Industrial Hydraulics | Electric Drives and Controls | Linear Motion and Assembly Technologies | Pneumatics | Service Automation | Mobile Hydraulics

Rexroth
Bosch Group

Rexroth PNC V10.x
PLC Interface

R911307969
Edition 01

Project planning

Title

Rexroth PNC V10.x
PLC Interface

Type of Documentation

Project planning

Document Typecode

DOK-PNC***-PLC*INT*V10-PR01-EN-P

Internal File Reference

Document number 120-2600-B302-01/EN

Purpose of Documentation

This manual describes the interface signals and the program modules for the integrated PLC.

Record of Revisions

Description	Release Date	Notes
DOK-PNC***-PLC*INT*V10-PR01-EN-P	08.2004	Valid for V10.x

Copyright

Validity

The data specified above only serve to describe the product. No statements concerning a certain condition or suitability for a certain application can be derived from our information. The given information does not release the user from the obligation of own judgement and verification. It must be remembered that our products are subject to a natural process of wear and aging.

Published by

Bosch Rexroth AG
Electric Drives and Controls
Postfach 13 57
D-97803 Lohr
Bgm.-Dr.-Nebel-Str. 2
D-97816 Lohr
Tel.: +49 (0) 93 52/40-0
Fax: +49 (0) 93 52/40-48 85
Abt.: BRC/ESM11 (WE)

Contents

Page

Contents

Contents

Notes:

1 Safety Instructions

Please read this manual before commissioning the integrated PLC. Store this manual in a place to which all users have access at any time.

1.1 Intended use

This manual contains all information required for the proper use of the control units. For reasons of clarity, however, it cannot contain each and every detail about each and all combinations of functions. Likewise, it is impossible to consider each and any aspect of integration or operation.

The PNC control 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 process a workpiece (CNC). Furthermore, I/O components are required for the integrated PLC which – in communication with the actual CNC – controls the machine processing cycles holistically and acts as a technical safety monitor.
- program contours and the processing 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 normally pose no danger to persons or property if they are used in accordance with the handling stipulations and safety notes prescribed for their configuration, mounting, 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, 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 the following warning is attached:

☞ **This is a Class A device. In a residential area, this device may cause radio interference. In such case, the user may be required to introduce suitable countermeasures, and to bear the cost of the same.**

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

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.

This manual is designed for **NC project engineers and commissioning personnel** They require specialized knowledge of

- methods for configuring the PNC and
- ways of adapting it to a machine environment.

Programming, start and operation as well as the modification of program parameters is reserved to 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 specialized Rexroth 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 damage to property.

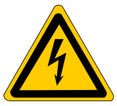
Only electrotechnicians as recognized under IEV 826-09-01 (modified) who are familiar with the contents of this manual may install and service the products described.

Such personnel are

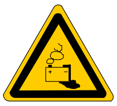
- those who, being well trained and experienced in their field and familiar with the relevant norms, are able to analyze the jobs being carried out and recognize any hazards which may have arisen.
- those who have acquired the same amount of expert knowledge through years of experience that would normally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

With regard to the foregoing, please note our comprehensive range of training courses. Please visit our website at www.boschrexroth.com for the latest information concerning training courses, teachware and training systems. Personal information is available from our Didactic Center Erbach, Telephone: (+49) (0) 60 62 78-600.

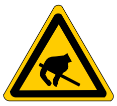
1.3 Safety markings on products



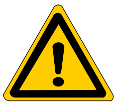
Warning of dangerous electrical voltage!



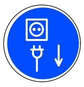
Warning of danger caused by batteries!



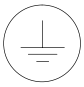
Electrostatically sensitive components!



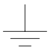
Warning of hazardous light emissions
(optical fiber cable emissions)!



Disconnect mains power before opening!



Lug 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 injury**.

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 for the described product

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 offers the possibility of inhibiting axis movements and/or auxiliary function outputs by appropriate softkeys in the ’Automatic’ mode.

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

Furthermore, Rexroth 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 injury, damage to equipment, or environmental hazards. Possible retrofits or modifications to the system using third-party equipment therefore have to be approved by Rexroth.

DANGER
Do not look directly into the LEDs in the optical fiber connection. Due to their high output, this may result in eye injuries.
When the inverter is switched on, do not look into the LED or the open end of a short connected lead.

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 an active machine tool!**
- **Never deactivate safety-relevant functions!**
- **Report any malfunction of the unit to your servicing and repairs department immediately!**

CAUTION
use only spare parts approved by Rexroth!

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 MΩ 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

Overview of available documentation	Part no.			
	German	English	French	Italian
PNC-R – Connectivity Manual for project engineering and maintenance	1070073 704	1070073 736	–	–
PNC-R – Software installation	1070073 796	1070 073 797	–	–
PNC-P – Connectivity Manual	1070073 880	1070 073 881	–	–
PNC-P – BF2xxT/BF3xxT Control Panel Connectivity Manual	1070073 814	1070073 824	–	–
PNC-P – Software installation	1070073 882	1070 073 883	–	–
Description of functions	1070073 870	1070073 871	–	–
MACODA Operation and configuration of the machine parameters	1070073 705	1070073 742	–	–
Operating instructions – Standard operator interface	1070073 726	1070073 739	1070073 876	1070073887
Operating instructions – Diagnostics Tools	1070073 779	1070073 780	–	–
Error Messages	1070073 798	1070073 799	–	–
PLC project planning manual (V8.x), Software interfaces of the integrated PLC	1070073 728	1070073 741	–	–
PLC project planning manual (V10.x, PNC-P only), Software interfaces of the integrated PLC	R911307968	R911307969	–	–
iPCL system description and programming manual	1070073 874	1070073 875	–	–
ICL700 system description (PNC-R only), Program structure of the integrated PLC ICL700	1070073 706	1070073 737	–	–
DIN programming manual for programming to DIN 66025	1070073 725	1070073 738	1070073888	1070073886
CPL programming manual	1070073 727	1070073 740	1070073 877	1070073885
CPL Debugger Operating Instructions	1070073 872	–	–	–
Tool Management – Parameterization	1070073 782	1070 073 793	–	–
Drive oscilloscope, Operation and programming	1070073 878	–	–	–
Software PLC Development environment for Windows NT	1070073 783	1070073 792	–	–
Measuring cycles for touch-trigger switching probes	1070073 788	1070073 789	–	–
Universal Milling Cycles	–	1070073 795	–	–

Safety Instructions

Release

☞ **This manual refers to the following versions:**
Software release: V10.x

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

The software version of Windows may be displayed as follows:

1. Click the right mouse button on the My Computer icon on your desktop.
2. Select Properties.

1.7 Trademarks

Trademarks

All trademarks of software installed on Rexroth 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 Rexroth or in accordance with the license agreement of the respective manufacturer.

PROFIBUS® is a registered trademark of the PROFIBUS Nutzerorganisation e.V. (user organization).

SERCOS interface™ is a registered trademark of Interessengemeinschaft SERCOS interface e.V. (Joint VDW/ZVEI Working Committee).

Notes:

2 Communication structure and interfaces

2.1 Overview

The PNC control units enable users to interact and integrate their own machine-specific developments in various function areas of the control unit.

- **BOF** (user interface)
- **iPCL** (**i**ntegrated **P**ersonel **C**omputer and **C**ontrol **L**ogic)
- **ACB** (axis movement)
- **IPO** (**I**nter**PO**lation)
- **DB** (**D**ata**B**ase)

This manual describes software interfaces and their functioning in terms of the **PCL integrated** into the NC. These can be:

- iPCL or
- KNS in the developer environment of PNC-P .

☞ **A detailed description of the respective integrated PLC is given in further manuals; see page 1–8.**

The PLC sequential program is principally capable of communicating with all function areas and subsystems of the overall system.

The subsystems of the NC control, e.g. the man-machine interface (MMI) provide services for other subsystems (actual value display, status information, etc.).

These services can be called up by the PLC sequential program:

- via the NCS
- using APS modules.

2.2 The NCS (NC interface)

The **NC interface** (NCS) is the communication basis of the PNC and links the internal and external computer systems. It contains functions for information and data exchange among the control subsystems according to the client/server principle.

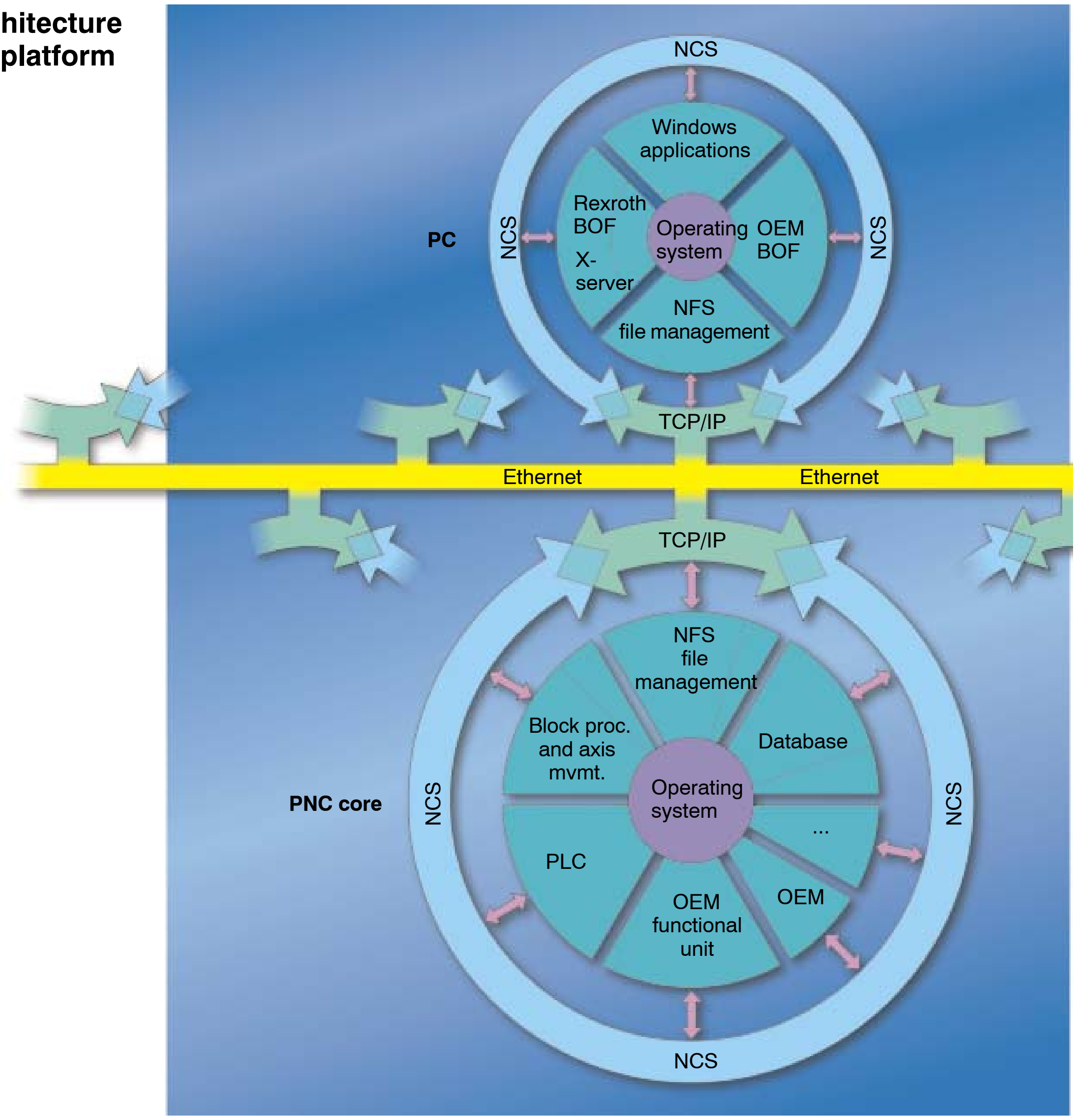
The standard **Ethernet coupling** (via TCP/IP) allows further control units and PCs to be directly connected to the control unit core via NFS file management as well as the integration of the system into networks.

Communication structure and interfaces

The following areas communicate via the NC interface (NCS) and via the TCP/IP:

- **PNC**: iPCL, block processing and axis movement, data base, OEM functional units possible.
- **PC**: Rexroth MMI, OEM MMI, Windows applications, NFS file management)

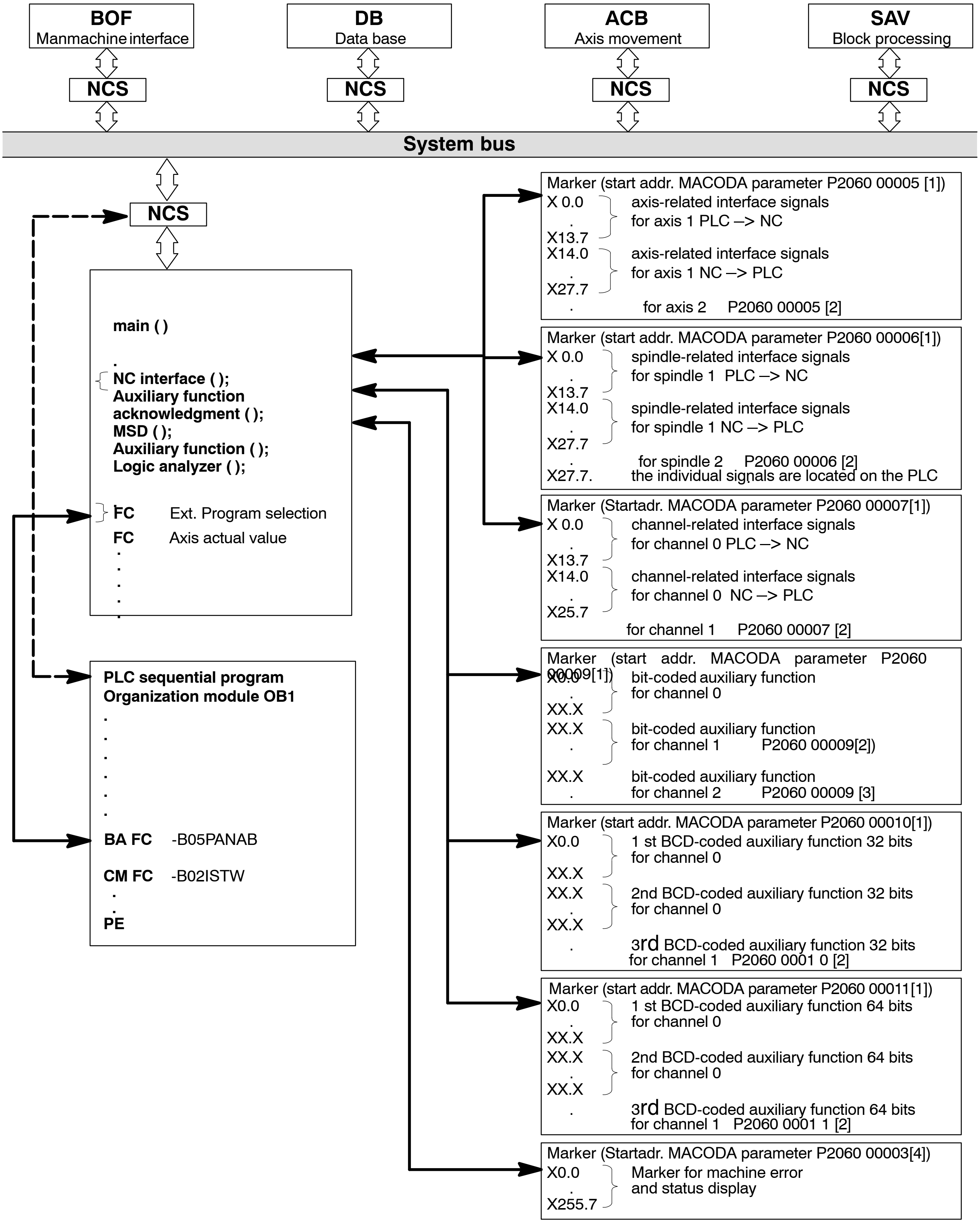
System architecture of the PNC platform



☞ **Data exchange to the PNC-P can only be performed via APS modules (not via NCS) at present.**

Communication structure and interfaces

Example of NCS communication structure in the iPCL
(for KNS communication structure, see ”Software PLC development environment” manual)


BOF
Manmachine interface
DB
Data base
ACB
Axis movement
SAV
Block processing
NCS
NCS
NCS
NCS
System bus
NCS
main ( )
.
NC interface ( );
Auxiliary function
acknowledgment ( );
MSD ( );
Auxiliary function ( );
Logic analyzer ( );
FC Ext. Program selection
FC Axis actual value
PLC sequential program
Organization module OB1
BA FC -B05PANAB
CM FC -B02ISTW
PE
Marker (start addr. MACODA parameter P2060 00005 [1])
X 0.0 . X13.7 axis-related interface signals for axis 1 PLC –> NC
X14.0 . X27.7 axis-related interface signals for axis 1 NC –> PLC
. for axis 2 P2060 00005 [2]
Marker (start addr. MACODA parameter P2060 00006[1])
X 0.0 . X13.7 spindle-related interface signals for spindle 1 PLC –> NC
X14.0 . X27.7 spindle-related interface signals for spindle 1 NC –> PLC
. for spindle 2 P2060 00006 [2]
X27.7. the individual signals are located on the PLC
Marker (Startadr. MACODA parameter P2060 00007[1])
X 0.0 . X13.7 channel-related interface signals for channel 0 PLC –> NC
X14.0 . X25.7 channel-related interface signals for channel 0 NC –> PLC
for channel 1 P2060 00007 [2]
Marker (start addr. MACODA parameter P2060 00009[1])
X0.0 . XX.X bit-coded auxiliary function for channel 0
XX.X . bit-coded auxiliary function for channel 1 P2060 00009[2])
XX.X . bit-coded auxiliary function for channel 2 P2060 00009 [3]
Marker (start addr. MACODA parameter P2060 00010[1])
X0.0 . XX.X 1 st BCD-coded auxiliary function 32 bits for channel 0
XX.X . XX.X 2nd BCD-coded auxiliary function 32 bits for channel 0
. 3rd BCD-coded auxiliary function 32 bits for channel 1 P2060 0001 0 [2]
Marker (start addr. MACODA parameter P2060 00011[1])
X0.0 . XX.X 1 st BCD-coded auxiliary function 64 bits for channel 0
XX.X . XX.X 2nd BCD-coded auxiliary function 64 bits for channel 0
. 3rd BCD-coded auxiliary function 64 bits for channel 1 P2060 0001 1 [2]
Marker (Startadr. MACODA parameter P2060 00003[4])
X0.0 . X255.7 Marker for machine error and status display

2.3 The APS (APplication software)

The **APS** is a software package developed by Rexroth.

The APS has the task of simplifying the data exchange for the PLC project engineer with the **subsystems of the control unit** and providing relevant data for the PLC sequential program in the form of markers, data modules or other operands to the PLC (iPCL), or transferring data contained in these operands to the NC.
The APS is continuously upgraded by Rexroth and expanded with other useful functions for the user.

The APS utilizes the functions of the NCS in order to perform a data exchange with the subsystems of the control unit.

The APS provides the NC output signals in markers and transfers the NC input signals, MSD signals and signals of the logic analyzer to the corresponding function areas in the control unit.
The APS also contains a number of program modules to be parameterized by the user which – after being called up by the PLC sequential program – select an NC program, read axis actual values or program statuses and transfer them to the iPCL. The program modules are fully accessible, facilitating the creation of the PLC sequential program by simple parameterization and calling (BA –FC1, etc.).

☞ **The program examples shown in the context of APS program module descriptions show the PROFI software syntax. For PLC user programs written with WinSPS, this syntax must be adapted correspondingly.**

3 Global Interface Signals

3.1 Overview of global interface signals

Global interface signals are higher-level signals used with functions relating to the NC as a whole.
The start address is defined in MACODA parameter 2060 00008.

