, or value to be written to the interface in case of access other than bit access
- p3:** status of the respective bit (0–1)

4.2.1 Access to non-existent markers (flags)

When marker functions are called whose byte numbers are outside the valid ranges, no specific error message is displayed. When access is attempted in the read mode, the value returned will always be 0. When access is attempted in the write mode, the addressed range will remain unmodified.

4.3 Functions for accessing data modules

4.3.1 General functions for accessing data modules

Data modules are global memory areas which must first be created by the user.

Available data modules include


- 128 volatile data modules and (length:128...1024 bytes)
- up to 247 non-volatile data modules (length:128...31616 bytes)

The **length** of a data module is defined when it is created.

The number of non-volatile data modules depends on the lengths of the individual modules.

Example: 1 data module of a length of 31,616 bytes, or
247 data modules of a length of 128 bytes, ea.

Creating memory for volatile data module

 **Do not use for new programs.**
Will be replaced by 'Aps_DMCreate' (see page 4-7).

uword **Aps_Erzeuge_DB**(uword *p1, ubyte p2, ubyte p3, uword p4)

The parameters to be transferred define the following:

- p1:** number of the new data module
- p2:** upper search limit in the 0...127 range
- p3:** lower search limit in the 0...127 range
- p4:** specified length of the data module

The maximum length of a volatile data module to be created is 1024 bytes. Because storage allocation is only made in steps of 128 bytes, the length is set to a permissible boundary value.

If the function has been executed successfully, the return value is 0. If an error has occurred, values other than 0 are returned.

Errors will occur if

- p2 or p3 are outside the 0...127 range
- or p2 < p3.

Supplying handles to data module

 **Do not use for new programs.**
Will be replaced by Aps_DM_Open (see page 4-7).

void* **Aps_Oeffne_DB**(ubyte p1)

The parameter to be transferred defines the following:

- p1:** number of the data module (0...127)

This handle is used for all data module access operations. If a non-existent data module number is entered to open a data module, or if the data module has not been created with function **Aps_Erzeuge_DB**() (exception: non-volatile data modules), the handle value returned is **ZERO**.

 **This handle does not refer to the data area of a data module.**

Creating data module

uword **Aps_DMCreate**(uword *nr, ubyte ug, ubyte og, uword lng, ubyte dbk)

This function contains the following parameters:

- *nr:** handle on the variable where the number of the data module created is located
- ug:** lower limit of data module number:
 - volatile data modules: max. 0–127
 - non-volatile data modules: max. 0–255
- og:** upper limit of data module number:
 - volatile data modules: max. 0–127
 - non-volatile data modules: max. 0–255
- lng:** length of the data module:
 - volatile data modules: max. 1,024 bytes
 - non-volatile data modules: max. 31,616 bytes
- dbk:** data module identifier:
 - volatile data modules: 0
 - non-volatile data modules: 1

Volatile data modules are stored in dynamic RAM, whereas for storing **non-volatile** data modules, existing memory space is simply partitioned. Storage allocation is only made in steps of 128 bytes. If an error occurs when creating a data module, this is indicated in the form of an error return value.

Supplying handles to the administration part of the data module

void ***Aps_DMOpen**(ubyte dbnr, ubyte dbk)

This function contains the following parameters:

- dbnr:** number of the data module:
 - volatile data modules: 0...127
 - non-volatile data modules: 0...255
- dbk:** data module identifier:
 - volatile data modules: 0
 - non-volatile data modules: 1

This handle is used for all data module access operations. If a module is requested for which no memory exists, the function returns a zero handle.

Starting address of the data range of a data module

uword **Aps_DMAdresse**(void *p1)

The parameter to be transferred defines the following:

p1: handle on administration part of the data module

Length of the data range of a data module

uword **Aps_DMSize**(void *p1)

The parameter to be transferred defines the following:

p1: handle on administration part of the data module

4.3.2 Functions for read and write access on a data module

Read access

ubyte	Aps_DMBit_R (void *p1, uword p2, ubyte p3)	read bit from data module
ubyte	Aps_DMByte_R (void *p1, uword p2)	read byte from data module
uword	Aps_DMWord_R (void *p1, uword p2)	read word from data module
udword	Aps_DMDWord_R (void *p1, uword p2)	read double word from data module

The parameters to be transferred define the following:

- p1:** handle on administration part of the data module
- p2:** byte number within the data module area (0– data module length –1)
- p3:** bit number within a byte (0–7)

Write access

void	Aps_DMBit_W (void *p1, uword p2, ubyte p3, ubyte p4)	write bit to data module
void	Aps_DMByte_W (void *p1, uword p2, ubyte p3)	write byte to data module
void	Aps_DMWordS_W (void *p1, uword p2, ubyte p3)	write word to data module
void	Aps_DMDWord_W (void *p1, uword p2, ubyte p3)	write double word to data module

The parameters to be transferred define the following:

- p1:** handle on administration part of the data module
- p2:** byte number within a data module area (0– data module length –1)
- p3:** bit number within a byte (0–7)
- p3:** value to be written in the case of byte, word, or double word access
- p4:** state of a bit (0–1)

4.4 Functions for accessing timer modules

4.4.1 Timer module types

There are 128 timer modules available for the user to implement timer functions. These timer modules are grouped into 5 different types:

- pulse timers
- extended pulse timers
- on-delay timers
- extended on-delay timers
- off-delay timers

Timing is based on the system clock frequency of the Typ3 osa, which has a timing resolution of 10 ms.

The 5 timer module types are described in the following.

Key of the following graphs:

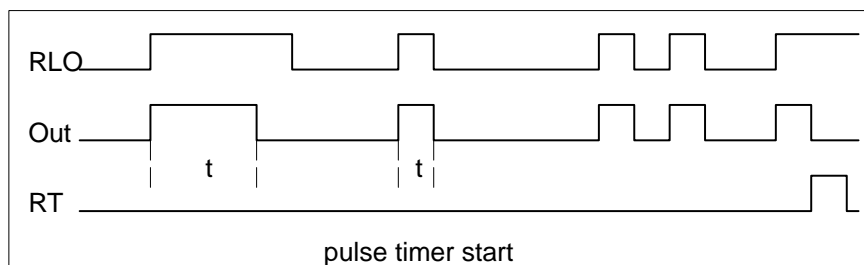
RLO = result of logic operation (starting pulse of timer module)

Out = timer module output

RT = timer module reset

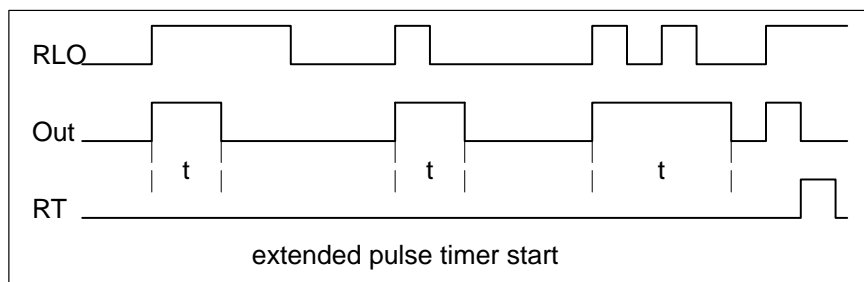
Timer module type 1: Pulse timer

Upon starting the timer, the output changes to 1 and remains high until the time has elapsed, the input has changed to low, or the timer has been reset.



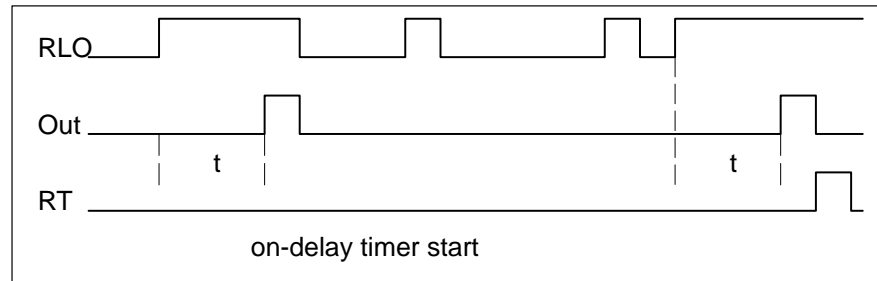
Timer module type 2: Extended pulse timer

Upon starting the timer, the output changes to 1 and remains high until the time has elapsed or the timer has been reset. The length of the timer run (RLO) has no effect on the preset time.