☞ **When –1 is the start address, the global interface is not evaluated (= preset value of MACODA parameter 2060 00008).**

3.1.1 Overview of output signals (PLC –> NC)

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
0.0	qGen_Reset	System control reset	1.0	qGen_StrokeInhibit	Stroke inhibit
0.1	qGen_EditInhibit	Edit inhibit	1.1	qGen_StrokeReserv	Stroke reservation
0.2	–	res.	1.2	qGen_StrokeRel	Stroke on
0.3	–	res.	1.3	–	res.
0.4	–	res.	1.4	–	res.
0.5	–	res.	1.5	–	res.
0.6	–	res.	1.6	–	res.
0.7	–	res.	1.7	–	res.
2.0	–	res.	3.0	–	res.
2.1	–	res.	3.1	–	res.
2.2	–	res.	3.2	–	res.
2.3	–	res.	3.3	–	res.
2.4	–	res.	3.4	–	res.
2.5	–	res.	3.5	–	res.
2.6	–	res.	3.6	–	res.
2.7	–	res.	3.7	–	res.

3.1.2 Overview of input signals (NC –> PLC)

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
0.0	–	res.	1.0	iGen_StrokeIntend	Stroke intended
0.1	–	res.	1.1	iGen_NoStroke	Stroke is not running
0.2	–	res.	1.2	–	res.
0.3	–	res.	1.3	–	res.
0.4	–	res.	1.4	–	res.
0.5	–	res.	1.5	–	res.
0.6	–	res.	1.6	–	res.
0.7	–	res.	1.7	–	res.
2.0	–	res.	3.0	–	res.
2.1	–	res.	3.1	–	res.
2.2	–	res.	3.2	–	res.
2.3	–	res.	3.3	–	res.
2.4	–	res.	3.4	–	res.
2.5	–	res.	3.5	–	res.
2.6	–	res.	3.6	–	res.
2.7	–	res.	3.7	–	res.

3.2 Signal description

3.2.1 Output signals (PLC –> NC)

System control reset qGen_Reset

This signal acts on all channels of the NC. Its effect on the individual channels is identical with a control reset of each single channel.

System reset affects:

- axis-related signals of the NC to the PLC
- channel-related signals of the NC:
  - Control reset executed
  - NC ready
  - Program running

Signal level

1: A system reset is effected on **all** channels.

0: No effect.

Edit inhibit qGen_EditInhibit

Edit inhibit prohibits the user from performing certain operations on the NC. Only user actions directly related with the production process are allowed.

The effect of the Edit inhibit signal is channel-independent.

Signal level

1: The following functions are either locked or restricted:
- file management
- tables
- editor
- MACODA

0: No effect.

CAUTION
Edit inhibit does not act on tool management.
If required, tool management can be locked via the PLC (see manual ”Tool Management – Parameterization”).

Overview of user levels to which edit inhibit applies:

User level	Edit inhibit applicable
System	No
Developer	No
MTB	No
Setter	No
User	Yes

☞ **When activated, Edit inhibit affects the “lowest user level” only.**

For further information on edit inhibit, see the manual ”Operating instructions, Standard operator interface”.

Stroke inhibit qGen_StrokeInhibit

With this signal, the PLC inhibits the triggering of a punching stroke, i.e. the corresponding fast output on the High Speed I/O plug-in card is locked.

Signal level

1: The high-speed output HSO is disabled.

0: No effect.

Stroke reservation qGen_StrokeReserv

With this signal, the PLC reserves a high-speed output (HSO) on the High Speed I/O plug-in card for its own stroke trigger (see the figure on the following page).

Signal level

1: The high-speed output HSO is reserved for the PLC.

0: No effect.

Stroke on qGen_StrokeRel

With the **Stroke on** signal, the PLC instructs the NC to trigger a punching stroke via the high-speed output HSO on the High Speed I/O plug-in card.

Signal edge/level

0 → 1: Punching stroke to be triggered by the PLC via the high-speed output HSO.

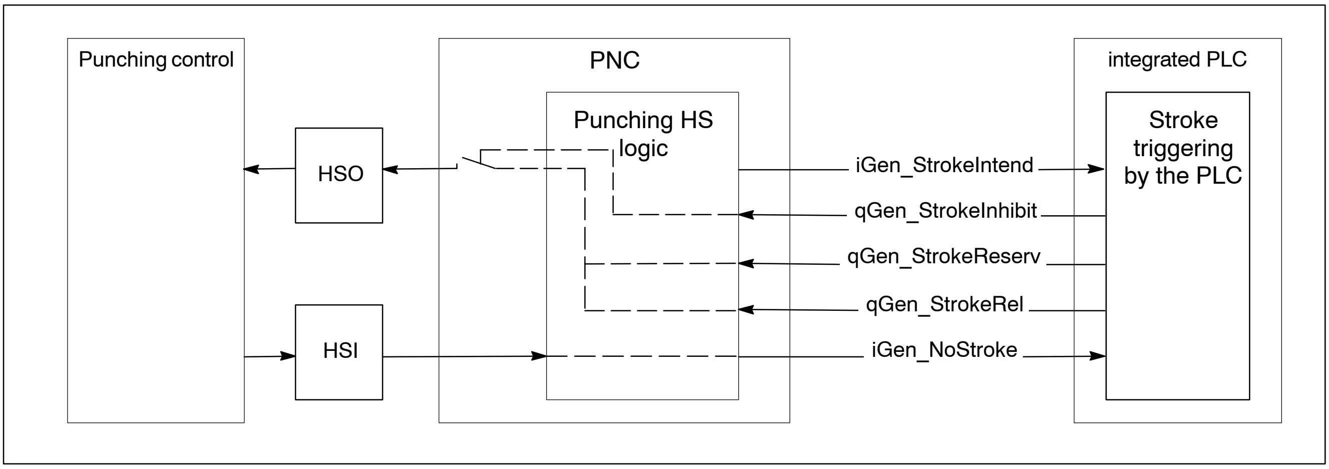
0: No effect.

Punching HS logic

Individual strokes can be triggered by the PLC.
The PLC can trigger a stroke by instructing the NC to do so via interface signals.

The bit signals used in the communication among the NC, the punching control and the PLC are shown in the following figure:

Global Interface Signals



The punching HS logic can process five interface signals:

iGen_StrokeIntend ”Stroke intended”
The NC communicates to the PLC that the NC wants to trigger a stroke.

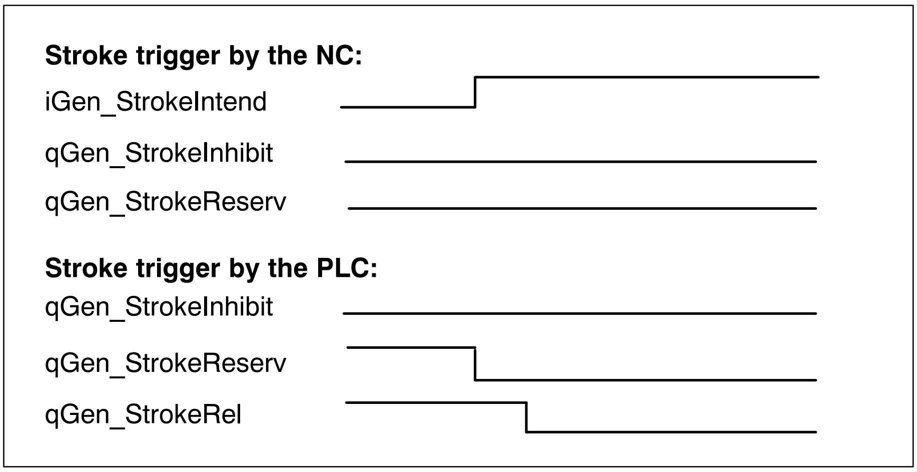
iGen_NoStroke ”Stroke is not running”
The high-speed input (HSI) is passed on to the PLC by the punching HS logic.

qGen_StrokeInhibit ”Stroke inhibit”
The PLC inhibits the HSO from being set.

qGen_StrokeReserv ”Stroke reservation”
The PLC reserves the high-speed output (HSO) for its own stroke triggering.

qGen_StrokeRel ”Stroke on”
The PLC instructs the NC to trigger a stroke.

A punching stroke can be triggered both from the NC and from the PLC:



3.2.2 Input signals (NC –> PLC)

Stroke intended iGen_StrokeIntend

With the **Stroke intended** signal, the NC communicates to the PLC that the NC wants to trigger a punching stroke. If the PLC permits the stroke to be triggered (qGen_StrokeInhibit = 0 and qGen_StrokeReserv = 0), the high-speed output (HSO) on the High Speed I/O plug-in card is set by the punching HS logic (see also Chapter 3.2.1).

Signal level

1: The NC communicates to the PLC that the NC wants to trigger a stroke.

0: No effect.

Stroke is not running iGen_NoStroke

The response signal **Stroke is not running** from the punching control (high-speed input HSI on the High Speed I/O plug-in card) is relayed to the PLC.

Signal level

1: Stroke is not running = response signal from the punching control to the PLC.

0: Stroke running.

4 Channel-Related Interface Signals

4.1 Overview of Channel-Related Interface Signals

The **channel structure** of the PNC is designed to allow for several tasks to be processed simultaneously.
Channel-related interface signals thus mainly affect NC functions which are related to the execution of the NC part program.

Every channel has its own interface; i.e. with n channels, the interface depicted in sections 4.1.1 and 4.1.2 is mapped n times on the corresponding markers.
In addition to the channel-related interface, the axis-related interface is of significance to the machining axes combined to form a channel as well as to the auxiliary axes.

The start addresses for the channel-related interface signals are defined in MACODA parameter 2060 00007.

There must always be at least 2 channels entered:

- channel 0 for switching functions
- channel 1 for a machining channel

The following description shows the interface signals for a channel.

4.1.1 Overview of output signals (PLC –> NC)

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
0.0	qCh_OpModeSel_00	Sel. mode, bit 0	1.0	qCh_OpModePlc	PLC mode
0.1	qCh_OpModeSel_01	Sel. mode, bit 1	1.1	qCh_Restart	Automatic restart
0.2	qCh_OpModeSel_02	Sel. mode, bit 2	1.2	qCh_NCStart	NC start
0.3	qCh_OpModeSel_03	Sel. mode, bit 3	1.3	qCh_TransferLock	Block transfer inhibit
0.4	–	res.	1.4	qCh_FeedHold	Feed hold
0.5	–	res.	1.5	qCh_FeedStop	Feed inhibit
0.6	–	res.	1.6	–	res.
0.7	–	res.	1.7	qCh_ReSelOff	Autom. reselection off
2.0	qCh_CtrlReset	Control reset	3.0	qCh_ASub1	Asynchr.-subrout. 1
2.1	–	res.	3.1	qCh_ASub2	Asynchr.-subrout. 2
2.2	qCh_CancDist	Cancel distance to go	3.2	qCh_ASub3	Asynchr.-subrout. 3
2.3	qCh_NextBlk	Switch to next block	3.3	qCh_ASub4	Asynchr.-subrout. 4
2.4	–	res.	3.4	qCh_ASub5	Asynchr.-subrout. 5
2.5	–	res.	3.5	qCh_ASub6	Asynchr.-subrout. 6
2.6	qCh_RetCont	Restarting	3.6	qCh_ASub7	Asynchr.-subrout. 7
2.7	qCh_Retract	Rapid retraction	3.7	qCh_ASub8	Asynchr.-subrout. 8

Channel-Related Interface Signals

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
4.0	qCh_JogPlusWcs	WCS manual +	5.0	qCh_BlkSlash	Block skip
4.1	qCh_JogMinusWcs	WCS manual –	5.1	qCh_OptStop	Optional stop
4.2	–	res.	5.2	qCh_OptJump	Conditional jump
4.3	–	res.	5.3	–	res.
4.4	–	res.	5.4	qCh_RedRap	Limit rapid travel
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	–	res.
4.7	–	res.	5.7	qCh_Override100	Override 100%
6.0	qCh_Override_00	Override bit 0	7.0	qCh_Override_08	Override bit 8
6.1	qCh_Override_01	Override bit 1	7.1	qCh_Override_09	Override bit 9
6.2	qCh_Override_02	Override bit 2	7.2	qCh_Override_10	Override bit 10)
6.3	qCh_Override_03	Override bit 3	7.3	qCh_Override_11	Override bit 11
6.4	qCh_Override_04	Override bit 4	7.4	qCh_Override_12	Override bit 12
6.5	qCh_Override_05	Override bit 5	7.5	qCh_Override_13	Override bit 13
6.6	qCh_Override_06	Override bit 6	7.6	qCh_Override_14	Override bit 14
6.7	qCh_Override_07	Override bit 7	7.7	qCh_Override_15	Override bit 15
8.0	qCh_Custom1	Customer input 1	9.0	qCh_OnlCorrWcs	Online correction enable
8.1	qCh_Custom2	Customer input 2	9.1	qCh_OnlCorrWcsDir	Online correction direction
8.2	qCh_Custom3	Customer input 3	9.2	–	res.
8.3	qCh_Custom4	Customer input 4	9.3	–	res.
8.4	qCh_Custom5	Customer input 5	9.4	–	res.
8.5	qCh_Custom6	Customer input 6	9.5	–	res.
8.6	qCh_Custom7	Customer input 7	9.6	qCh_CoordCoupleOff	Terminate coupling
8.7	qCh_Custom8	Customer input 8	9.7	qCh_TangTRotRel	G131 release
10.0	–	res.	11.0	–	res.
10.1	–	res.	11.1	–	res.
10.2	–	res.	11.2	–	res.
10.3	–	res.	11.3	–	res.
10.4	–	res.	11.4	–	res.
10.5	–	res.	11.5	–	res.
10.6	–	res.	11.6	–	res.
10.7	–	res.	11.7	–	res.
12.0	–	res.	13.0	–	res.
12.1	–	res.	13.1	–	res.
12.2	–	res.	13.2	–	res.
12.3	–	res.	13.3	–	res.
12.4	–	res.	13.4	–	res.
12.5	–	res.	13.5	–	res.
12.6	–	res.	13.6	–	res.
12.7	–	res.	13.7	–	res.

4.1.2 Overview of input signals (NC –> PLC)

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
0.0	iCh_OpMode_00	Sel. mode, bit 0	1.0	iCh_DryRun	Test mode
0.1	iCh_OpMode_01	Sel. mode, bit 1	1.1	iCh_NCReady	NC ready
0.2	iCh_OpMode_02	Sel. mode, bit 2	1.2	iCh_ProgRun	Program running
0.3	iCh_OpMode_03	Sel. mode, bit 3	1.3	iCh_TransferLockAct	Block transfer inhibit active
0.4		res.	1.4	iCh_FeedHoldAct	Feed hold active
0.5		res.	1.5	–	res.
0.6		res.	1.6	iCh_ProgStopM0	Program stop M0
0.7		res.	1.7	iCh_ProgStopM30	Program end M30
2.0	iCh_Reset	Channel reset	3.0	iCh_ASub1	Asynchr.-subrout. 1
2.1	–	res.	3.1	iCh_ASub2	Asynchr.-subrout. 2
2.2	–	res.	3.2	iCh_ASub3	Asynchr.-subrout. 3
2.3	–	res.	3.3	iCh_ASub4	Asynchr.-subrout. 4
2.4	iCh_RemoveFinish	Remove finished	3.4	iCh_ASub5	Asynchr.-subrout. 5
2.5	iCh_ReadyReEnter	Ready for re-entry	3.5	iCh_ASub6	Asynchr.-subrout. 6
2.6	iCh_ReEnterAct	Re-entry active	3.6	iCh_ASub7	Asynchr.-subrout. 7
2.7	–	res.	3.7	iCh_ASub8	Asynchr.-subrout. 8
4.0	iCh_State_00	Channel status bit 0	5.0	iCh_BlkSlash	Activate block skip
4.1	iCh_State_01	Channel status bit 1	5.1	iCh_OptStop	Optional stop activated
4.2	iCh_State_02	Channel status bit 2	5.2	–	res.
4.3	iCh_State_03	Channel status bit 3	5.3	–	res.
4.4	iCh_State_04	Channel status bit 4	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	iCh_Override0	Override 0%
4.7	–	res.	5.7	iCh_Override100	Override 100%
6.0	iCh_Cpl01	CPL customer output 1	7.0	iCh_Cpl09	CPL customer output 9
6.1	iCh_Cpl02	CPL customer output 2	7.1	iCh_Cpl10	CPL customer output 10
6.2	iCh_Cpl03	CPL customer output 3	7.2	iCh_Cpl11	CPL customer output 11
6.3	iCh_Cpl04	CPL customer output 4	7.3	iCh_Cpl12	CPL customer output 12
6.4	iCh_Cpl05	CPL customer output 5	7.4	iCh_Cpl13	CPL customer output 13
6.5	iCh_Cpl06	CPL customer output 6	7.5	iCh_Cpl14	CPL customer output 14
6.6	iCh_Cpl07	CPL customer output 7	7.6	iCh_Cpl15	CPL customer output 15
6.7	iCh_Cpl08	CPL customer output 8	7.7	iCh_Cpl16	CPL customer output 16
8.0	iCh_Custom1	Customer output 1	9.0	iCh_G0Act	Rapid traverse active
8.1	iCh_Custom2	Customer output 2	9.1	iCh_InPosAct	In-pos range 2 active
8.2	iCh_Custom3	Customer output 3	9.2	iCh_G41G141Act	G41/G141 active
8.3	iCh_Custom4	Customer output 4	9.3	iCh_G42G142Act	G42/G142 active
8.4	iCh_Custom5	Customer output 5	9.4	–	res.
8.5	iCh_Custom6	Customer output 6	9.5	–	res.
8.6	iCh_Custom7	Customer output 7	9.6	iCh_CoordCoupleAct	Coord. coupling active
8.7	iCh_Custom8	Customer output 8	9.7	iCh_TangTRotCmd	Tool turn (TangTool)

Channel-Related Interface Signals

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
10.0	iCh_ActFunc01	G70 active	11.0	iCh_ActFunc09	Tool compensation active bit 0
10.1	iCh_ActFunc02	Feed 100% active	11.1	iCh_ActFunc10	Tool compensation active bit 1
10.2	iCh_ActFunc03	not assigned	11.2	iCh_ActFunc11	Tool compensation active bit 2
10.3	iCh_ActFunc04	not assigned	11.3	iCh_ActFunc12	Tool compensation active bit 3
10.4	iCh_ActFunc05	G92 active	11.4	iCh_ActFunc13	Tool compensation active bit 4
10.5	iCh_ActFunc06	Thread cycle active	11.5	iCh_ActFunc14	Tool compensation active bit 5
10.6	iCh_ActFunc07	Tapping without compensating chuck active	11.6	iCh_ActFunc15	not assigned
10.7	iCh_ActFunc08	Thread cutting active	11.7	iCh_ActFunc16	not assigned
12.0	iCh_ActFunc17	not assigned	13.0	–	res.
12.1	iCh_ActFunc18	G96 active	13.1	–	res.
12.2	iCh_ActFunc19	not assigned	13.2	–	res.
12.3	iCh_ActFunc20	not assigned	13.3	–	res.
12.4	iCh_ActFunc21	not assigned	13.4	–	res.
12.5	iCh_ActFunc22	not assigned	13.5	–	res.
12.6	iCh_ActFunc23	not assigned	13.6	–	res.
12.7	iCh_ActFunc24	not assigned	13.7	–	res.

4.2 Signal description

4.2.1 Output signals (PLC –> NC)

Mode selection qCh_OpModeSel_00 – 03

The operating mode can be entered using either the PLC or the NC user interface (see qCh_OpModePlc, page 4–6).

The following operating modes are possible:

Mode	**Mode selection**			
	Bit 3 (qCh_ OpModeSel_03)	**Bit 2 (qCh_ OpModeSel_02)**	**Bit 1 (qCh_ OpModeSel_01)**	**Bit 0 (qCh_ OpModeSel_00)**
No operating mode	0	0	0	0
Manual set-up	0	0	0	1
Manual traverse to reference point	0	0	1	0
Reserved	0	0	1	1
Manual NC block entry	0	1	0	0
Next block	0	1	0	1
Program block	0	1	1	0
Single step	0	1	1	1
Reserved	1	0	0	0
Reserved	1	0	0	1
Single block	1	0	1	0
Restarting	1	0	1	1
CPL debugger program block	1	1	0	0
CPL debugger, next block	1	1	0	1
Manual set-up, workpiece coordinates	1	1	1	0
Reserved	1	1	1	1

Switching from one operating mode to another is possible under the following conditions:

Mode		**Mode switch, channel active**		**Mode switch, channel inactive** (iCh_Reset = 1, channel reset)
Next block	Automatic mode, group 1	yes, within group 1	yes, within groups 1 and 2: if channel status is ready for program start, i.e. at the beginning or end of a program	yes
Single step				yes
Single block				yes
Program block	Automatic mode, group 2	yes, within group 2		yes
CPL debugger program block				yes
CPL debugger next block				yes
Manual NC block entry		yes, with a control reset executed for the channel prior to switching		yes
Restarting				yes
Manual set-up				yes
Manual traverse to reference point				yes
Manual set-up, workpiece coordinates				yes

PLC mode qCh_OpModePlc

The selection of operating modes can be carried out manually via the NC user interface or via the PLC sequential program.

Signal level

1: The mode is prescribed by the PLC using **qCh_OpModeSel_00 – 03**. As long as signal level = 1 is set, it is not possible to preselect an operating mode via the NC user interface.

0: The operating mode is entered via the **NC operating interface**. As long as signal level = 0 is set, it is not possible to specify the operating mode via the PLC.

Automatic restart qCh_Restart

Signal level

1: When a program has come to an end, it is restarted automatically.

0: No automatic program reselection.

NC start qCh_NCStart

Affects the operating modes **Next block**, **Single block, Single step, Program block** and **MDI NC block**.
Input signals NC ready, Feed hold active and Program running are significant in conjunction with the NC start.

Signal edge

0 → 1: Starts the execution of a part program (next block) or an NC block. **Feed hold active** is reset and the axis movement is initiated again.

1 → 0: No effect.

Block transfer inhibit qCh_TransferLock

Affects the operating modes **Next block**, **Single block, Single step** and **MDINC block**.

Signal level

1: Execution of the next NC block is inhibited. However, execution of the active block continues. Upon completion of the active block, the signal **Block transfer inhibit active** is set. The part program / the entered NC block cannot be executed with the NC start while the block transfer inhibit is set. The Block transfer inhibit signal will not stop block preparation, which continues to run regardless until the memory space for block preparation is allocated.

0: The processing of an NC part program interrupted by the Block transfer inhibit is continued.

Feed hold qCh_FeedHold

Affects **all machining axes** of a channel**.**

Feed hold has no effect on NC blocks in which no movement is programmed, i.e. the execution of the NC program is continued until a machining axis is about to perform a traversing movement.

Feed hold is self-sustaining. Input signal Feed hold active remains set as long as **Feed hold** is active.
Self-sustainment is cancelled by the channel-related interface signal **NC start** .

Signal level

1: Axis motions are slowed to a halt with the deceleration; traversing motions cannot be started again before self-sustainment is cancelled.
Feed hold has **no effect** on thread cutting (G33). In this case, the feed rate is dependent on the spindle speed and the thread pitch.

0: As soon as Feed hold is reset, its self-sustainment can be cancelled by setting **NC start** and the next traversing motion can be started.

Feed inhibit qCh_FeedStop

Acts simultaneously on all machining axes assigned to a channel in operating modes **Next block**, **Single block, Single step** and **NC block-manual data input** (MDI).

This signal has no effect on auxiliary axes since auxiliary axes are not assigned to any particular channel.

Axis-related feed inhibit has priority over the channel-related feed inhibit general signal.

Signal level

1: The axes do not traverse while the signal is set. If the signal is set while the axes are in motion, all axes of the channel are slowed to a halt at the respective path deceleration.
Feed inhibit has **no effect** on thread cutting (G33). The feed rate is dependent on the spindle speed and the thread pitch.

0: When Feed inhibit has been reset, the axes can be put back into movement using **NC start**. They accelerate to the preset speed.

Automatic reselection off qCh_ReSelOff

This signal affects the automatic reselection of a part program.

Signal level

1: Automatic program reselection is suppressed.

0: Automatic reselection permitted.

Control reset qCh_CtrlReset

This signal has an effect on the axes interrelated by interpolation on **one** channel as well as on the part program running on this channel. It has the same effect as the Control reset function, which can be performed using the NC operator interface.

In connection with the Control reset, the channel-related signals **Channel reset**, **NC ready** and Program running are significant. Furthermore, Control reset has an effect on the axis-related signals **Axis in position**, **Axis running** and **Travel command**.

Signal level

1: A reset results in, among other things, the deselection of the NC part program and resetting of NC functions to the power-up state defined in MP 7060 00020. Furthermore, the machining axes are braked to a standstill.

If the Control reset function is terminated from within the control, the NC sets signal **Control reset executed**. Then the PLC resets the signal level to 0.

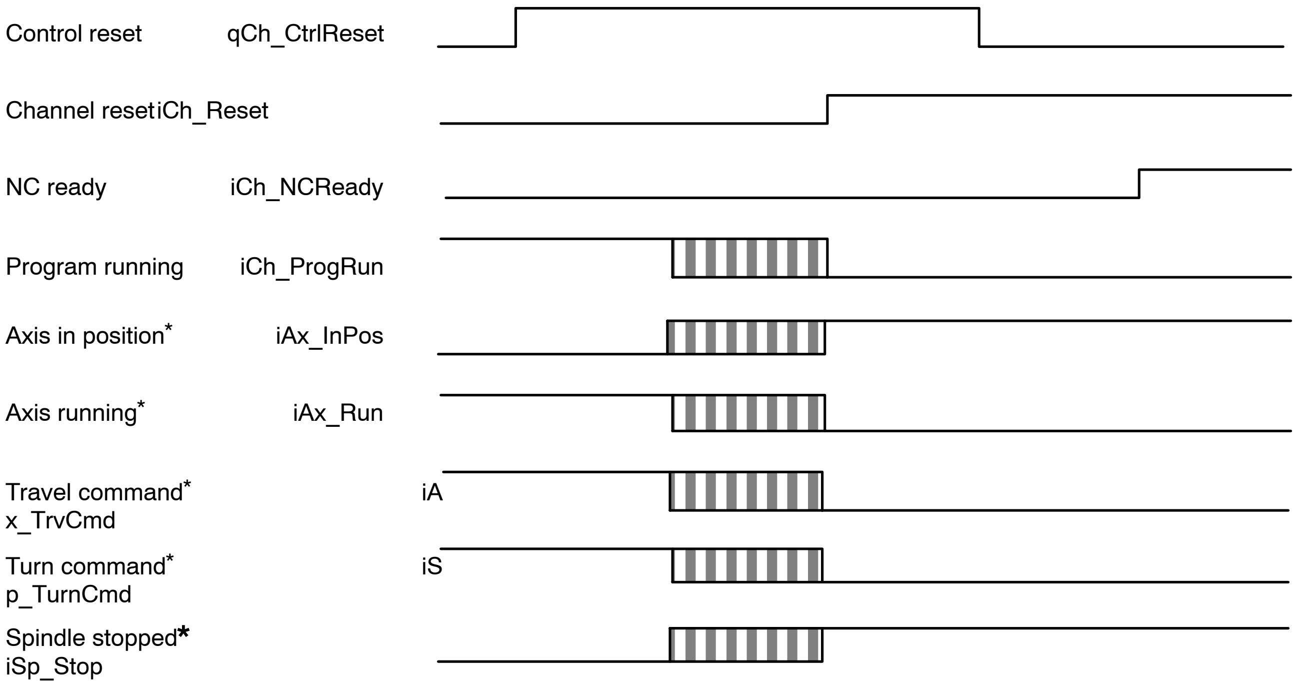
If the signal edges of all channels are simultaneously set from 0 to 1, a general control reset will be performed. This means that, in addition to the individual resetting operations, auxiliary axes and spindles are decelerated until they come to a halt and cross-channel machine parameters are applied.

In the case of thread cutting (G33), **Control reset** triggers a rapid retraction (also see Rapid retraction, page 4–12).

0: No effect.

Example:

Signal **Control reset** is set via the channel-related interface in order to abort the processing of a part program.



*) for auxiliary axes and spindles only with a general control reset.

☞ **The signal NC ready is not set until an NC part program has been selected.**

Cancel distance to go qCh_CancDist

This has an effect in operating modes **Next block, Single block, Single step** and **Program block** – when the program is active – and **NC block manual data input**.

First, the program is stopped using Feed hold (qCh_FeedHold; this is not compulsory). The display shows the distance the axes still have to traverse until the end of the block.

Having initiated ”Cancel distance to go”, all prepared NC blocks, including the remainder of the current block, are cancelled and prepared again.

Please note:

- CPL blocks or CPL parts are not taken into consideration:

  Example:

  CPL variable POS has a value of 10 during preparation: the X[POS] NC word is then interpreted as X10 following ”Cancel distance to go” even though POS may have a totally different value at this time.
- Any changed compensation values are taken into account.

In the display, the end point is set to the current position, at the same time deleting the indicated distance to go. The channel changes to the NC ready state (iCh_NCReady)
Following the NC start (NC-I 1.2), the program is continued at the **point of interruption**, taking into account the new compensation values.

Signal edge

0 → 1: Triggers Cancel distance to go when the program is active.

1 → 0: No effect.

☞ **This signal must not be set when thread cutting is active.**

Switch to next block qCh_NextBlk

When the part program is active, this signal has an effect in operating modes **Next block, Single block, Single step** and **Program block.**

Signal edge

0 → 1: The synchronous axis movement of the channel is stopped with the admissible path deceleration. Subsequently, all prepared NC blocks are cancelled. The channel changes to the **NC ready** state (iCh_NCReady)
After the NC start (qCh_NCStart), the subsequent blocks are prepared again and executed. The part of the interrupted NC block that has not yet been executed is ignored in the process.
To be able to activate the function selectively, it is necessary to previously stop the program at a defined position using Feed hold.

1 → 0: No effect.

☞ **This function generally only works if the aborted block is followed by a linear block (due to the changed starting position, a consistent circle is no longer achieved using, for example, the programmed parameters).**
If cutter path compensation is active, the function usually leads to undesired traversing movements and thus to the destruction of the part to be machined.

Restarting qCh_RetCont

Signal level / edge

1: Following the NC start, the tool returns to contour on the stored departing movement.

0: Machining is continued immediately after the NC start.

0 → 1: Recording of the motion in departing from the contour is stopped.

Rapid retraction qCh_Retract

If thread cutting (G33) is active, this triggers retraction if the cutting movement has been started.

Prerequisite:

The retraction movement is configured via MP 7050 00645 – 7050 00650 or using the function ThreadSet(RD(,)) and is released via MP 7050 00640 or using the function ThreadSet(RON1).

Signal level

1: Retraction from the contour is initiated.
A linear retraction movement is superimposed on the ”second cutting axis”, which moves the cutting tool away from the machined part.

Second cutting axis:

- secondary axis of the plane in case of longitudinal thread, or
- main axis of the plane in case of face thread

0: No effect.

Asynchronous subroutine 1 – 8 qCh_ASub1 – 8

Each of these signals triggers one of a maximum of 8 asynchronous subroutines. The processing of an NC part program is interrupted and the asynchronous subroutine is started immediately. Processing of the NC part program can be continued after the asynchronous subroutine is terminated.

WCS manual + qCh_JogPlusWcs
WCS manual – qCh_JogMinusWcs

These signals are effective in the **Manual set-up workpiece coordinates** operating mode. For a detailed description of this operating mode, please refer to the ”Function Description” manual (for order No., see page 1–8).

Signal edge

0 → 1: When a coordinate is selected in the ”Manual set-up workpiece coordinates” operating mode, the movement of the coordinate is started using Manual + or Manual –.

1 → 0: Manual + or Manual – continuous movement is stopped.

Block skip qCh_BlkSlash

Has an effect in operating modes **Next block, Program block, Single block** and **Single step.**

Signal level

1: At the time of **block preparation**, the NC blocks marked with a slash ”/” in the NC part program are skipped during execution.
Block preparation may be ahead of the active block by several NC blocks. To make sure that the skip is definitely carried out, it is necessary to interrupt block preparation at an appropriate point (prior to the NC block with the slash /), depending on the application. Interruption of block preparation is effected in the part program using the **WAIT** command.

0: The NC block with the slash ”/” is not skipped.

Optional stop qCh_OptStop

Effective in conjunction with function **M1**, which is programmed in the NC part program.

Signal level

1: The execution of the NC part program is stopped after the NC block in which function **M1** has been programmed. It may be restarted with **NC start**.

0: Function **M1** is not effective; the NC part program is not stopped.

Conditional jump qCh_OptJump

Effective in conjunction with function **M1**, which is programmed in the NC part program.

Signal level

1: In the NC part program, a jump is executed to the NC block which is defined as the jump destination.
For this purpose, the signal level logic 1 must be detected already at the **time of block preparation**.

Block preparation may be ahead of the active block by several NC blocks.
To make sure that the jump is actually carried out, it is necessary to interrupt block preparation at an appropriate point (prior to the NC block with the GoCond function), depending on the application. The interruption is effected in the part program using the **WAIT** command.

0: The jump in the part program is not performed.

Limit rapid travel qCh_RedRap

In operating modes **NC block manual data input, Next block, Program block** and **Single step**, G0 rapid travel is limited to a maximum value specified in machine parameter 703000110.

If the reduction is set after the program is started, it becomes effective only after all the blocks already prepared at this time have been executed.
If this is not the required action, machining must be stopped by using the Feed hold function and the prepared blocks must be canceled using ”Cancel distance to go” before machining can be continued using NC start.

Channel override is also valid with the Limit rapid travel function active, i.e. 100% override is exactly equivalent to the limit rapid travel velocity.

The function is effective for all NC functions traversing at rapid travel velocity.
The Limit rapid travel function does not affect the spindle speed or the programmed feed rate.
This function has an effect only on the synchronous axes assigned to the channel.

Signal level

1: All rapid travel blocks are prepared for execution at Limit rapid travel velocity.

0: Limit rapid travel is inactive and all rapid travel blocks are prepared for execution at the maximum travel velocity possible.

Override 100% qCh_Override100

Acts on all axes defined on the channel.

Signal level

1: Cancels the **Override function**; the axes travel with the programmed/preselected speed.

0: The **Override function** is effective.

Override bit 0 – Bit 15 qCh_Override_00 – 15

The traversing speed of all machining axes of a channel can be controlled with the **Override function.** The Override function affects the speeds programmed in the NC part program and on the preselectable speeds in Set-up mode. There are two procedures for this:

- The interface is used to activate a **binary-coded** override switch in the PNC with a maximum of 32 settings.
  Each binary code is assigned a value via MACODA parameter 7030 00010, which represents the respective current speed in per cent.
- If all the settings in MACODA parameter 7030 00010 are 0 (except 1: Override value), the PNC interprets the 16-bit preset value directly as an override value in 0.01%.

☞ **Channel 0 defines the values for the auxiliary axes for which the values cannot be set directly.**

Customer input 1 – 8 qCh_Custom1 – 8

Interface that can be configured on the NC side and can be assigned as required.
(This option is currently not yet active.)

Online correction enable qCh_OnlCorrWcs

This signal enables the ”Online correction in workpiece coordinates” NC function for a selected coordinate.
In the case of online correction in workpiece coordinates, a handwheel is used to correct the position and orientation in the workpiece coordinate system (WCS) of a channel. This correction is possible with the part program active or inactive. For a detailed description, please refer to the ”Function Description” manual (for the order No., see page 1–8).

☞ **The online correction cannot be enabled in the ”Manual set-up”, ”Manual set-up workpiece coordinates” and ”Manual approaching the reference point” operating modes.**

Signal level

1: The selected online correction processes the handwheel data.

0: The selected online correction does not respond to the handwheel data.

Online correction direction qCh_OnlCorrWcsDir

Using this signal, it is possible to change the direction of correction.

Signal level

1: A positive direction of rotation of the handwheel results in a **negative** online correction.

0: A positive direction of rotation of the handwheel results in a **positive** online correction.

Terminate coupling qCh_CoordCoupleOff

Using this signal, the coordinate coupling for a slave channel is terminated.
Since the master channel must be at a standstill when the slave is coupled, it is recommended that the master be stopped during the coupling process using **Block transfer inhibit** and **Feed inhibit**. Both signals can be reset by the slave channel outputting the coordinate coupling active signal.
When coordinate coupling is completed, the axes of the master channel should be at a standstill.

Signal edge

0 → 1: The coordinate coupling for a slave channel is terminated. The slave channel then switches to the next block; the **Coordinate coupling active** signal is set to logic 0.

1 → 0: No effect.

G131 release qCh_TangTRotRel

This signal allows the control to execute an intermediate block if the contour knee angle for the tool rotation between two blocks is too large (see also iCh_TangTRotCmd on page 4–28).

Signal level

1: Release for the control to execute the intermediate block.

0: The release of the execution of an intermediate block is terminated. The control can execute additional blocks.

4.2.2 Input signals (NC –> PLC)

Active mode iCh_OpMode_00 – 03

The active operating mode is output in coded form:

Mode	**Active mode**			
	Bit 3 (iCh_ OpMode_03)	**Bit 2 (iCh_ OpMode_02)**	**Bit 1 (iCh_ OpMode_01)**	**Bit 0 (iCh_ OpMode_00)**
No operating mode	0	0	0	0
Manual set-up	0	0	0	1
Manual traverse to reference point	0	0	1	0
Reserved	0	0	1	1
Manual NC block entry	0	1	0	0
Next block	0	1	0	1
Program block	0	1	1	0
Single step	0	1	1	1
Reserved	1	0	0	0
Reserved	1	0	0	1
Single block	1	0	1	0
Restarting	1	0	1	1
CPL debugger program block	1	1	0	0
CPL debugger, next block	1	1	0	1
Manual set-up, workpiece coordinates	1	1	1	0
Reserved	1	1	1	1

Test mode iCh_DryRun

This signal is set when every axis of the channel is in Test mode.
In test mode, axes are still interpolated internally; however, the calculated command vlaues are only shown on the display and not transmitted to the drive. The axis is held in the same position where it was when test mode was activated.

Test mode can be activated using a softkey, automatically for drives that are not connected (no SERCOS ring present) or using MP 01001 00010 ”Virtual drive”.

NC ready iCh_NCReady

The NC ready signal is a prerequisite for setting the **NC start** signal.

Signal level

1: This is set when:

- a program has been selected or, while in MDI mode, a block is activated but not yet started,
- execution of a block in operating modes Program block, Single block or Single step is finished but execution of the next block has not yet started,
- execution of a program in operating modes Next block, Program block, Single block or Single step is finished.
- NC functions **M0** or **M1**, 'Program stop' were executed, or
- Cancel distance to go (qCh_CancDist) has been executed.

0: This is reset when:

- no NC program has been selected,
- the NC block has been entered
- execution has already been initiated with the NC start and the Program running signal has been set.

NC start must not be set while NC ready is reset.
There is only one exception, i.e. in combination with **Feed hold**. If the Feed hold signal has been set, NC ready remains = 0; however, the machining program can be restarted with NC start.