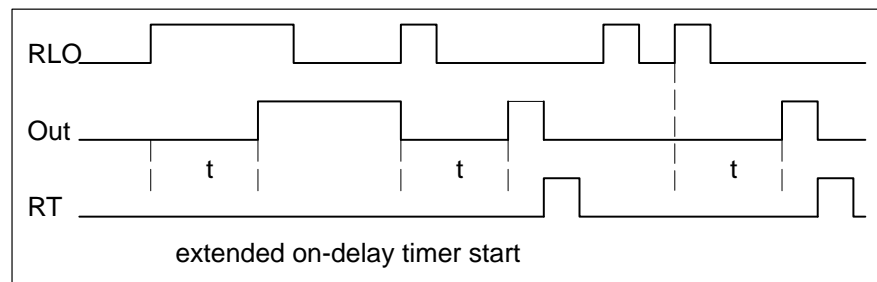
Timer module type 3: On-delay timer

Upon starting the timer, the output changes to 1 as soon as the programmed time interval has elapsed and as long as the timer is still running. If the starting pulses are shorter than the time set, the timer module output remains at low.



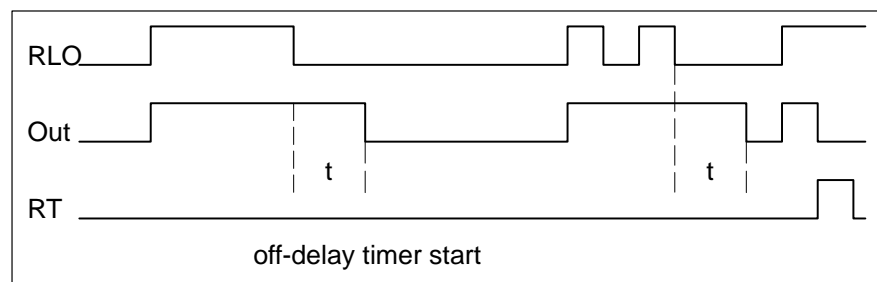
Timer module type 4: Extended on-delay timer

Upon starting the timer, the output changes to 1 when the programmed time has elapsed. The output remains at high until the timer is reset or restarted.



Timer module type 5: Off-delay timer

Upon starting the timer, the timer output changes to 1. It remains high until the timer start and running condition has ceased to exist and the preset time has elapsed, too. Via the reset input, the output can be reset immediately.



4.4.2 Timer functions

Timer modules are accessed through functions available at the subsystem interface. There are 9 different functions available for user application.

Start pulse timer

void **Aps_TimeSignal**(ubyte p1, ubyte p2, uword p3, ubyte p4)

Start extended pulse timer

void **Aps_TimeSignalDelay**(ubyte p1, ubyte p2, uword p3, ubyte p4)

Start on-delay timer

void **Aps_TimeOnSignalDelay**(ubyte p1, ubyte p2, uword p3, ubyte p4)

Start extended on-delay timer

void **Aps_TimeMemOnSignalDelay**(ubyte p1, ubyte p2, uword p3,
ubyte p4)

Start off-delay timer

void **Aps_TimeOffSignalDelay**(ubyte p1, ubyte p2, uword p3, ubyte p4)

Reset timer module

void **Aps_TimeStop**(ubyte p1, ubyte p2)

Read remaining time

uword **Aps_TimeValue**(ubyte p1)

Read set time increment value

ubyte **Aps_TimeScale**(ubyte p1)

The values returned by this function indicate the following:

0: 10 ms
1: 100 ms
2: 1 s
3: 10 s
255: incorrect timer module number

Read timer module status

ubyte **Aps_TimeState**(ubyte p1)

The values returned by this function indicate the following:

0: timer has not been started or time has elapsed
1: timer is running
255: incorrect timer module number

Implement user's own programmable clock

Beside the timer modules described above, the user may also implement his own programmable clock.

```
void Aps_GetSystemTime(unsigned long *p1)
```

The parameter to be transferred defines the following:

p1: time elapsed in milliseconds since system start

Timer function parameters

Timer functions contain from 1 to 4 parameters defining the following:

p1: number of the timer module (0-127)

p2: start/stop condition (0-1)

p3: time value set (0-1023)

p4: time increment (0 = 10 ms, 1 = 100 ms, 2 = 1 s, 3=10 s)

4.5 Functions for accessing counters

There are 64 counters available to the user in the software PLC. Maximum counter value is 1023.