Program running iCh_ProgRun

This has an effect in operating modes **Next block, Program block, Single block** and **Single step** only if a program or, in **MDI** mode, an NC block has been selected previously.

Signal level

1: The signal is set if an NC part program has been started with **NC start** in operating modes Next block, Program block, Single block or Single step, or if the execution of a single NC block has been initiated with the NC start while in **MDI** mode. Output signals **Feed inhibit, Block transfer inhibit** and **Drive off** have no effect on this signal. If **Override = 0** is entered, the signal level remains at logic 1.

0: This is reset when:

- no program has been selected in operating modes Next block, Program block, Single block or Single step.
- a program has been selected or, while in MDI mode, a block is activated but not yet started,
- execution of a block in operating modes Program block, Single block or Single step is finished but execution of the next block has not yet started,
- NC functions **M0, M1, M2** or **M30** were executed previously, or
- Feed hold (qCh_FeedHold) is active.

Block transfer inhibit active iCh_TransferLockAct

This signal indicates that the channel will, due to an active block transfer inhibit, delay the execution of blocks until the block transfer inhibit is reset.

Signal level

1: After a block transfer inhibit has been activated and execution of the NC block active at that time has been completed, the channel waits for the block transfer inhibit to be reset and sets the ”Block transfer inhibit active” signal.
This signal can be used if, for example, **Cancel distance to go** is to be executed for the transfer of the external compensation at the block end.
As soon as this signal has been set after the block transfer inhibit has been activated, **Cancel distance to go** can be used to delete the prepared blocks so that the transferred compensation values are active as of the next block.

0: Block transfer inhibit inactive

☞ **This signal is not an acknowledgement signal indicating that the block transfer inhibit has been recognized by the NC; it indicates only that the block transfer inhibit has become active.**

Feed hold active iCh_FeedHoldAct

This signal remains set as long as the signal **Feed hold** is self-sustaining.

Signal level

1: The self-sustainment of the Feed hold signal is active; no traversing motions of machining axes can be executed until the Feed hold signal is reset by **NC start**.

0: Feed hold is not active; traversing motion of machining axes can be executed.

Program stop M0 iCh_ProgStopM0

This signal is output when the NC function **Program stop** (conditional and unconditional) is carried out. At the same time, NC-ready (iCh_NCReady) is set.
If no NC start (qCh_NCStart) is set before the block is completed, the program stops; it must be restarted using qCh_NCStart.

Signal level

1: This is set when:

- an NC block becomes active with Program stop (M0),
- an NC block becomes active with a conditional program stop (M1) and the signal Optional stop is set simultaneously.
  (qCh_OptStop) is set.

0: This is reset when:

- a program is restarted with NC start after a program stop (M0, M1),
- a program is deselected.

Program end M30 iCh_ProgStopM30

This signal is output when the main program is terminated.

Signal level

1: This is set when execution of the NC block is completed with M30 in the main program or when the main program has been fully executed, i.e. M30 terminates.

0: The signal is reset on an NC start (qCh_NCStart) a t the beginning of a program or when a program is deselected.

Channel reset iCh_Reset

This signal has an effect in conjunction with the Control reset signal and the Control reset function which is initiated via the NC operator interface (see also the signal diagram for Control reset on page 4–10).

Signal level

1: The Control reset function has been carried out internally by the control; the channel is in the power-up state.
The Control reset signal must be reset by this signal.
Signal level 1 is maintained for as long as the channel is in the power-up state and no program has yet been selected.

0: The channel is no longer in the power-up state; an NC part program has been selected or is already active.

Remove finished iCh_RemoveFinish

Signal level

1: The motion in departing from the contour is completed with the leading edge of ”Return to contour” (qCh_RetCont) or by pressing the ”Return to path” softkey. Any subsequent traversing motions will not be recorded.

0: Departing motions are recorded up to a maximum number of blocks (specified in MP 3080 00102).

Ready for reentry iCh_ReadyReEnter

Signal level

1: In the ”Automatic return to path” mode, with the Return to contour (qCh_RetCont) signal set, you can return to the saved contour using the NC start. In all other cases, machining can be continued with the NC start.

0: No automatic return to path or continuation of machining is possible.

Re-entry active iCh_ReEnterAct

Signal level

1: Automatic return to the saved contour has been started with an NC start; the point of return to path has not been reached yet.

0: Automatic return to path has not been started yet or is already completed.

Asynchronous subroutine 1 – 8 iCh_ASub1 – 8

Each of these signals is assigned to one of a maximum of 8 asynchronous subroutines.

Signal level

1: Asynchronous subroutine No. x active

0: Asynchronous subroutine No. x inactive

Channel status bit 0 – 4 iCh_State_00 – 04

Depending on the operating mode, the following channel states are output in binary code:

Mode	**Channel status**	**Bit 4, (iCh_ State_04)**	**Bit 3, (iCh_ State_03)**	**Bit 2, (iCh_ State_02)**	**Bit 1, (iCh_ State_01)**	**Bit 0, (iCh_ State_00)**
Inactive	Mode is not active; a process may be selected.	0	0	0	0	1
Ready	The mode is ready; the process may be started.	0	0	0	1	0
Running	The mode is active and executing a program or NC block.	0	0	0	1	1
Internal/reserved	Reserved	0	0	1	0	0
Internal/reserved	Reserved	0	0	1	0	1
Error	An error has occurred in the mode; this can be cleared only by ”Control reset” or ”Program deselection”.	0	0	1	1	0
Internal/reserved	Reserved	0	0	1	1	1
Control reset active	Control reset is being executed on the channel. No jobs may be accepted until the status switches to ”inactive”.	0	1	0	0	0
Program selection running	A program has been selected and is being prepared (e.g. linked).	0	1	0	0	1
Cancel distance to go preparation	”Cancel distance to go” was triggered; cleanup is taking place.	0	1	0	1	0
Cancel distance to go running	The mode is active and is reprocessing existing buffers.	0	1	0	1	1
Ready for program start	The mode is ready; the process is at the beginning of the program and may be started.	0	1	1	0	0
Ready for next block	All the blocks of the buffered NC block input have been executed. Waiting for the next input.	0	1	1	0	1

Activate block skip iCh_BlkSlash

This exists in conjunction with the character for **Block skip (/)** programmed in the NC part program and the **Block skip** signal.

Signal level

1: This is set when ”Block skip” was actuated on the NC operator interface. The signal remains set until the block skip softkey is pressed again.
The NC-internal function **Block skip** is not yet activated by actuating the softkey; to activate the function, the input signal Block skip (qCh_BlkSlash) must be set. This can be easily achieved by feeding **Activate block skip** back to the **Block skip** signal.
All NC blocks in the part program identified with a slash (/) are then skipped.

0: The **Block skip** softkey was not activated.

Optional stop activated iCh_OptStop

This exists in conjunction with the **M1** function programmable in the NC part program and the signal **Optional stop**.

Signal level

1: This is set when the Optional stop softkey is actuated on the NC operator interface. The signal remains set until the Optional stop softkey is pressed again.
The NC-internal function **Optional stop** is not yet activated by actuating the softkey; to activate the function, the input signal Optional stop (qCh_OptStop) must be set. This can be easily achieved by feeding **Activate optional stop** back to the **Optional stop** signal.
During execution of the NC part program, program execution is then interrupted in the block in which the function M1 has been programmed. Program execution can be continued by setting an NC start.

0: The Optional stop softkey was not activated.

Override 0% iCh_Override0

This signal acts in conjunction with the **Override** function.
The stages of impacting the feed rate can be weighted using MP 7030 00010.

Signal level

1: Signal Override 100% (qCh_Override100) is **not** set for the channel.
Any 100% feed programmed with G63 has no effect on this signal.

0: Override 100% has been selected using the interface.

Override 100% iCh_Override100

This signal acts in conjunction with the **Override** function.
The stages of impacting the feed rate can be weighted using MP 7030 00010.

Signal level

1: Signal Override 100% (qCh_Override100) is set for the channel.
Any 100% feed programmed with G63 has no effect on this signal.

0: No Override 100% has been selected using the interface.

CPL customer output 1 ... 16 iCh_Cpl01 – 16

These 16 signals can be read and written directly from the CPL part program by CPL command ”**BITIF(..)**” (see the programming manual).

Customer output 1 – 8 iCh_Custom1 – 8

Interface that can be configured on the NC side and can be assigned as required.
(This option is currently not yet active.)

Rapid traverse active iCh_G0Act

This signal is set if ”Rapid traversing” is active modally (G0, G10, G200).

Signal level

1: The signal is output as long as rapid traversing variants are active.

0: No Rapid traversing effective.

Inpos range 2 active iCh_InPosAct

This signal indicates that the Accurate stop mode is active in **In-position window 2 (rough)** (e.g. G165(IPS2)) for all of the axes assigned to the channel. Positioning window 2 is specified for each axis with SERCOS parameter S-0-0261.

If Accurate stop modes ..(IPS1) or ..(IPS3) are active, this signal is not set; the positioning window of SERCOS parameter S-0-0057, which is used to form the axis interface signal Axis in position (iAx_InPos), is used for the corresponding channel axes.

Signal level

1: **Positioning window 2 (rough)** is active for all axes assigned to the channel.

0: The ”normal” positioning window is active for all axes assigned to the channel.

G41/141 active iCh_G41G141Act
G42/142 active iCh_G42G142Act

These signals indicate that the respective function is active modally on the channel. G41/141 and G42/142 mutually exclude each other in programming, i.e. only one of these signals can be active at any time. If neither of the signals is set, no tool radius compensation is selected, i.e. G40 and G140 are active.

Signal level

1: G41/141 or G42/142 is active.

0: G41/141 or G42/142 is not active.

Coordinate coupling active iCh_CoordCoupleAct

This signal indicates that the respective channel is coupled to a master channel.

Signal level

1: Coordinate coupling is active.

0: Coordinate coupling is not active.

Tool turn (G131) iCh_TangTRotCmd

This signal indicates that the angle between 2 movement blocks has exceeded the current contour knee angle that has been programmed with G131 (Tangential tool leading).

Before executing the internally generated intermediate block, the NC waits until the PLC has set the acknowledgment **G131 release (qCh_TangTRotRel)**.
The iCh_TangTRotCmd signal remains set until the execution of the intermediate block is completed.

Signal level

1: The angle between two blocks exceeds the programmed contour knee angle.
The NC is waiting for the PLC to set the release or is executing an intermediate block.

0: The tool rotation does not exceed the contour knee angle.
No intermediate block waiting for the release to be set is currently present in the NC.

G70 active iCh_ActFunc01

(G0, G10, G200). Dieses Signal wird gesetzt, wenn im aktuellen Kanal G70 angewählt wurde. (G0, G10, G200). The values programmed for the linear synchronous axes are then interpreted as being set in inches.

Signal level

1: G71 is active (metric programming).

0: G70 is active (inch programming).

Feed 100% active iCh_ActFunc02

This signal indicates that the Set **override 100%** function has been programmed for the channel and is applied modally (G0, G10, G200). (G0, G10, G200). (G0, G10, G200). The **Override** 100% interface signal has no effect on the Feed 100% active signal (G0, G10, G200). (G0, G10, G200). If this signal is not set, the G66 function (Feed 100% off) is active on the channel, which in turn can be superimposed by the **Override** 100% interface signal (G0, G10, G200).

Signal level

1: The G63 function is active.

0: The G63 function is not active.

G92 active iCh_ActFunc05

The G92 active signal indicates that a **G92 shift** is active on the channel. The G92 setting is selected by control reset or by programming G92 without any axis addresses.

Signal level

1: A G92 shift is active.

0: A G92 shift is not active.

Thread cycle active iCh_ActFunc06

This signal can be set from a part program using the G533 TClx signal.

☞ **End of main program (M30) or control reset cancel the signal.**

Signal level

1: G533 TCl1 is programmed.

0: G533 TCl0 is programmed or no program is active.

Tapping without compensating chuck active

iCh_ActFunc07

(G0, G10, G200). Dieses Signal zeigt an, dass die Funktion Gewindebohren aktiv ist. (G0, G10, G200).

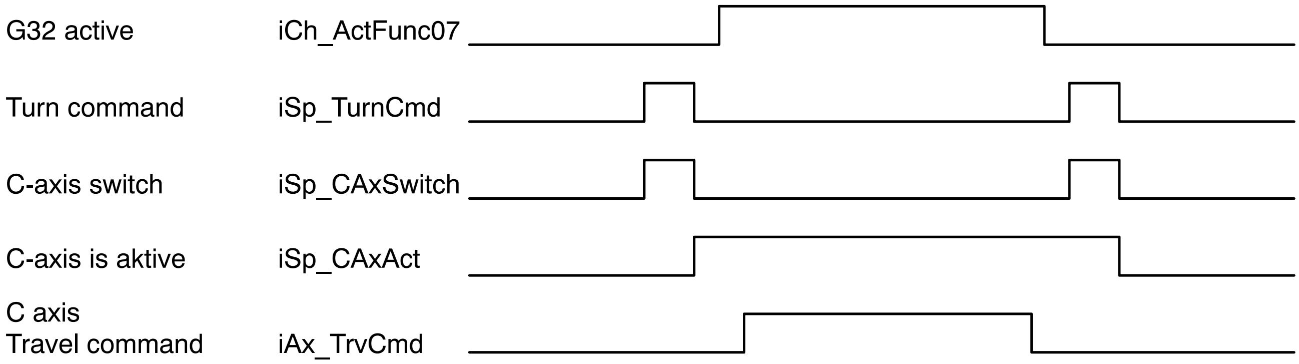
Signal level

1: Tapping (G32) is active.

0: Tapping is not active.

Example:

The sequence of the most important signals for tapping can be illustrated in an example.



Thread cutting active

iCh_ActFunc08

Is set in case of active thread cutting (G33).

Signal level

1: G33 active. The feedrate is dependent on the spindle speed and the thread pitch.

The following input signal reacts **differently**:

- qCh_CtrlReset (control reset)
  If configured and released, triggers a rapid retraction (see qCh_Retract, Rapid retraction).

The following input signals have **no effect**:

- qCh_Override100 (Override 100%)
- qCh_Override_00...15 (Override Bit 0...15)
- qCh_FeedStop (feed stop)
- qCh_FeedHold (feed hold)

The following interface signal must **not** be set:

- qCh_CancDist (cancel distance to go)

0: G33 is not active.

Tool compensation active bit 0 Bit 5 iCh_ActFunc09 ... 14

These signals indicate which tool compensation function is active in the channel.

correction	Output signal					
	Bit 5 iCh_ActFunc14	Bit 4 iCh_ActFunc13	Bit 3 iCh_ActFunc12	Bit 2 iCh_ActFunc11	Bit 1 (iCh_ ActFunc10	Bit 0 (iCh_ ActFunc10
no tool compensation (G146, G148)	0	0	0	0	0	0
G145 active	0	0	0	0	0	1
G245 active	0	0	0	0	1	0
G345 active	0	0	0	0	1	1
G445 active	0	0	0	1	0	0
G545 active	0	0	0	1	0	1
G645 active	0	0	0	1	1	0
G745 active	0	0	0	1	1	1
G845 active	0	0	1	0	0	0
G147 active	0	0	1	0	0	1
G247 active	0	0	1	0	1	0
G347 active	0	0	1	0	1	1
G447 active	0	0	1	1	0	0
G547 active	0	0	1	1	0	1
G647 active	0	0	1	1	1	0
G747 active	0	0	1	1	1	1
G847 active	0	1	0	0	0	0

☞ **When evaluating these signals, note the fact that the simultaneous use of universal tool compensation and external tool compensation cannot be reported correctly.**

G96 active iCh_ActFunc18

The constant cutting velocity active signal indicates whether the spindle on the channel traverses at constant cutting speed (G96 or G196).

Signal level

1: Constant cutting velocity is active.

0: Direct speed programming for spindle is active.

5 Axis-related interface signals

5.1 Overview of axis-related interface signals

The axis-related interface signals act on the NC functions which are associated with the control of a machining or auxiliary axis.
The number of existing axis interfaces corresponds to the index of the last axis or C-axis/spindle entered in MP 1001 00001 (identical to the physical axis or drive index). The axis interface of a C-axis is evaluated or written by the NC only when the C-axis is being operated.

When axes had been suppressed or ”simple” spindles had been configured before the last axis or C-axis/spindle defined, an axis-related interface is mapped to the corresponding markers for each of these axes; however, these interfaces will not be served by the NC!

The start addresses for the axis-related interface signals are defined axis-specifically in MACODA parameter 2060 00005.

The following description refers to the interface signals for one axis.

5.1.1 Overview of output signals (PLC –> NC)

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
0.0	qAx_OpModeSel_00	Bit 0 axis operating mode	1.0	qAx_TrvLim_00	Bit 0 limit switch range
0.1	qAx_OpModeSel_01	Bit 1 axis operating mode	1.1	qAx_TrvLim_01	Bit 1 limit switch range
0.2	qAx_JogPlus	Manual +	1.2	qAx_SwLimOff	Suppress limit switches
0.3	qAx_JogMinus	Manual –	1.3	–	res.
0.4	qAx_JogInch	Inch incr. step	1.4	–	res.
0.5	qAx_JogDia	Diameter incr. step	1.5	–	res.
0.6	qAx_NextNotch	Next grid position	1.6	–	res.
0.7	qAx_Reset	Basic axis setting	1.7	qAx_FxStopRel	Cancel fixed stop
2.0	qAx_SafOpModeSel	Mode selection	3.0	qAx_HandwSel_00	Bit 0 handwheel sel.
2.1	–	res.	3.1	qAx_HandwSel_01	Bit 1 handwheel sel.
2.2	qAx_SafAgreeButton	Consent key	3.2	qAx_HandwDir	Handwheel direction of rotation
2.3	qAx_SafSwitch1	Safety sw. 1 (S1)	3.3	–	res.
2.4	–	res.	3.4	–	res.
2.5	qAx_ SafCheckInputState	Safety status insp. input	3.5	–	res.
2.6	qAx_SafSignalState	Safety signal status	3.6	–	res.
2.7	–	res.	3.7	–	res.

Axis-related interface signals

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
4.0	qAx_ManFeed_00	Bit 0 manual feed	5.0	–	res.
4.1	qAx_ManFeed_01	Bit 1 manual feed	5.1	–	res.
4.2	qAx_ManFeed_02	Bit 2 manual feed	5.2	–	res.
4.3	qAx_ManFeed_03	Bit 3 manual feed	5.3	–	res.
4.4	–	res.	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	–	res.
4.7	–	res.	5.7	qAx_Override100	Axis override 100%
6.0	qAx_Override_00	Override bit 0	7.0	qAx_Override_08	Override bit 8
6.1	qAx_Override_01	Override bit 1	7.1	qAx_Override_09	Override bit 9
6.2	qAx_Override_02	Override bit 2	7.2	qAx_Override_10	Override bit 10
6.3	qAx_Override_03	Override bit 3	7.3	qAx_Override_11	Override bit 11
6.4	qAx_Override_04	Override bit 4	7.4	qAx_Override_12	Override bit 12
6.5	qAx_Override_05	Override bit 5	7.5	qAx_Override_13	Override bit 13
6.6	qAx_Override_06	Override bit 6	7.6	qAx_Override_14	Override bit 14
6.7	qAx_Override_07	Override bit 7	7.7	qAx_Override_15	Override bit 15
8.0	qAx_Custom1	Customer input 1	9.0	–	res.
8.1	qAx_Custom2	Customer input 2	9.1	–	res.
8.2	qAx_Custom3	Customer input 3	9.2	–	res.
8.3	qAx_Custom4	Customer input 4	9.3	–	res.
8.4	qAx_Custom5	Customer input 5	9.4	–	res.
8.5	qAx_Custom6	Customer input 6	9.5	–	res.
8.6	qAx_Custom7	Customer input 7	9.6	–	res.
8.7	qAx_Custom8	Customer input 8	9.7	–	res.
10.0	qAx_TrqErrOff	Supress standstill error	11.0	qAx_Discharge	Axis discharged
10.1	qAx_LagErrOff	Suppress coupling error	11.1	qAx_FrzIpoPos	Hold command position
10.2	qAx_MasterPos	Gantry in master position	11.2	–	res.
10.3	–	res.	11.3	qAx_TrqLim	Torque reduction
10.4	–	res.	11.4	–	res.
10.5	–	res.	11.5	–	res.
10.6	–	res.	11.6	qAx_DrvOn	Drive on
10.7	–	res.	11.7	qAx_DrvLock	Feed inhibit

Axis-related interface signals

5.1.2 Overview of input signals (NC –> PLC)

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
0.0	iAx_RefKnow	Reference point known	1.0	iAx_DistCtrl	Axis near endpoint
0.1	iAx_RefReached	Reference point reached	1.1	–	res.
0.2	iAx_TrvCmd	Travel command	1.2	–	res.
0.3	iAx_TrvDirNeg	Negative travel direction	1.3	–	res.
0.4	iAx_Run	Axis running	1.4	–	res.
0.5	iAx_InPos	Axis in position	1.5	–	res.
0.6	iAx_NotchPos	Axis in grid position	1.6	iAx_FxStopReached	Fixed stop reached
0.7	iAx_Reset	Axis homed	1.7	iAx_FxStopAct	Fixed stop active
2.0	iAx_SafOpMode_00	Bit 0 safety mode	3.0	iAx_PosSwitch1	Position switch point 1
2.1	iAx_SafOpMode_01	Bit 1 safety mode	3.1	iAx_PosSwitch2	Position switch point 2
2.2	iAx_SafOpMode_02	Bit 2 safety mode	3.2	iAx_PosSwitch3	Position switch point 3
2.3	iAx_SafOpMode_03	Bit 3 safety mode	3.3	iAx_PosSwitch4	Position switch point 4
2.4	iAx_SafStatePos	Safe position status	3.4	iAx_PosSwitch5	Position switch point 5
2.5	iAx_ SafCtrlOutputState	Control status output	3.5	iAx_PosSwitch6	Position switch point 6
2.6	–	res.	3.6	iAx_PosSwitch7	Position switch point 7
2.7	–	res.	3.7	iAx_PosSwitch8	Position switch point 8
4.0	iAx_ChIndex_00	Channel number bit 0	5.0	–	res.
4.1	iAx_ChIndex_01	Channel number bit 1	5.1	–	res.
4.2	iAx_ChIndex_02	Channel number bit 2	5.2	–	res.
4.3	iAx_ChIndex_03	Channel number bit 3	5.3	–	res.
4.4	–	res.	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	iAx_Override0	Axis override 0 %
4.7	–	res.	5.7	iAx_Override100	Axis override 100%
6.0	iAx_ScsState00	SCS signal status 0	7.0	iAx_ScsState08	SCS signal status 8
6.1	iAx_ScsState01	SCS signal status 1	7.1	iAx_ScsState09	SCS signal status 9
6.2	iAx_ScsState02	SCS signal status 2	7.2	iAx_ScsState10	SCS signal status 10
6.3	iAx_ScsState03	SCS signal status 3	7.3	iAx_ScsState11	SCS signal status 11
6.4	iAx_ScsState04	SCS signal status 4	7.4	iAx_ScsState12	SCS signal status 12
6.5	iAx_ScsState05	SCS signal status 5	7.5	iAx_ScsState13	SCS signal status 13
6.6	iAx_ScsState06	SCS signal status 6	7.6	iAx_ScsState14	SCS signal status 14
6.7	iAx_ScsState07	SCS signal status 7	7.7	iAx_ScsState15	SCS signal status 15
8.0	iAx_Custom1	Customer output 1	9.0	iAx_MasterAxIndex_00	Index of master axis bit 0
8.1	iAx_Custom2	Customer output 2	9.1	iAx_MasterAxIndex_01	Index of master axis bit 1
8.2	iAx_Custom3	Customer output 3	9.2	iAx_MasterAxIndex_02	Index of master axis bit 2
8.3	iAx_Custom4	Customer output 4	9.3	iAx_MasterAxIndex_03	Index of master axis bit 3
8.4	iAx_Custom5	Customer output 5	9.4	iAx_MasterAxIndex_04	Index of master axis bit 4
8.5	iAx_Custom6	Customer output 6	9.5	–	res.
8.6	iAx_Custom7	Customer output 7	9.6	–	res.
8.7	iAx_Custom8	Customer output 8	9.7	–	res.

Axis-related interface signals

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
10.0	iAx_TrqExceed	Standstill error	11.0	iAx_DrvErrClass1	Error diagn. class 1
10.1	iAx_CoupleLag	Tracking error	11.1	iAx_DrvChangeClass2	Change diagn. class 2
10.2	–	res.	11.2	iAx_DrvChangeClass3	Change diagn. class 3
10.3	–	res.	11.3	iAx_TrqLim	Torque reduced
10.4	–	res.	11.4	iAx_DryRun	Test mode
10.5	–	res.	11.5	iAx_DrvPower	Enabled for power activation
10.6	–	res.	11.6	iAx_DrvReady	Drive ready
10.7	–	res.	11.7	iAx_DrvAct	Drive under control

5.2 Signal description

5.2.1 Output signals (PLC –> NC)

Depending on their function, the PNC evaluates the signals as either **static signals** with levels logic 0 and logic 1 or as **edge-triggered signals** with signal edges 0→1 or 1→0.

Bit 0 and 1 axis operating mode qAx_OpModeSel_00 – 01

This signal serves to define whether the auxiliary axes are to travel in the **Manual setup** or **Manual traverse to reference point** mode when Manual +/– is actuated (provided that the axis has not been assigned to a channel by programming, i.e. it must not move).
The selected modes of the channels are of no importance because asynchronous axes are not assigned to a specific channel.

Operating modes are coded as follows:

Operating modes	**Mode selection**	
	Bit 1 (qAx_OpModeSel_01)	**Bit 0 (qAx_OpModeSel_00)**
Programming	0	0
Manual set-up	0	1
Manual traverse to reference point	1	0
Reserved	1	1

Manual + qAx_JogPlus
Manual – qAx_JogMinus

The Manual+ and Manual– signals are effective for **channel axes** (synchronous axes) and **auxiliary axes** (asynchronous axes) in the **Manual set-up** and **Manual traverse to reference point** operating modes. In addition, the signals are used for manual jogging of synchronous axes when **moving away from the contour**. The active part program is stopped with feed hold in this context.

Signal level

1: Continuous jogging:

If channel/axis mode **Manual set-up** has been set and all other requirements have been fulfilled (feed hold, feed inhibit, etc. are not set), the axis is moved in the appropriate direction using Manual+ or Manual–.
Feed is selected using ”Manual feed” (qAx_ManFeed_00 – 03); acceleration is set in MP 1010 00006.
Signals ”Travel command” (iAx_TrvCmd), ”Axis running” (iAx_Run) and, depending on the travel direction, ”Negative travel direction” (iAx_TrvDirNeg) are set on the axis interface. The ”Axis in position” (iAx_InPos) signal is cancelled at the same time.

Incremental jogging:

The same as above, but the axis travels the incremental step selected using ”Manual feed” (qAx_ManFeed_00 ... 03). The axis continues to move even when the Manual+ or Manual– signal is no longer active and the movement along the preselected path has not been completed. The axis moves along the path with the speed entered in MP 1005 00007 (defined steps) or MP 1005 00008 (variable steps).
A new movement can be triggered only if Manual+ or Manual– previously had signal level logic 0 (edge triggering).

Traverse to reference point:

If the channel or axis operating mode **Manual traverse to reference point** has been set and all other prerequisites have been met, drive-controlled referencing is triggered for the axis using Manual+ or Manual–. The drive’s behavior is determined in SERCOS referencing parameter S-0-0147.

Signal level

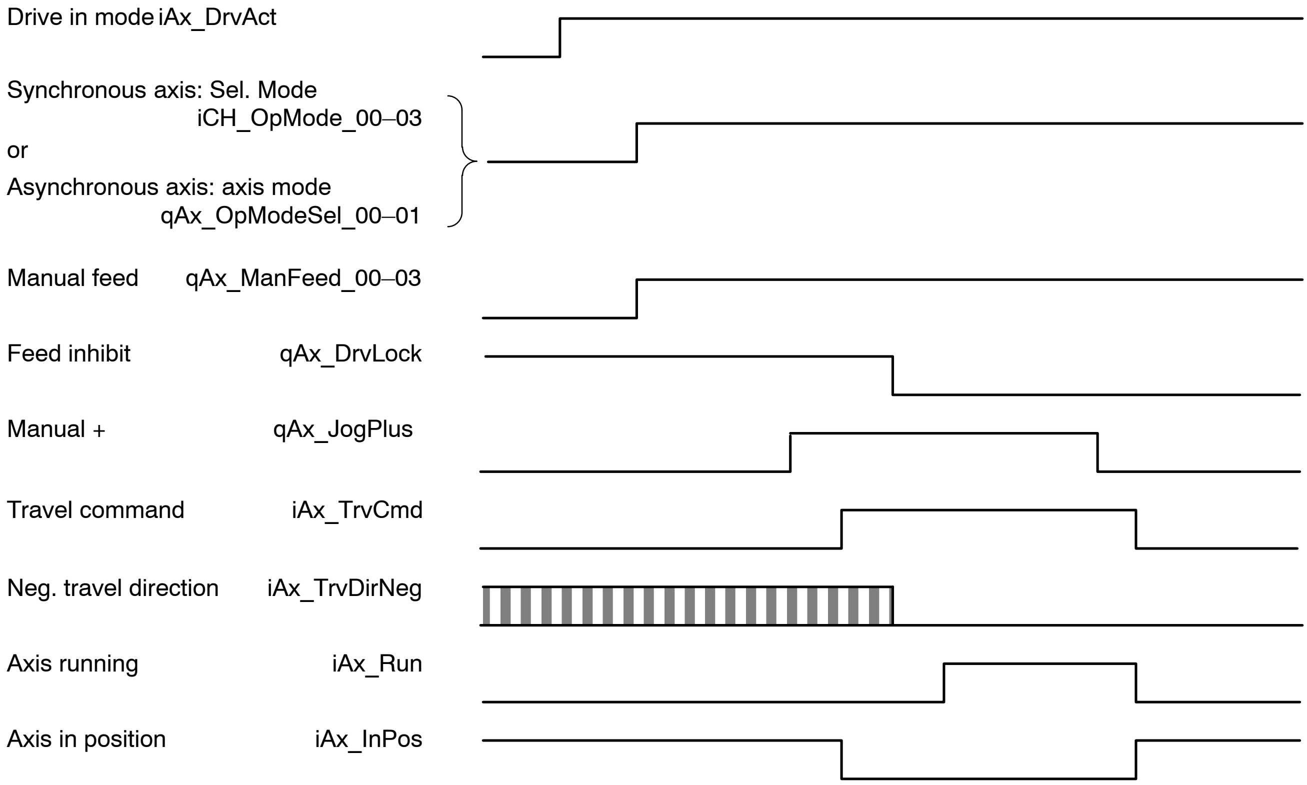
0: Continuous jogging:

When the signal Manual+ or Manual– is cancelled, the traversing movement is stopped with the deceleration set in MACODA. The travel command is reset. As soon as the axis is located in the in-position window, the Axis running signal is reset and Axis in position is output.

Axis-related interface signals

Example:

Moving an axis (synchronous or asynchronous) in the positive direction in **Manual** operating mode.


Drive in mode iAx_DrvAct
Synchronous axis: Sel. Mode iCH_OpMode_00–03
or
Asynchronous axis: axis mode qAx_OpModeSel_00–01
Manual feed qAx_ManFeed_00–03
Feed inhibit qAx_DrvLock
Manual + qAx_JogPlus
Travel command iAx_TrvCmd
Neg. travel direction iAx_TrvDirNeg
Axis running iAx_Run
Axis in position iAx_InPos

The control is switched on, the control circuit is already closed and the drive generates a torque. **Drive under control** (iAx_DrvAct) has been set.

Setup mode must be active on the respective channel in order to move an axis **manually** in the positive or negative direction. **Setup** mode is coded into the channelrelated interface and output on signals iCH_OpMode_00–03.
For auxiliary axes, the mode selection qAx_OpModeSel_00–01 must have been set accordingly. **Manual feed** or **Incremental step** is set in coded format with signals qAx_ManFeed_00–03.

When the **Manual+** signal is set, the **travel command** and **traversing direction signals are output.**
When the **travel command** is output, the **Axis in position** signal is reset.

As long as **Feed inhibit** is active, movement is impossible. Once the Feed inhibit signal is reset, the axis moves as long as Manual+ is actuated or until the specified **incremental step** has been executed. While the axis is traversing, the **Axis running** signal is output. If feed inhibit was reset at an earlier point in time, the axis will start moving as soon as the Manual+ signal is set.
The **Travel command** signal remains set until the traversing movement has been completed. When the axis has come to a standstill, the **Axis running** signal is reset and the **Axis in position** signal is output if the axis is within the In-position window. The **Traversing direction** signal remains until the next travel command in the opposite direction is output.

Inch incr. step qAx_JogInch

Setting this signal allows jogging by ”incremental steps in inches” or traversing with the handwheel while in setup mode. The incremental step specified via the interface is also interpreted in inches.

The inch increment of an axis results from
0.0001 x MP 1015 00001 [axis] in inches.

☞ **For rotary and endless axes, interface signal ”Inch incr. step” makes no difference.**

Example: MP 1015 00001 = 1

1 incremental step in inches = 0.0001 inch

1 incremental step in metric units = 0.001 mm

Signal level

1: In jog mode or handwheel mode, the incremental step specified is interpreted in ”inches”.

0: In jog mode or handwheel mode, the incremental step specified is interpreted in ”metric”.

Diameter incr. step qAx_JogDia

Setting this signal allows jogging by ”incremental steps in diameter” or traversing with the handwheel while in setup mode. The incremental step specified via the interface is also interpreted as a diameter difference.
The unit of measurement of the increments is specified by the signal Inch incr. step (qAx_JogInch).

Signal level

1: In jog mode or handwheel mode, the incremental step specified is interpreted as a diameter difference.

0: In jog mode or handwheel mode, the incremental step specified is interpreted as a radius difference.

Next grid position qAx_NextNotch

With this signal, the NC can move a Hirth axis to its next grid position. For this purpose, the NC must be in **Manual set-up** mode and the Hirth axis must not be activated by **Jog mode** at the same time (iAx_NotchPos has been set).

The traversing direction which was active last (see iAx_TrvDirNeg) determines the direction in which the Hirth axis is moved.
If a Hirth axis has come to a halt between 2 grid positions (e.g. due to an emergency shutdown), this signal may be used to move a Hirth axis to a grid position in the traversing direction specified.

Basic axis setting qAx_Reset

This signal is used to cancel the movement of an asynchronous axis. After the axis has come to a standstill, signal iAx_Reset is set.

Limit switch range 0 ... 1 qAx_TrvLim_00 ... 01

Selects one of four possible limit switch ranges.

Bit combination 0:	1st limit switch range active
Bit combination 1:	2nd limit switch range active
Bit combination 2:	3rd limit switch range active (not yet available)
Bit combination 3:	4th limit switch range active (not yet available)

☞ **Signal ”Suppress limit switches” (qAx_SwLimOff) affects the selected limit switch range.**

Suppress limit switches qAx_SwLimOff

This signal suppresses the positive and negative limit switches of the respective axis. In this case, no position limit monitoring is active on the NC for this axis.
As long as the reference point of the axis is not known (see iAx_RefKnow), this signal is not relevant. The internal checks performed by the NC are always switched off in this case.

Monitoring within the drive is not affected by this signal (e.g. S-0-0049 Position limit value positive; S-0-0050 Position limit value negative). Other safety-relevant functions must be assumed by the machine and/or PLC when the software limit switches are suppressed.
The **Suppress limit switches** signal is active in all operating modes.

Signal level

1: The software limit switches are suppressed.

0: The software limit switches are not suppressed. The NC monitors the position limits as defined in the machine parameters in accordance with the limit switch range selected.

Cancel fixed stop qAx_FxStopRel

Setting this signal cancels an active fixed stop for an asynchronous axis.

Mode selection qAx_SafOpModeSel

This signal is a component of the safety technology for the IndraDrive series.
The operating mode selection of the PLC is transferred as channel 1 to the drive via the SERCOS interface in real time.

Signal level

1: Normal mode NO

0: Special mode SO
(special mode at standstill / with movement)

Consent key qAx_SafAgreeButton

This signal is a component of the safety technology for the IndraDrive series.
Switching to ”Special mode with movement” via the PLC is transferred as channel 1 to the drive via the SERCOS interface in real time.

Signal level

1: Special mode with movement

0: Special mode at standstill

Safety sw. 1 (S1) qAx_SafSwitch1

These signals are components of the safety technology for the IndraDrive series.
Two different parameter sets can be stored in the drive for ”Special mode with movement”. The parameter set selected using the PLC is transferred as channel 1 to the drive via the SERCOS interface in real time.

Signal level

1: Parameter set 2 for safe movement active

0: Parameter set 1 for safe movement active

Safety status insp. input qAx_SafCheckInputState

This signal is a component of the safety technology for the IndraDrive series.
The safe status signal set using the PLC is transferred as channel 1 to the drive via the SERCOS interface in real time.

Safety signal status qAx_SafSignalState

This signal is a component of the safety technology for the IndraDrive series.
The control signals of the PLC (qAx_Saf...) that are transferred via channel 1 to the drive must be adjusted periodically so that they can be checked. As a result, the PLC sets the signals to ”0” for one cycle. At the same time, the PLC sets the safety signal status from ”valid” to ”adjusted”, thus informing the drive that the control signals are invalid.
Then the signals are reset.

Signal level

1: The control signals of the safety technology are being adjusted

0: The control signals of the safety technology are valid

Handwheel selection, bit 0 and bit 1 qAx_HandwSel_00 ... 01

Two handwheels can be connected to the NC control. As soon as one of the two signals is set, the axis can be traversed with the selected handwheel. Signals **Manual+** and **Manual–** have no effect.

If the axis is traversed with the handwheel, signals **Manual feed 0–3** are active. In this context, the incremental steps that can be selected are used in the calculation to weight the scale marks of the handwheel.

Handwheel	**Handwheel assignment**	
	Bit 1 (qAx_HandwSel_01)	**Bit 0 (qAx_HandwSel_00)**
no handwheel active	0	0
Handwheel 1	0	1
Handwheel 2	1	0
Reserved	1	1

Handwheel direction of rotation qAx_HandwDir

This signal permits the direction of axis rotation to be specified, regardless of the direction of handwheel rotation.

Signal level

1: The direction of axis rotation is opposite to the direction of handwheel rotation.

0: The directions of axis and handwheel rotation are the same.

Manual feed for bit 0 – bit 3 qAx_ManFeed_00 – 03

These signals are effective in the Setupoperating mode. If the axes are traversed manually with the jog button or handwheel, the **feed rates** or **incremental steps** which are assigned to the corresponding bit combination become active. Feedrates are defined in MP parameters 1005 00002 .. 1005 00008.

When the axes are moved using a handwheel, the **incremental steps are evaluated as the weighting factor** for the scale divisions of the handwheel.
The incremental steps for **x increments** are defined in machine parameter 1015 00002.