Counters are accessed with the functions shown below.

Set counter module

void **Aps_CounterSet**(ubyte p1, ubyte p2, uword p3)

Reset counter module

void **Aps_CounterReset**(ubyte p1, ubyte p2)

Increment counter module

void **Aps_CounterInc**(ubyte p1, ubyte p2)

Decrement counter module

void **Aps_CounterDec**(ubyte p1, ubyte p2)

Read counting value

uword **Aps_CounterLoad**(ubyte p1, ubyte p2)

Read counter status

ubyte **Aps_CounterState**(ubyte p1, ubyte p2)

The value returned by this function indicates the following:

- 0:** counter status = 0
- 1:** counter status > 0
- 255:** incorrect counter number

Turn counter on / off

void **Aps_CounterOnOff**(ubyte p1, ubyte p2)

Parameters of the counter functions

The counter functions contain from 1 to 3 parameters which define the following:

- p1:** counter number (0–63)
- p2:** enable signal (RLO 0–1)
- p3:** counter value for setting the counter (0–4096)

4.6 Analog I/O

The software PLC is used with a hardware configuration with 8 analog inputs and 4 analog outputs. The hardware is controlled by the PLC using **basic logic functions**. Accessing these functions is only permitted if the analog inputs of the software PLC have previously been assigned with MACODA parameter 407500103 and the analog outputs via parameter 407500104. The value 2 must be entered in these parameters as PLC identification.

Initialize analog interface

```
void Aps_InitAnalog(ubyte p1)
```

The parameter to be transferred defines the following:

p1: time of analog output (0–2)

The values returned by this function indicate the following:

0: not initialized

1: direct output

255: output in I/O state

Analog output

```
void Aps_AnalogOut(ubyte p1, double p2)
```

The parameters to be transferred define the following:

p1: number of the analog output (1–4)

p2: voltage value to be output (–10.0V <-> 10.0V)

Analog input

```
double Aps_AnalogIn(ubyte p1)
```

The parameter to be transferred defines the following:

p1: number of the analog input (1–8)

Analog voltage limitation

The output voltage on the analog outputs may range from –10.0V to 10.0V. With this function, the user can set the upper and lower voltage limits.

```
void Aps_AnalogSetLimits(ubyte p1, double p2, double p3)
```

The parameters to be transferred define the following:

p1: number of the analog output (1–4)

p2: upper voltage limit (max. 10.0V)

p3: lower voltage limit (min. –10.0V)

Read time of output

The time of output set when the analog interface was initialized can be read.

```
ubyte Aps_AnalogOutTime(void)
```

The values returned by this function indicate the following:

0: not initialized

1: direct output

255: output in I/O state

4.7 Access to the extended interface

The extended interface has 6 bytes, ea., and is added to the inputs and outputs of the axis, spindle and channel areas. The interface extension on the output side is inaccessible currently.

The extended interface includes:

- axis area: 16 drive off, 16 drive inhibit, and 16 feed inhibit signals
- spindle area: 16 drive off, 16 drive inhibit, and 16 spindle inhibit signals
- channel area: 16 feed hold, 16 feed inhibit, and 16 block transfer inhibit signals

In the APS subsystem, an inclusive-OR operation is performed on the signals set in the extended area and the respective composite signals are set:

- drive inhibit – drive off – feed inhibit (axis, spindle)
- feed inhibit – feed hold – block transfer inhibit (channel)

The user should not manipulate the RLO signals directly because this may cause errors in signal interpretation.

4.7.1 Read and write access functions to the extended interface

Read access

ubyte **Aps_ExtIF_R**(ubyte p1, ubyte p2, ubyte p3, ubyte p4)

The parameters to be transferred define the following:

- p1:** axis number, spindle number (1–n), channel number (0–n)
- p2:** type of interface (AXIS – SPINDLE – CHANNEL)
- p3:** byte number (8–13)
- p4:** bit number (0–7)

The values returned by this function indicate the following:

- 0:** correct execution
- 1:** incorrect axis, spindle, or channel number
- 2:** wrong byte number
- 3:** wrong bit number
- 4:** wrong type of interface
- 5:** no extended interface available