Manual feed / incremental step	**Bit 3 (qAx_ ManFeed_03)**	**Bit 2 (qAx_ ManFeed_02)**	**Bit 1 (qAx_ ManFeed_01)**	**Bit 0 (qAx_ ManFeed_00)**
No input	0	0	0	0
Rapid	0	0	0	1
Fast	0	0	1	0
Medium	0	0	1	1
Slow	0	1	0	0
x increments	0	1	0	1
1000 increments	0	1	1	0
100 increments	0	1	1	1
10 increments	1	0	0	0
1 increment	1	0	0	1

Axis override 100% qAx_Override100

This signal suppresses the axis override of an asynchronous axis. In this case, the axis traverses at the preselected or programmed velocity. The signal refers to programmed asynchronous movements/external command value inputs as well as to manual mode (jogging).
The signal has no effect on movements interpolated by the drive (e.g. traverse to reference point).

Signal level

1: Axis override is suppressed.

0: Axis override is enabled.

Override bit 0 – bit 15 qAx_Override_00 – 15

The traversing speed of asynchronous axes can be influenced by the **override function**. It affects the speeds programmed in the **part program** and on the preselected speeds in **Setup mode**.

This signal is used to activate one override weighting level for an auxiliary axis if the override has not been suppressed either using the interface (qAx_Override100) nor the machine parameters.

The levels of the override switch are queried by the NC in **binarily coded** form. Each binary code is assigned a value via MP 7030 00010 of **channel 0 (!)**, which represents the respective speed as a percentage.

Customer input 1 – 8 qAx_Custom1 – 8

Interface that can be configured on the NC side and can be assigned as required.
(This option is currently not yet active.)

Supress standstill error qAx_TrqErrOff

This signal suppresses an internal error state when the standstill torque has been exceeded. In this case, it is necessary for standstill torque monitoring to be programmed in the PLC.

☞ **Regardless of this signal, the ”Standstill error” (iAx_TrqExceed) error signal is set when the standstill torque has been exceeded!**

Signal level

1: If the standstill torque has been exceeded, the NC will **not** create an internal error signal. Exceedance is tolerated.

0: Error status when the level drops below the standstill torque.

Suppress coupling error qAx_LagErrOff

This signal suppresses an internal error state when the coupling lag limit has been exceeded. In this case, it is necessary for coupling lag monitoring to be programmed in the PLC.

☞ **Regardless of this signal, the ”Coupling lag” (iAx_CoupleLag) signal is set if the coupling lag limit has been exceeded!**

Signal level

1: In case a coupling lag limit is exceeded, the NC will **not** create an internal error signal. Exceedance is tolerated.

0: Coupling lag limit monitoring generates an error.

Gantry in master position qAx_MasterPos

CAUTION
The signal must be set only if the encoders have been commissioned – otherwise, mechanical damage may occur.

In case of a Gantry slave with an absolute encoder, this signal is used to transfer the command position of the master axis without any offset as the command position of the slave axis.

Prerequisites:

- Master and slave with absolute encoder.
- Difference of positions less than the maximum following error configured in the machine parameters.

In case of cyclically absolute encoders, the axis positions are not known prior to referencing. This is why the distance between the Gantry axes, which is determined from the actual position values provided by the encoders, is maintained until referencing for safety reasons.
In case of axes with absolute encoders, the actual position values are also known without referencing. When the encoders have been commissioned, a possible skew can be recognized from the values they supply.

Signal level

1: The skew of the slave axis is compensated by a jerk as soon as the drives are in operation.
If the distance is too great, a servo error will occur.

0: No effect.

Axis discharged qAx_Discharge

When this signal is set, the axis is no longer checked within the Inpos logic. Nevertheless, the axis can remain in the configuration of a channel if signal **Drive on** has not also been set.

Hold command position qAx_FrzIpoPos

This signal prevents the actual drive position from being corrected with the position control loop open, i.e. ”Drive under control” (iAx_DrvAct) is not set.

Signal level

1: The actual position of the drive is not transferred to the NC when the position control loop is open.

0: When the position control loop is open, the actual drive position is input in the NC.

Torque reduction qAx_TrqLim

This signal allows the torque limit (**SERCOS parameter S-0-0092**) valid in the drive to be changed.

This signal is of particular significance in conjunction with the axis clamping function: With the axis clamped, the position control loop may remain closed while the torque of the drive can be reduced in order to minimize mechanical stress. To a large extent, this prevents mechanical axis displacement when clamping is applied or released.
The value of the changed torque limit is set individually for each axis via MP 1003 00010. It can also be programmed via the NC function G177.

A torque changeover is permissible only when the axis is at a standstill.

If the torque limit is not accepted by the drive (e.g. because the parameter is write-protected), no error message is displayed. However, signal ”Torque limited” (iAx_TrqLim) is not set.

Signal edge

0 → 1: The torque reduction is transmitted to the drive via the SERCOS interface.

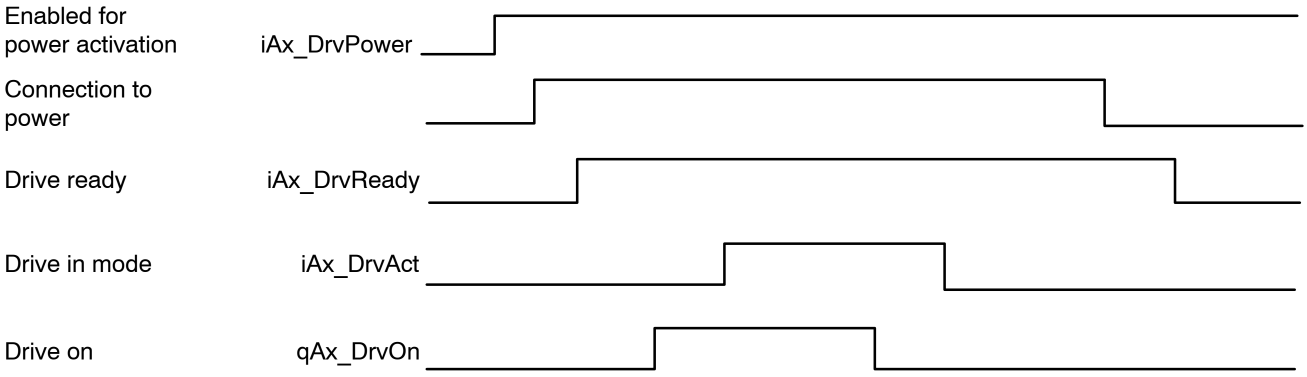
1 → 0: The original torque limit, as specified in SERCOS parameter S-0-0092, is transmitted to the drive.

Axis-related interface signals

Drive on qAx_DrvOn

This signal switches the torque in the drive on when it is ready for operation (iAx_DrvReady has been set). The control circuit is closed.

Resetting **Drive on** initiates a shutdown process. If the axis is still in motion at this time, it is brought to a halt at the preset deceleration. When the axis has come to a standstill, the position control loop is opened and the **Drive under control** signal is reset. Torque is no longer applied on the drive.



Example: Functioning of the starting and shutdown cycles

When the machine is started or voltage is applied, the NC unit and the servo and spindle drives go through an initialization phase. While the initialization phase of the individual components is running, the digital interface for communication between the NC and the drives is established, among other things.

This digital interface corresponds to the settings of the **SERCOS interface.** Initialization of the **SERCOS interface** is in-system, automatic and consists of 4 phases. Once the interface is initialized and cyclic operation is active, the **Enabled for power activation** signal is output. Power can be switched on for the affected drives. When power has been switched on, the **Drive ready** signal is output. The position control loop is closed by setting **Drive on**. The drive is under torque and the **Drive under control** signal is issued.

The shutdown cycle or disconnection from the system is initiated by the PLC sequential program resetting the **Drive off** signal. If the axis is still in motion at this time, it is brought to a halt at the preset deceleration. When the axis has come to a standstill, the position control loop is opened and the **Drive under control** signal is reset. Torque is no longer applied on the drive.

The power can be switched off in the next step. The **Enabled for power activation** signal remains set as long as the machine components (NC and drives) remain initialized.

Feed inhibit qAx_DrvLock

This signal is effective when the control loop is closed and torque is applied on the axis.

It prevents a programmed or manually preset axis movement from being carried out; however, it has no influence on the Travel command, Negative travel direction and Axis in position input signals.

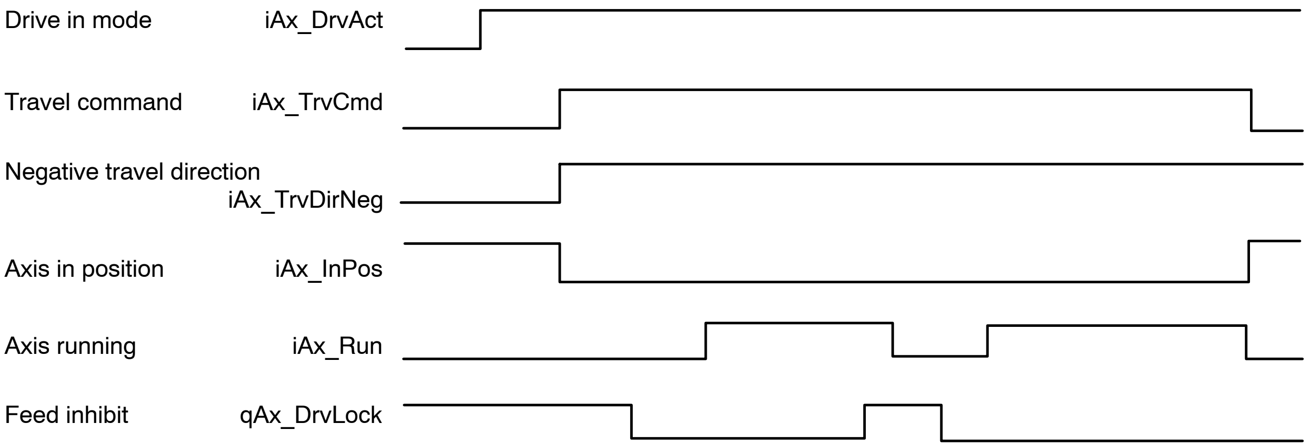
Signal level

1: Either the axis cannot be moved or it is stopped with the deceleration set in the parameters if a movement is already active.
If **Feed inhibit** is set for an axis which participates in an interpolation with other axes, the other axes cannot be traversed either.

0: Axis feed is not inhibited. If an axis travel command is initiated (via the NC part program or manually), the axis can start traversing immediately.
If a traversing axis was previously brought to a standstill with **Feed inhibit**, it will start accelerating to its programmed speed as soon as the Feed inhibit signal is reset.

Example:

The machine is switched on and the control loop has been closed. A negative traversing motion has been programmed for an axis via the NC part program.


Drive in mode iAx_DrvAct
Travel command iAx_TrvCmd
Negative travel direction iAx_TrvDirNeg
Axis in position iAx_InPos
Axis running iAx_Run
Feed inhibit qAx_DrvLock

The **Drive under control** signal informs the PLC sequential program that the control loop is closed.
The **Travel command** and **Neg. traversing direction** signals are output as soon as the NC block where the traversing motion is programmed is executed. The traversing direction is up-to-date only if a traversing command has been set.

The **Axis in position** signal is reset when the travel command is activated.
Traversing movement is not carried out as long as the **Feed inhibit** signal is set.
When Feed inhibit is reset, the programmed traversing movement is performed and **Axis running** is set.

If **Feed inhibit** is set again during the traversing movement, the traversing movement is stopped and **Axis running** is reset when the axis has come to a standstill.

When **Feed inhibit** is reset, the axis accelerates up to the previously applied speed, **Axis running** is set and the traversing movement is completed.

When the programmed end position is reached, the **Travel command** and **Axis running** signals are reset and the **Axis in position** signal is set.

5.2.2 Input signals (NC –> PLC)

Reference point known iAx_RefKnow

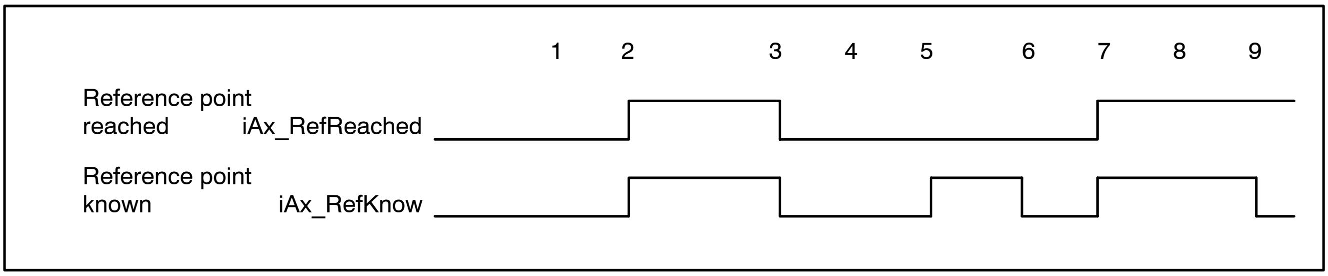
This signal is set as soon as the reference point of the axis has been reached, or, if absolute encoders are used, as soon as they have been recognized by the control. To do this, the control evaluates SERCOS parameter **Actual position value status** S-0-0403. The NC does not monitor any limit switches as long as the **Reference point known** signal has not yet been set.

☞ **After this signal has been set once, it will be reset only when a new SERCOS phase run-up has been initiated, or in the case of a drive fault (e.g. defective encoder).**

Signal level

1: The reference point has been reached, or the axis is equipped with absolute encoders.

0: No reference point has been approached yet.


1
2
3
4
5
6
7
8
9
Reference point reached iAx_RefReached
Reference point known iAx_RefKnow

1 Start approaching reference point

2 Approaching reference point completed

3 Start approaching reference point

4 Cancel approaching reference point
(reference point has already been recognized by the drive)

5 Control reset

6 Start approaching reference point

7 Approaching reference point completed

8 Drive lost reference point due to a fault

9 Control reset

Reference point reached iAx_RefReached

In contrast to the **Reference point is known** signal, this signal is set only when traversing to the reference point has been successfully completed. If traversing to the reference point was cancelled, neither of the two signals will be set.
While a subsequent basic setting causes a status check of the actual position values to be performed in the drive (parameter S-0-0403) and sets (or clears) the **Reference point is known** signal, as applicable, the **Reference point was reached** signal remains unchanged.

This signal is reset at the beginning of any traverse to a reference point.

Signal level

1: The last traverse to the reference point was properly completed by reaching the reference point.

0: Either no reference points have been approached so far, or traversing to the reference point is still active and the reference point has not yet been reached, or the last traverse to the reference point was cancelled.

Travel command iAx_TrvCmd

The signals **Travel command, Neg. traversing direction, Axis running** and **Axis in position** affect each other.

Examples of their relationships are shown in the signal diagrams and explanations of signals **Manual+/–** (page 5–7) and **Feed inhibit**(page 5–18).
The **Neg. traversing direction** signal is output in conjunction with the **Travel command** signal.

Signal level

1: The input for a traversing movement exists. It can be carried out by manual input or via the NC part program.

0: No command for the traversing movement is present.

Negative travel direction iAx_TrvDirNeg

This signal is effective only in conjunction with the signal **Travel command.** In the absence of a travel command signal, this signal is irrelevant.

Signal level

1: If ”Negative traversing direction” and ”Travel command” are set, the respective axis is to execute a traversing movement in the negative direction or the previous movement of this very axis was executed in the negative direction.

0: If ”Negative traversing direction” is not set and ”Travel command” is set, the respective axis is to execute a traversing movement in the positive direction or the previous movement of this very axis was executed in the positive direction.

Axis running iAx_Run

The signals **Travel command, Negative traversing direction, Axis running** and **Axis in position** affect each other.

Examples of their relationships are shown in the signal diagrams and explanations of signals **Manual+/–** (page 5–7) and **Feed inhibit**(page 5–18).

If the traversing movement of an axis has been concluded on the NC side, no **travel command** is active any more, and the signals **Axis in position** or **Axis running** will be output.

Signal level

1: Axis running remains set for as long as the axis executes a programmed or manually preset movement.
This signal is not set in the case of **drive-controlled approaching of reference point** even if the axis is still in motion.

0: The axis performs no movement.

Axis in position iAx_InPos

The signal is active when the control loop is closed and torque is applied on the axis.

Signal level

1: The axis is located within the in-position window set in the parameters and no travel command is active. The in-position window is specified by SERCOS parameter S-0-0057.
The ”In-position rough” window is specified in SERCOS parameter S-0-0261; that for Diax/EcoDrives and HNC100 is set in machine parameter 1015 00100.

0: The axis is not in the parameterized in-position window.

Axis in grid position iAx_NotchPos

This signal indicates when the Hirth axis has reached a grid position. The level of this signal is 0 as long as the reference point has not been approached. This signal is set to 1 as soon as the axis is within the selected in-position window.

Signal level

1: The Hirth axis is within the selected in-position window.

0: Indicates that the Hirth axis has not yet approached a reference point.

Axis homed iAx_Reset

This signal indicates that the asynchronous axis accepted the control reset task and that the asynchronous axis is ready to accept new inputs for motions.

Axis near endpoint iAx_DistCtrl

This signal indicates that this axis is near its endpoint in an area set via MP 1015 00010.

Signal level

1: Axis command position within the indicated distance.

0: Axis command position outside of the indicated distance, not configured, approaching reference point is active or handwheel is active.

Fixed stop reached iAx_FxStopReached

This signal is output when the axis is positioned at a fixed stop.

The signal is reset when the axis leaves the fixed stop monitoring window (MP 1003 00032) or when the fixed stop is cancelled with G476.

Signal level

1: Axis has reached the fixed stop.

0: Fixed stop left or cancelled.

Fixed stop active iAx_FxStopAct

This signal is output when the traverse movement to the fixed stop is active.
A movement to the fixed stop is triggered by the G375 (measuring fixed stop) or G475 (move to fixed stop) functions.

Signal level

1: Axis moves to fixed stop.

0: No move to fixed stop active.

Safety mode, bit 0 ... bit 3 iAx_SafOpMode_00 – 03

These signals are components of the safety technology for the IndraDrive series.
The NC reads the selected operating mode of the drive (P-0-3215) and passes the information to the PLC via these outputs.

Safety function	Safety mode			
	Bit 3 (iAx_ SafOpMode_00)	Bit 2 (iAx_ SafOpMode_00)	Bit 1 (iAx_ SafOpMode_00)	Bit 0 (iAx_ SafOpMode_00)
Normal mode NO	0	0	0	0
Safe starting lockout (ASP)	0	0	0	1
Safe stop (SH) / Safe mode stop (SBH)	0	0	1	0
Safe movement (SBB1)	0	0	1	1
Safe movement (SBB2)	0	1	0	0
Safe movement (SBB3)	0	1	0	1
Safe movement (SBB4)	0	1	1	0
Safety technology not active	1	1	1	1

Safe position status iAx_SafStatePos

This signal is a component of the safety technology for the IndraDrive series.
The drive uses this signal to report whether safe referencing has been carried out; this is a requirement for safety function ”Safe absolute position”.

Signal level

1: The drive is safely referenced.

0: No safe referencing

Control status output **iAx_SafCtrlOutputState**

This signal is a component of the safety technology for the IndraDrive series.
The drive uses this signal to provide feedback about its safe status.

Signal level

1: Safe status of the drive

0: No safe status of the drive

Position switch point 1 – 8 **iAx_PosSwitch1 – 8**

This signal **serves to monitor the actual position of an axis.** An axis may be assigned a maximum of 8 axis monitoring positions (electric limit switches).
Using machine parameters 2010 00100 through 2010 00130, you can define up to 64 axis monitoring positions for the whole system.

By **combining** two signals, you can create ”monitoring windows”.
For further information, see the ”Konfiguration parameter and MACODA” manual.

Signal level

1: The actual axis position is greater than or equal to the monitoring position.

0: The actual axis position is less than the monitoring position.

Channel number bit 0 – bit 3 **iAx_ChIndex_00 – 03**

This signal is used to output the binarily coded channel number of the corresponding channel.
”0” is output if the axis is not assigned to any channel (e.g. an asynchronous axis).