Write access

ubyte **Aps_ExtIF_W**(ubyte p1, ubyte p2, ubyte p3, ubyte p4, ubyte p5)

The parameters to be transferred define the following:

- p1:** axis number, spindle number (1–n), channel number (0–n)
- p2:** type of interface (AXIS – SPINDLE – CHANNEL)
- p3:** byte number (8–13)
- p4:** bit number (0–7)
- p5:** value (0–1)

The values returned by this function indicate the following:

- 0:** correct execution
- 1:** incorrect axis, spindle, or channel number
- 2:** wrong byte number
- 3:** wrong bit number
- 4:** wrong type of interface
- 5:** no extended interface available

4.8 Accessing actual axis values

The following function is used to read this data:

```
long Aps_Achsdaten(ubyte p1)
```

The parameter to be transferred defines the following:

p1: axis number (1–8)

p1: read potentiometer value (9)

Actual axis values are required quite frequently in the PLC. The user can obtain these values either by using the respective NC interface (Ncs) function, or by accessing the axis function (**Aps_Achsdaten ()**) available at the subsystem interface.

User access to the axis function is very easy. Upon calling this function, the requested data is immediately available for further processing in the software PLC.

The application function can access a global buffer in the software PLC where the axis data is stored. Asynchronous NC interface (Ncs) functions outside the PLC program are used for refreshing the buffer with up-to-date axis data.

In addition to reading actual axis values, function (**Aps_Achsdaten ()**) can be used to read the potentiometer output for the channel.

4.9 Access to the global interface

The global interface signals are higher-level signals which refer to functions valid for the entire NC.

The starting address of the global interface is defined in MACODA parameter **2060 000 8**. If the starting address is "-1", the global interface will not be evaluated. This is the default setting when delivered from the factory.

4.9.1 Read and write access functions to the global interface

Read access

char **Aps_GlobInBit_R**(ubyte p1, ubyte p2) read input bit
 char **Aps_GlobInByte_R**(ubyte p1) read input byte
 char **Aps_GlobOutBit_R**(ubyte p1, ubyte p2) read output bit
 char **Aps_GlobOutByte_R**(ubyte p1) read output byte

The parameters to be transferred define the following:

p1: Byte number of the global interface (0–3)
p2: Bit number of the global interface (0–7)

Write access

char **Aps_GlobInBit_W**(ubyte p1, ubyte p2, ubyte p3) write input bit
 char **Aps_GlobInByte_W**(ubyte p1, ubyte p3) write input byte

The parameters to be transferred define the following:

p1: Byte number of the global interface (0–3)
p2: Bit number of the global interface (0–7)
p3: Value to be transferred

Notes:

5 Auxiliary functions

The auxiliary functions are programmed in the NC part program. All auxiliary functions programmed in one block are transferred via the NC interface (Ncs) to the Aps server, which then distributes them to allocated memory areas in the software PLC. Machine parameters are used to define the allocation of these memory areas. In the software PLC, only the marker areas are available for this purpose to date.

There are different types of auxiliary functions:

- bit-coded auxiliary functions
- 32-bit BCD-coded functions
- 64-bit BCD-coded functions
- 32-bit BCD-coded functions (channel)
- 64-bit BCD-coded functions (channel)
- specific functions

Further, a distinction is made between auxiliary functions

- requiring acknowledgement and those
- not requiring acknowledgement.

Functions requiring acknowledgement must be reset by the user in the software PLC because otherwise the next block will not be executed. To acknowledge an auxiliary function, the marker assigned to it is reset. This is carried out by using the marker functions described above (see section 4.2). For a detailed description of how to acknowledge auxiliary functions, see the ICL 700 Project Planning Manual.

MACODA parameters of the 2060 and 3010 groups are used to specify the auxiliary functions. For detailed information, see the MACODA description.

Notes:

6 Machine error and status display (MSD)

For a description of the MSD functions, see the section on MSD in the ICL 700 Project Planning Manual.

Notes:

7 Ncs functions

Because the software PLC is a module equal to the other modules of the Typ3 software, the user has a number of NC interface (Ncs) functions available for unrestricted use.

However, for reasons of PLC run time, usage should be preferably limited to asynchronous NC interface functions. With these functions, the PLC program run can be continued once the request has been initiated. Independent of the PLC program flow, the data to be received is sent to the proper address where it can be evaluated in one of the next PLC runs. In the case of synchronous function calls, the PLC run is interrupted until the answer to the NC interface (Ncs) function call has been received.