Channel number	**Bit 3 (iAx_ ChIndex_03)**	**Bit 2 (iAx_ ChIndex_02)**	**Bit 1 (iAx_ ChIndex_01)**	**Bit 0 (iAx_ ChIndex_00)**
No channel	0	0	0	0
Channel 1	0	0	0	1
...	...	...	...	...
Channel 12	1	1	0	0

If axes are lent to another channel, they are output in the following manner:

Status	Channel information
Channel is activated (for program selection or manual data input) with one axis.	Number of this channel
Channel is deactivated (for program deselection or end of manual data input) with one axis.	Number of this channel
Channel releases axis with G512; axis has not been released on its original channel.	Number of the original channel
Channel releases axis with G512; axis has been released on its original channel.	0

Definitions:

- ”Lent axis”: an axis is borrowed if a channel ”accesses an axis of an inactive channel” while the axis was not released on this channel with G512.
- ”Original channel”: when the NC starts up, all synchronous axes belong to an original channel.

Axis override 0% iAx_Override0

This signal for an asynchronous axis indicates that its axis override is set to 0% and that no Override 100% has been selected using the interface (qAx_Override100).

Signal level

1: 0% acts on the axis.

0: A value not equal to 0% acts on the axis.

Axis override 100% iAx_Override100

This signal for an asynchronous axis insicates that its axis override is set to 100% and/or that Override 100% has been selected using the interface (qAx_Override100).

Signal level

1: A value of 100% acts on the axis.

0: A value not equal to 100% acts on the axis.

SCS signal status 0 ... 15 iAx_ScsState00 – 15

The bit signals as configured in the cyclic telegram from the axis drive are transferred to the NC in the signal status word (SERCOS parameter S-0-0144).

The status signals can be configured in parameter S-0-0026. Parameter S-0-0144 must be included in the configuration list of the drive telegram (S-0-0016) to transfer the signals in the cyclic telegram.

Configuration example:

- S-0-0026 = (330, 331,0, 333, 0, 335, 336, 337, 0, 0, 0, 0, 0, 0, 0, 0): Configuration of the signal status word
- S-0-0016 = (40, 144): Configuration of the drive telegram

The following status messages are then allocated to the individual signals:

iAx_ScsState00:	S-0-0330	message $n_{is} = n_{comm}$
iAx_ScsState01:	S-0-0331	message $n_{is} = 0$
iAx_ScsState02:	-	
iAx_ScsState03:	S-0-0333	message $M_d \geq M_{dx}$
iAx_ScsState04:	-	
iAx_ScsState05:	S-0-0335	message $n_{comm} > n_{lim}$
iAx_ScsState06:	S-0-0336	message in-position
iAx_ScsState07:	S-0-0337	message $P \geq P_x$
iAx_ScsState08:	-	
iAx_ScsState09:	-	
iAx_ScsState10:	-	
iAx_ScsState11:	-	
iAx_ScsState12:	-	
iAx_ScsState13:	-	
iAx_ScsState14:	-	
iAx_ScsState15:	-	

☞ **For a detailed description of all status messages available, please see the SERCOS interface Specifications.**

Customer output 1 – 8 iAx_Custom1 – 8

Interface that can be configured on the NC side and can be assigned as required.
(This option is currently not yet active.)

Index of master axis bit 0 .... bit 4 iAx_MasterAxindex_00 – 04

Each axis is given the axis index of its master axis in binary code; the following applies:

- The index of the master axis must be less than 32.
- If the axis is not coupled, the index of the master axis has a value of 0.
- If the axis is the master itself, the index of the master axis is identical with the axis index.

Standstill error iAx_TrqExceed

This signal serves to monitor the standstill torque specified for an axis.

☞ **In the case of Gantry axes, this signal refers to the sum total of the standstill torque values (regardless of the sign) of all the axes of a group of coupled axes. The signal is set only for the master axis.**

Signal level

1: The standstill torque specified for an axis has been exceeded.

0: The standstill torque specified for an axis has not been exceeded.

Tracking error iAx_CoupleLag

This signal serves to monitor the specified coupling lag for a group of coupled axes.

Signal level

1: The maximum coupling lag specified for an axis has been exceeded.

0: The maximum coupling lag specified for an axis has not been exceeded.

Error diagn. class 1 iAx_DrvErrClass1

This signal is a direct map of the corresponding signal in the SERCOS drive status word of the axis in question (the drive switches off and is interlocked).
The interlock can be canceled only by the command ”Reset diagnostics class 1”, which is set automatically by the NC upon a **control reset**.

The pending error can be evaluated manually using the SERCOS monitor or automatically by having the PLC directly read the error.

Parameter S-0-0011 comprises 2 bytes and has the following structure:

Bit 0:	Reserved	
Bit 1:	Amplifier shut-off temperature	S-0-0203
Bit 2:	Motor shut-off temperature	S-0-0204
Bit 3:	Reserved	
Bit 4:	Control voltage fault	
Bit 5:	Feedback error (encoder, measuring system)	
Bit 6:	Error in electronic commutation system	
Bit 7:	Overcurrent	
Bit 8:	Overvoltage	
Bit 9:	Undervoltage fault	
Bit 10:	Phase fault in power supply	
Bit 11:	Excessive controller deviation	S-0-0159
Bit 12:	Communication error	
Bit 13:	Position limit value exceeded	S-0-0049
Bit 14:	Reserved	
Bit 15:	Manufacturer-specific error	S-0-0129

Change diagn. class 2 iAx_DrvChangeClass2

This signal is a direct map of the corresponding signal in the SERCOS drive status word of the axis in question (switch-off prewarning). The bit is set when diagnostic class 2 is modified and is reset after reading the status via parameter S-0-0012. Reading can be carried out manually using the SERCOS monitor or directly by the PLC.

Parameter S-0-0012 comprises 2 bytes and contains the following warnings:

Bit 0:	Reserved	
Bit 1:	Amplifier overtemperature warning	S-0-0311
Bit 2:	Motor overtemperature warning	S-0-0312
Bit 3:	Reserved	
Bit 4:	Reserved	
Bit 5:	Reserved	
Bit 6:	Reserved	
Bit 7:	Reserved	
Bit 8:	Reserved	
Bit 9:	Reserved	
Bit 10:	Reserved	
Bit 11:	Reserved	
Bit 12:	Reserved	
Bit 13:	Reserved	
Bit 14:	Reserved	
Bit 15:	Manufacturer-specific warning	S-0-0181

The effect of switch-off prewarnings concerning the alteration bit can be suppressed on the screen for **diagnostics class 2** (S-0-0097).

Change diagn. class 3 iAx_DrvChangeClass3

This signal is a direct map of the corresponding signal in the SERCOS drive status word of the axis in question (operating status messages). The bit is set when diagnostic class 3 is modified and is reset after reading the status via parameter S-0-0013. Reading can be carried out manually using the SERCOS monitor or directly by the PLC.

Parameter S-0-0013 comprises 2 bytes and contains the following messages:

Bit 0:	nact = ncomm	S-0-0330
Bit 1:	$n_{act} = 0$	S-0-0331
Bit 2:	$\lvert n_{act} \rvert < \lvert n_x \rvert$	S-0-0332
Bit 3:	$\lvert Md \rvert \geq \lvert Md_x \rvert$	S-0-0333
Bit 4:	$\lvert Md \rvert \geq \lvert Md_{lim} \rvert$	S-0-0334
Bit 5:	$\lvert n_{comm} \rvert > \lvert n_{lim} \rvert$	S-0-0335
Bit 6:	In-position	S-0-0336
Bit 7:	$\lvert P \rvert \geq \lvert P_x \rvert$	S-0-0337
Bit 8:	–	
Bit 9:	$\lvert n_{act} \rvert \geq$ min. spindle speed	S-0-0339
Bit 10:	$\lvert n_{act} \rvert \geq$ max. spindle speed	S-0-0340
Bit 11:	In-position rough	S-0-0341
Bit 12:	Target position reached	S-0-0342
Bit 13:	Interpolator halted	S-0-0343
Bit 14:	–	
Bit 15:	Manufacturer-specific operating states	S-0-0182

The effect of operating states concerning the alteration bit can be suppressed on the screen for diagnostics class 3 (S-0-0098).

Torque reduced iAx_TrqLim

This signal indicates that the axis concerned is running at a reduced maximum torque.

Signal level

1: Maximum torque of the axis is reduced.

0: Maximum torque of the axis is not reduced.

Test mode iAx_DryRun

This signal is set when the axis is in Test mode.
In test mode, axes are still interpolated internally; however, the calculated command vlaues are only shown on the display and not transmitted to the drive. The axis is held in the same position where it was when test mode was activated.

Test mode can be activated using a softkey, automatically for drives that are not connected (no SERCOS ring present) or using MP 1001 00010 ”Virtual drive”.

Enabled for power activation iAx_DrvPower

This signal is closely related to the signals **Drive on, Drive ready** and **Drive under control** (see S-0-0135, bit 14/15).
The functional relationship is set out in detail in the signal diagram for the **Drive on** signal (page 5–17) .

Signal level

1: Initialization of the SERCOS interface, the drive and the NC control are complete and no errors are present. This signal is a prerequisite for connecting the drive to power.

0: The power cannot be connected to the drive.

Drive ready iAx_DrvReady

This signal is closely related to the signals **Drive on, Drive connected** and **Drive under control** (see S-0-0135, bit 14/15).
The functional relationship is set out in detail in the signal diagram for the **Drive on** signal (page 5–17) .

Signal level

1: Power is connected to the affected drive. The signal remains set until the drive is cut out.

0: The power is not connected to the drive.

Drive under control iAx_DrvAct

This signal is closely related to the signals **Drive on, Drive connected** and **Drive ready**(see S-0-0135, bit 14/15).
The functional relationship is set out in detail in the signal diagram for the **Drive on** signal (page 5–17) .

Signal level

1: The control loop remains closed and torque is applied on the drive. The axis can be moved unless it is locked by any other function, e.g. **Feed inhibit.**

0: The control loop is open and no torque is applied on the drive.

6 Spindle-related interface signals

6.1 Overview of spindle-related interface signals

Spindle-related interface signals have an effect on those NC functions that are related to the control of a machining spindle.
An interface is available for each projected spindle, i.e. for n spindles, the spindle-related interface is also mapped n times on the corresponding marker.

In the case of spindles assigned to a channel, the channel-related interface is also of importance.

The start addresses of spindle-related interfaces are defined in MACODA parameter 2060 00006.

In the following description, the interface signals are outlined for one spindle.

6.1.1 Overview of output signals (PLC –> NC)

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
0.0	qSp_CAxOn	C axis on	1.0	qSp_TurnCW	Spindle M3 manual
0.1	qSp_CAxOff	C axis off	1.1	qSp_TurnCCW	Spindle M4 manual
0.2	qSp_JogPlus	Spindle jog M3	1.2	qSp_Stop	Spindle M5 manual
0.3	qSp_JogMinus	Spindle jog M4	1.3	qSp_Orientate	Spindle M19 manual
0.4	–	res.	1.4	–	res.
0.5	–	res.	1.5	–	res.
0.6	–	res.	1.6	–	res.
0.7	qSp_Reset	Spindle control reset	1.7	–	res.
2.0	qSp_SafOpModeSel	Mode selection	3.0	qSp_Gear1Act	GTS 1 acknowledgement
2.1	–	res.	3.1	qSp_Gear2Act	GTS 2 acknowledgement
2.2	qSp_SafAgreeButton	Consent key	3.2	qSp_Gear3Act	GTS 3 acknowledgement
2.3	qSp_SafSwitch1	Safety sw. 1 (S1)	3.3	qSp_Gear4Act	GTS 4 acknowledgement
2.4	–	res.	3.4	–	res.
2.5	qSp_ SafCheckInputState	Safety status insp. input	3.5	–	res.
2.6	qSp_SafSignalState	Safety signal status	3.6	–	res.
2.7	–	res.	3.7	qSp_GearIdleAct	Idling acknowledgement
4.0	qSp_ManSpeed_00	Spindle speed jog bit 0	5.0	–	res.
4.1	qSp_ManSpeed_01	Spindle speed jog bit 1	5.1	–	res.
4.2	qSp_ManSpeed_02	Spindle speed jog bit 2	5.2	–	res.
4.3	–	res.	5.3	–	res.
4.4	–	res.	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	–	res.
4.7	–	res.	5.7	qSp_Override100	Spindle override 100%

Spindle-related interface signals

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
6.0	qSp_Override_00	Override bit 0	7.0	qSp_Override_08	Override bit 8
6.1	qSp_Override_01	Override bit 1	7.1	qSp_Override_09	Override bit 9
6.2	qSp_Override_02	Override bit 2	7.2	qSp_Override_10	Override bit 10
6.3	qSp_Override_03	Override bit 3	7.3	qSp_Override_11	Override bit 11
6.4	qSp_Override_04	Override bit 4	7.4	qSp_Override_12	Override bit 12
6.5	qSp_Override_05	Override bit 5	7.5	qSp_Override_13	Override bit 13
6.6	qSp_Override_06	Override bit 6	7.6	qSp_Override_14	Override bit 14
6.7	qSp_Override_07	Override bit 7	7.7	qSp_Override_15	Override bit 15
8.0	qSp_Custom1	Customer input 1	9.0	–	res.
8.1	qSp_Custom2	Customer input 2	9.1	–	res.
8.2	qSp_Custom3	Customer input 3	9.2	–	res.
8.3	qSp_Custom4	Customer input 4	9.3	–	res.
8.4	qSp_Custom5	Customer input 5	9.4	–	res.
8.5	qSp_Custom6	Customer input 6	9.5	–	res.
8.6	qSp_Custom7	Customer input 7	9.6	–	res.
8.7	qSp_Custom8	Customer input 8	9.7	–	res.
10.0	–	res.	11.0	–	res.
10.1	–	res.	11.1	–	res.
10.2	–	res.	11.2	–	res.
10.3	–	res.	11.3	–	res.
10.4	–	res.	11.4	–	res.
10.5	–	res.	11.5	–	res.
10.6	–	res.	11.6	qSp_DrvOn	Drive on
10.7	–	res.	11.7	qSp_DrvLock	Spindle lock

6.1.2 Overview of input signals (NC –> PLC)

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
0.0	iSp_CAxAct	C axis is active	1.0	iSp_ProgSpReach	Speed reached
0.1	iSp_CAxSwitch	C axis switching	1.1	iSp_SpLim	Spindle speed limited
0.2	iSp_TurnCmd	Turn command	1.2	iSp_Stop	Spindle stopped
0.3	iSp_TurnDirM4	Direction of rotation M4	1.3	iSp_OrientateFinish	Spindle orientated
0.4	–	res.	1.4	iSp_OrientateAct	Spindle orientation active
0.5	iSp_InPos	Spindle in position	1.5	–	res.
0.6	iSp_PosCtrl	Position control active	1.6	–	res.
0.7	iSp_Reset	Spindle control reset	1.7	–	res.
2.0	iSp_SafOpMode_00	Bit 0 safety mode	3.0	iSp_Gear1Sel	GTS 1 selection
2.1	iSp_SafOpMode_01	Bit 1 safety mode	3.1	iSp_Gear2Sel	GTS 2 selection
2.2	iSp_SafOpMode_02	Bit 2 safety mode	3.2	iSp_Gear3Sel	GTS 3 selection
2.3	iSp_SafOpMode_03	Bit 3 safety mode	3.3	iSp_Gear4Sel	GTS 4 selection
2.4	iSp_SafStatePos	Safe position status	3.4	iSp_GearChange	GTS change
2.5	iSp_SafCtrlOutputState	Control status output	3.5	iSp_IdleSpeed	Idling speed reached
2.6	–	res.	3.6	–	res.
2.7	–	res.	3.7	iSp_GearIdleSel	Idle gear selection
4.0	–	res.	5.0	–	res.
4.1	–	res.	5.1	–	res.
4.2	–	res.	5.2	–	res.
4.3	–	res.	5.3	–	res.
4.4	–	res.	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	iSp_Override0	Spindle override 0 %
4.7	–	res.	5.7	iSp_Override100	Spindle override 100 %
6.0	iSp_ScsState00	SCS signal status 0	7.0	iSp_ScsState08	SCS signal status 8
6.1	iSp_ScsState01	SCS signal status 1	7.1	iSp_ScsState09	SCS signal status 9
6.2	iSp_ScsState02	SCS signal status 2	7.2	iSp_ScsState10	SCS signal status 10
6.3	iSp_ScsState03	SCS signal status 3	7.3	iSp_ScsState11	SCS signal status 11
6.4	iSp_ScsState04	SCS signal status 4	7.4	iSp_ScsState12	SCS signal status 12
6.5	iSp_ScsState05	SCS signal status 5	7.5	iSp_ScsState13	SCS signal status 13
6.6	iSp_ScsState06	SCS signal status 6	7.6	iSp_ScsState14	SCS signal status 14
6.7	iSp_ScsState07	SCS signal status 7	7.7	iSp_ScsState15	SCS signal status 15
8.0	iSp_Custom1	Customer output 1	9.0	iSp_CoupleIndex_00	No. of coupling bit 0
8.1	iSp_Custom2	Customer output 2	9.1	iSp_CoupleIndex_01	No. of coupling bit 1
8.2	iSp_Custom3	Customer output 3	9.2	iSp_CoupleIndex_02	No. of coupling bit 2
8.3	iSp_Custom4	Customer output 4	9.3	–	res.
8.4	iSp_Custom5	Customer output 5	9.4	–	res.
8.5	iSp_Custom6	Customer output 6	9.5	–	res.
8.6	iSp_Custom7	Customer output 7	9.6	–	res.
8.7	iSp_Custom8	Customer output 8	9.7	–	res.

Spindle-related interface signals

Bit	Symbol. addr.	PLC input signal
10.0	iSp_Master	Spindle is master
10.1	iSp_CoupleErr	Coupling error
10.2	iSp_Synchr1	Synchronization 1
10.3	iSp_Synchr2	Synchronization 2
10.4	–	res.
10.5	–	res.
10.6	–	res.
10.7	–	res.

Bit	Symbol. addr.	PLC input signal
11.0	iSp_DrvErrClass1	Error diagn. class 1
11.1	iSp_DrvChangeClass2	Change diagn. class 2
11.2	iSp_DrvChangeClass3	Change diagn. class 3
11.3	–	res.
11.4	iSp_DryRun	Test mode
11.5	iSp_DrvPower	Enabled for power activation
11.6	iSp_DrvReady	Drive ready
11.7	iSp_DrvAct	Drive under control

6.2 Signal description

6.2.1 Overview of output signals (PLC –> NC)

Depending on their function, the PNC evaluates the signals as either **static signals** with signal levels logic 0 and logic 1 or as **edge-triggered signals** with signal edges 0→1 or 1→0.

C axis on **qSp_CAxOn**

With this signal, the PLC can switch a spindle to C-axis operation.

Application
In special cases (spindle/tool turret), it may be necessary to switch on C-axis operation immediately after starting the control.

Signal edge

0 → 1: The running spindle is decelerated and switched into C-axis operation.
The sequence within the NC is influenced by the settings in MP 1040 00060.

1 → 0: No function.

C axis off **qSp_CAxOff**

With this signal, the PLC can switch a spindle back to spindle operation.

Signal edge

0 → 1: The drive is switched back to spindle mode.
The sequence within the NC is influenced by the settings in MP 1040 00060.

1 → 0: No function.

Spindle jog M3 qSp_JogPlus
Spindle jog M4 qSp_JogMinus

These signals have an effect in NC mode **Setup**; their functions work together with the coded signals Jog speed bit 0 ... Bit 2.