 **A description of the Ncs functions and their call routines is available from Bosch on request.**

Notes:

8 Diagnostics

In the course of a software PLC run, a number of errors may occur which obstruct the execution of the PLC program. Errors may be caused by the PLC program written by the user or by faults of connected peripherals (Profibus DP). The user must then take appropriate steps for debugging.

8.1 Watchdog function

One of the most serious errors in a PLC program is an endless loop which knocks out all functions of the entire Typ3 osa. To prevent this, a watchdog function is activated with the cyclic calls. This function checks the PLC program continuously for time-outs of fixed time intervals. The time interval results from the cyclic run time set in machine parameter 2060000202: The watchdog function is triggered if twice the run time is exceeded. A critical system error occurs and the ready contact is opened.

An error like this may prove very difficult to debug in the software PLC. Therefore, we have created a way to deactivate the watchdog function. To do so, with the first call of the debugger the following string sequence must be entered:

apss (select the Aps subsystem)
p Aps_WatchOff() (deactivate watchdog)

Subsequently, debugging can be carried out. Upon completion of debugging, the watchdog must be reactivated:

p Aps_WatchOn() (reactivate watchdog)

8.2 PROFIBUS DP errors and warnings

When the Typ3 osa is started, the DP master software is loaded. Subsequently, the DP master is started. Errors occurring in the process may bring the software PLC to a standstill. These errors are displayed on the Typ3 osa operator interface.

Error number 1868

Text: Error PDP configuration

Explanation: The cause of this error is an aborted download of the PROFIBUS DP software.

Error number 1990

Text: DP master configuration file not found

Explanation: The file is required to describe the PROFIBUS DP users. It must be stored in the root directory or in the user/fep directory of the Typ3.

Error number 2029

Text: Wrong file format of the PDP configuration file.

Error number 2030

Text: No module address in the PROFIBUS DP configuration file.

Warning number 1991

Text: Not all PROFIBUS DP slaves on bus.

A Appendix

A.1 Index

A

Access function
 actual axis values, 4-16
 Analog I/O, 4-14
 Counters, 4-13
 data modules, 4-6
 flag, 4-4
 marker, 4-4
 NC input image, 4-2
 NC output image, 4-2
 timer modules, 4-9
 to the extended interface, 4-15
 Activation via machine parameter, cycle control, 2-4
 Actual axis values, 4-16
 Analog
 input, 4-14
 output, 4-14
 Analog I/O, access function, 4-14
 Analog interface, 4-14
 initialization, 4-14
 Analog voltage
 limit, 4-14
 limitation function, 4-14
 APS, 2-1
 Architecture, Software PLC, 2-1
 Auxiliary functions, 5-1

B

Byte numbers, future NC inputs, Table, 3-3
 Byte numbers, future NC outputs, Table, 3-17
 Byte numbers, machine (Profibus DP), Table, 3-3, 3-17
 Byte numbers, NC inputs, Table, 3-2
 Byte numbers, NC outputs, Table, 3-16

C

Communication structure, APS, KNS, NCS, 2-2
 Counter functions, 4-13
 parameters, 4-13
 Counter module
 decrement, 4-13
 increment, 4-13
 reset, 4-13
 set, 4-13
 Counter on/off, read, 4-13
 Counter status, read, 4-13
 Counters, access function, 4-13
 Counting value, read, 4-13
 Cycle control, 2-3

D

Data module
 creating, 4-7
 creating memory, 4-6
 handles to the administration part, 4-7
 Length of the data range, 4-7
 Read function, 4-8
 Starting address of a data range, 4-7
 supplying handles, 4-6
 write function, 4-8
 Data modules, Access function, 4-6
 Documentation, 1-8

E

EMC Directive, 1-1
 EMERGENCY-STOP devices, 1-6
 Error types, 8-1
 Errors, Watchdog, 8-1
 Errors and warnings, Profibus DP, 8-2
 ESD
 Electrostatic discharge, 1-7
 grounding, 1-7
 workplace, 1-7
 ESD-sensitive components, 1-7
 Extended interface
 access function, 4-15
 Read, 4-15
 Write, 4-15

F

Flag, access function, 4-4
 Floppy disk drive, 1-8

G

Global interface
 inputs, 3-25
 outputs, 3-26
 read, 4-17
 write, 4-17
 Grounding bracelet, 1-7

H

Hard disk, 1-8

I

Inputs
 global interface, 3-25
 NC interface, 3-2
 Inputs area 2, Machine, 3-28
 Installation, programming, Software PLC, 2-1

K

KNS, 2-2
 Kns_AnwenderInit, Initialization, 2-3

Kns_AnwenderMain, PLC program, 2–3

L

Low-Voltage Directive, 1–1

M

Machine interface, 3–27

Marker, access function, 4–4

Measuring activities, 1–6

Modules sensitive to electrostatic discharge. *See* ESD-sensitive components

MSD, 6–1

N

NC → PLC (interface signals)

axis-related byte numbers, 3–18

channel-related byte numbers, 3–22

spindle-related byte numbers, 3–20

NC input image, Access function, 4–2

NC interface

Inputs, 3–2

Outputs, 3–16

NC output image, Access function, 4–2

Ncs functions, 7–1

O

Outputs

global interface, 3–26

NC interface, 3–16

Outputs area 2, Machine, 3–29

P

PLC → NC (extended interface signals)

axis-related byte numbers, 3–6

channel-related byte numbers, 3–14

spindle-related byte numbers, 3–10

PLC → NC (interface signals)

axis-related byte numbers, 3–4

channel-related byte numbers, 3–12

spindle-related byte numbers, 3–8

PLC program, Kns_AnwenderMain, 2–3

Programmable clock, implementation, 4–12

Proper use, 1–1

Q

Qualified personnel, 1–3

R

Release, 1–8

Remaining time, read, 4–11

Run time, cycle control, 2–4

S

Safety instructions, 1–5

Safety markings, 1–4

Signal accessing, 4–1

Signal assignment area 2

Inputs, 3–28

Outputs, 3–29

Software PLC, Architecture, 2–1

Spare parts, 1–7

Start timer

extended on-delay timer, 4–11

extended pulse timer, 4–11

off-delay timer, 4–11

on-delay timer, 4–11

pulse timer, 4–11

T

Table

Byte numbers, future NC inputs, 3–3

Byte numbers, future NC outputs, 3–17

Byte numbers, machine (Profibus DP), 3–3, 3–17

Byte numbers, NC inputs, 3–2

Byte numbers, NC outputs, 3–16

Time increment value, read, 4–11

Time of output, read, 4–14

Timer functions, 4–11

parameters, 4–12

Timer module type

extended on-delay timer, 4–10

extended pulse timer, 4–9

off-delay timer, 4–10

on-delay timer, 4–10

pulse timer, 4–9

Timer modules

access function, 4–9

read status, 4–11

Reset, 4–11

types, 4–9

Trademarks, 1–9

W

Watchdog, 8–1

Bosch Automation Technology

Australia

Robert Bosch (Australia) Pty. Ltd.
Head Office
Cnr. Centre - McNaughton Roads
P.O. Box 66
AUS-3168 Clayton, Victoria
Fax (03) 95 41 77 03

Great Britain

Robert Bosch Limited
Automation Technology Division
Meridian South Meridian Business Park
GB-LE3 2WY Braunstone
Leicestershire
Fax (01 16) 28-9 28 78

Canada

Robert Bosch Corporation
Automation Technology Division
6811 Century Avenue
CAN-Mississauga, Ontario L5N 1R1
Fax (905) 5 42-42 81

USA

Robert Bosch Corporation
Automation Technology Division
Fluid Power Products
7505 Durand Avenue
USA-Racine, Wisconsin 53406
Fax (414) 5 54-81 03

Robert Bosch Corporation
Automation Technology Division
Factory Automation Products
816 East Third Street
USA-Buchanan, MI 49107
Fax (616) 6 95-53 63

Robert Bosch Corporation
Automation Technology Division
Industrial Electronic Products
40 Darling Drive
USA-Avon, CT 0 60 01-42 17
Fax (860) 4 09-70 80

We reserve the right to make technical alterations

Your concessionary

BOSCH



Robert Bosch GmbH
Geschäftsbereich
Automationstechnik
Antriebs- und Steuerungstechnik
Postfach 11 62
D-64701 Erbach
Fax +49 (0) 60 62 78-4 28