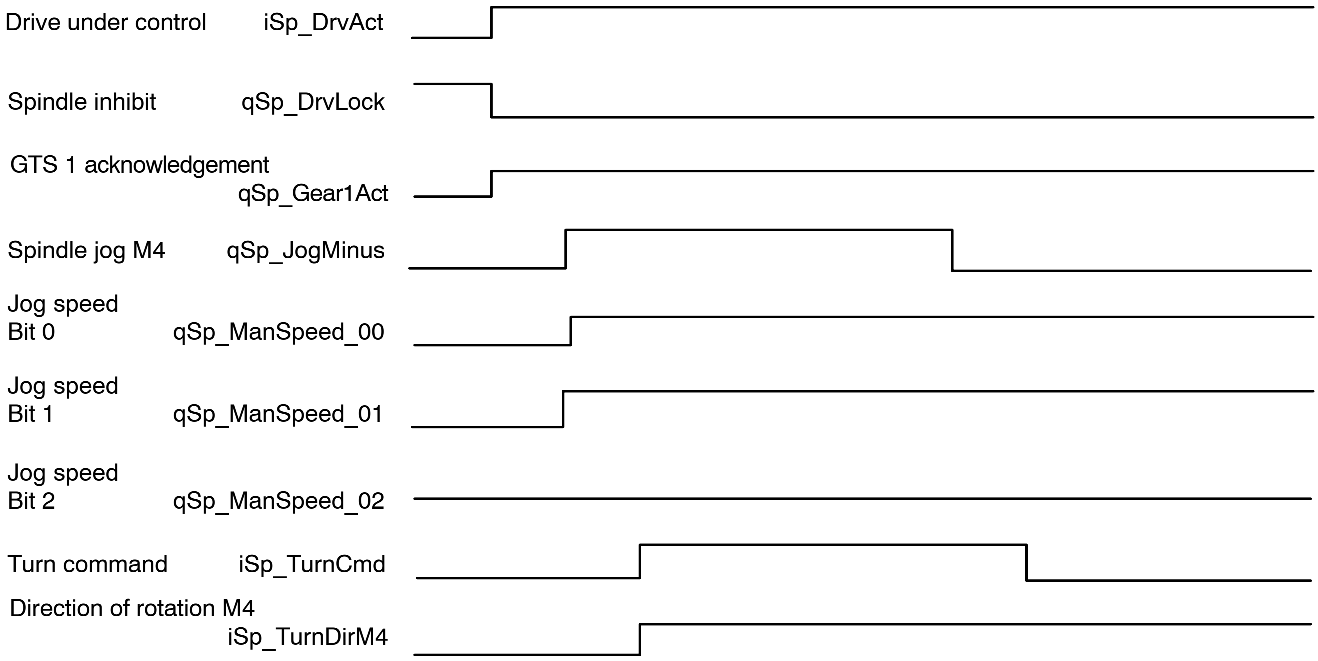
Signal level

1: Speed specification using the coded Jog speed bit 0 – bit 2 signals is required. Furthermore, the control loop must be closed (Drive under control is set), spindle inhibit must be reset and a gear range must be activated. As long as the **Spindle jog M3** or **Spindle jog M4** signal is set, the spindle rotates at the respective speed in the selected direction of rotation.

0: The spindle is decelerated until it comes to a standstill; the control loop remains closed.

Example:

In Setup mode, the spindle is to be traversed in the clockwise direction at a predefined speed.


Drive under control iSp_DrvAct
Spindle inhibit qSp_DrvLock
GTS 1 acknowledgement qSp_Gear1Act
Spindle jog M4 qSp_JogMinus
Jog speed Bit 0 qSp_ManSpeed_00
Jog speed Bit 1 qSp_ManSpeed_01
Jog speed Bit 2 qSp_ManSpeed_02
Turn command iSp_TurnCmd
Direction of rotation M4 iSp_TurnDirM4

The control loop is closed (**Drive under control** is set), **Spindle inhibit** is reset and gear range 1 (**Gear range 1 acknowledgement**) is activated.

At the same time that the **Spindle jog M3** signal is set, a spindle speed must be input (**Spindle speed jog bits 0 – 2**). The jogging speed is defined for each spindle individually in machine parameter 1040 00015. The spindle command is set by the NC control. **The direction of rotation M4** is output in conjunction with the spindle command.

The spindle accelerates to the preselected spindle speed because the spindle inhibit has already been reset.

If the **Spindle inhibit** signal is still set when the spindle command is output, the direction of rotation M4 will not be output until the spindle inhibit is reset.

As soon as the **Spindle jog** signal is reset, the spindle decelerating cycle is initiated. When the spindle has come to a standstill, the **Spindle command** signal is reset.

CAUTION
In normal operation, the spindle will generally be traversed in controlled mode only, not in the position control circuit. This means that it is possible that the spindle command has been reset while the spindle is still decelerating. In order to ensure that the spindle is stationary, the ”Spindle stopped” signal must also be queried.

Spindle control reset qSp_Reset

Using this signal, the current movement input or an active command (e.g. spindle orientation) can be canceled for the respective spindle. The spindle acknowledges the receipt of the reset instruction by setting the **Spindle reset signal** (iSp_Reset).

Spindle M3 manual qSp_TurnCW
Spindle M4 manual qSp_TurnCCW

Alternatively to the NC part program (programming M3, M4), these signals are used to determine the sense of rotation of the spindle:

- Spindle M3 manual:spindle clockwise
- Spindle M4 manual:spindle counterclockwise.

In this regard, signal **Spindle M5 manual** also has significance (see the example on page 6–8).

Signal edge

0 → 1: Using this edge, a spindle speed and the corresponding sense of rotation is input for the spindle drive. The spindle will start to rotate if the control loop is closed (**Drive under control**) and no **spindle inhibit** signal is set.
The spindle speed corresponds to the speed programmed last.

1 → 0: No effect.

Spindle M5 manual qSp_Stop

Alternatively to the NC part program (programming M5), this signal is used to activate the spindle stop function.

In this regard, signals Spindle M3 manual and Spindle M4 manual also have significance.

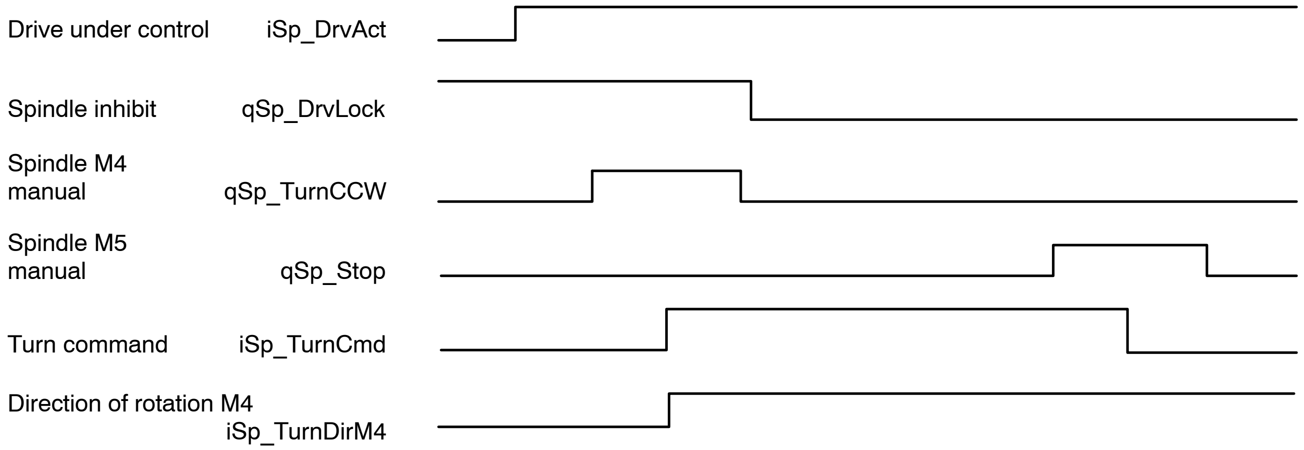
Signal edge

0 → 1: The spindle stops with the parameterized deceleration. The gear range and spindle speed that were active previously remain stored in the NC. The control loop remains closed and torque is still applied on the spindle drive.

1 → 0: No effect.

Example:

The spindle is to be accelerated manually in the counterclockwise direction up to the speed that was programmed last.



The **Drive under control** signal indicates that the control loop is closed and torque is applied on the spindle.

When the **Spindle M4 manual** signal is set, the speed programmed last is transmitted to the spindle drive.
As a result of this speed input, the NC sets the **Spindle command** and **Direction of rotation M4** signals.

When the **Spindle command** signal is output, the **Spindle M4 manual** signal can be reset. The spindle does not rotate while **Spindle inhibit** is set.

Only when the **Spindle inhibit** signal has been reset will the spindle start rotating and the **Spindle rotating** signal is set. By setting the **Spindle M5 manual** signal, the rotation of the spindle is stopped and the **Spindle command** and **Direction of rotation M4** signals are reset by the NC.

Spindle M19 manual qSp_Orientate

Alternatively to the NC part program (M19), this signal calls up the ”Spindle orientation” function. For this purpose, the control loop must be closed and the drive under control.
In this regard, signals **Spindle in position, Spindle orientation active** and **Spindle orientated** have significance.

☞ **This function is not available for analog spindles (MP 1040 00001).**

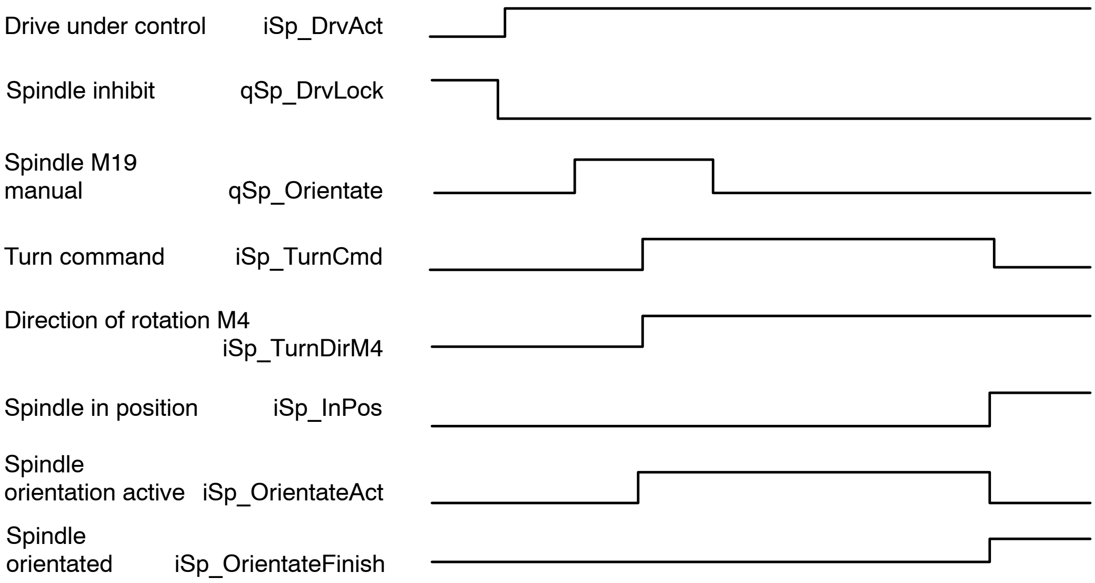
Signal edge

0 → 1: Spindle orientation is initiated. Spindle orientation is always performed on the zero pulse of the measuring system. The direction of rotation depends on SERCOS parameter S-0-0154.

1 → 0: No effect.

Example:

The spindle is to be oriented manually; the control loop is closed.


Drive under control iSp_DrvAct
Spindle inhibit qSp_DrvLock
Spindle M19 manual qSp_Orientate
Turn command iSp_TurnCmd
Direction of rotation M4 iSp_TurnDirM4
Spindle in position iSp_InPos
Spindle orientation active iSp_OrientateAct
Spindle orientated iSp_OrientateFinish

The control loop is closed (**Drive under control** is set) and **Spindle inhibit** is reset.

When the signal **Spindle orientation manual** is set, the spindle receives the command to rotate; **Spindle command** is set.
Depending on the direction of rotation, **Direction of rotation M4** is set when the spindle is rotating counterclockwise (M4).

When the **Spindle command** signal is output, the **Spindle orientation manual** signal can be reset.
When the spindle is rotating, **Spindle orientation active** is set.
Once the spindle has been oriented in line with the marker position, signals **Spindle orientated** and **Spindle in position** are set and the **Spindle command** signal is reset.

Mode selection qSp_SafModeSel

This signal is a component of the safety technology for the IndraDrive series.
The operating mode selection of the PLC is transferred as channel 1 to the drive via the SERCOS interface in real time.

Signal level

1: Normal mode NO

0: Special mode SO
(special mode at standstill / with movement)

Consent key qSp_SafAgreeButton

This signal is a component of the safety technology for the IndraDrive series.
Switching to ”Special mode with movement” via the PLC is transferred as channel 1 to the drive via the SERCOS interface in real time.

Signal level

1: Special mode with movement

0: Special mode at standstill

Safety sw. 1 (S1) qSp_SafSwitch1

These signals are components of the safety technology for the IndraDrive series.
Two different parameter sets can be stored in the drive for ”Special mode with movement”. The parameter set selected using the PLC is transferred as channel 1 to the drive via the SERCOS interface in real time.

Signal level

1: Parameter set 2 for safe movement active

0: Parameter set 1 for safe movement active

Safety status insp. input qSp_SafCheckInputState

This signal is a component of the safety technology for the IndraDrive series.
The safe status signal set using the PLC is transferred as channel 1 to the drive via the SERCOS interface in real time.

Safety signal status qSp_SafSignalState

This signal is a component of the safety technology for the IndraDrive series.
The control signals of the PLC (qAx_Saf...) that are transferred via channel 1 to the drive must be adjusted periodically so that they can be checked. As a result, the PLC sets the signals to ”0” for one cycle. At the same time, the PLC sets the safety signal status from ”valid” to ”adjusted”, thus informing the drive that the control signals are invalid.
Then the signals are reset.

Signal level

1: The control signals of the safety technology are being adjusted

0: The control signals of the safety technology are valid

GTS 1 – 4 acknowledgement qSp_Gear1Act – 4Act
Idling acknowledgement qSp_GearIdleAct

The current gear range is reported using these signals. In the case of a gear range inconsistency, ”idling” mode is activated when the next spindle command is output.
At control startup, the acknowledged gear range is adopted in the NC.

Analog spindle (104000001) gear changes are not supported. In the case of analog spindles, GTS 1 must be acknowledged.

Spindle speed jog bit 0 – bit 2 qSp_ManSpeed_00 – 02

These signals are effective exclusively in conjunction with the **Spindle jog M3** or **Spindle jog M4** signals.

Depending on the coding, the spindle speed defined in MP 1040 00015 will be activated. If the preselected spindle speed is not in the range of the current gear range, it will be limited to the minimum or maximum spindle speed of the activated gear range.

Machine parameters	**Jog speed**		
	Bit 2 (qSp_ ManSpeed_02)	**Bit 1 (qSp_ ManSpeed_01)**	**Bit 0 (qSp_ ManSpeed_00)**
1040 00015 [1]	0	0	0
1040 00015 [2]	0	0	1
1040 00015 [3]	0	1	0
1040 00015 [4]	0	1	1
1040 00015 [5]	1	0	0
1040 00015 [6]	1	0	1
1040 00015 [7]	1	1	0
1040 00015 [8]	1	1	1

Spindle override 100% qSp_Override100

This signal suppresses the spindle override. The spindle then turns at the programmed speed.

Signal edge

0 → 1: The spindle override is suppressed.

1 → 0: The spindle override is in effect.

Spindle-related interface signals

Override bit 0 – bit 15 qSp_Override_00 – 15

The current spindle speed can be influenced by using the **Override function**. The Override function affects the speeds programmed in the NC part program and the speed in Set-up mode.
There are two procedures for this:

- The interface is used to activate a **binary-coded** override switch in the PNC with a maximum of 32 settings.
  Each binary code is assigned a value via MACODA parameter 1040 00010 which represents the respective current speed in percent.
- If all the settings in MACODA parameter are 0 (except 1: Override value), the PNC interprets the 16-bit preset value directly as an override value in 0.01%.

Customer input 1 – 8 qSp_Custom1 – 8

Interface that can be configured on the NC side and can be assigned as required.
(This option is currently not yet active.)

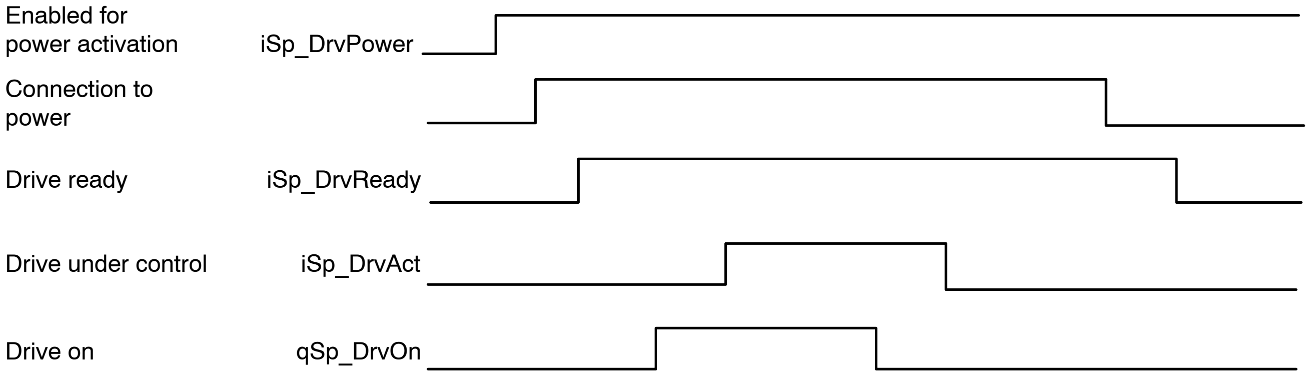
Drive on qSp_DrvOn

This signal switches on the torque in the drive when the drive is ready for operation (iSp_DrvReady is set). The control circuit is closed.

Resetting **Drive on** initiates a shutdown process. If the axis is still in motion at this time, it is brought to a halt at the preset deceleration.
When the spindle has come to a standstill, the control loop is opened and the **Drive under control** signal is reset. Torque is no longer applied on the drive.

☞ **The entire sequence does not apply to analog spindles.**

Spindle-related interface signals



Example: Functioning of the starting and shutdown cycles

When the machine is started or voltage is applied, the NC unit and the servo and spindle drives go through an initialization phase.
While the initialization phase of the individual components is running, the digital interface for communication between the NC and the drives is established, among other things.

This digital interface corresponds to the settings of the **SERCOS interface.**
Initialization of the **SERCOS interface** is in-system, automatic and consists of 4 phases. Once the interface is initialized and cyclic operation is active, the **Enabled for power activation** signal is output.
Power can be switched on for the affected drives. When power has been switched on, the **Drive ready** signal is output.
The position control loop is closed by setting **Drive on**. The drive is under torque and the **Drive under control** signal is issued.

The shutdown cycle or disconnection from the system is initiated by the PLC sequential program resetting the -Drive off **signal**.
If the axis is still in motion at this time, it is brought to a halt at the preset deceleration.
When the axis has come to a standstill, the position control loop is opened and the **Drive under control** signal is reset. Torque is no longer applied on the drive.

The power can be switched off in the next step.
The **Enabled for power activation** signal remains set as long as the machine components (NC and drives) remain initialized.

Spindle lock qSp_DrvLock

This signal is effective in any mode when the control loop is closed and torque is applied to the spindle.

It prevents the execution of a programmed or manually entered spindle movement. If Spindle inhibit is set, the idling speed is not carried out. The signals Spindle command, Direction of rotation M4 or Spindle in position are not affected.

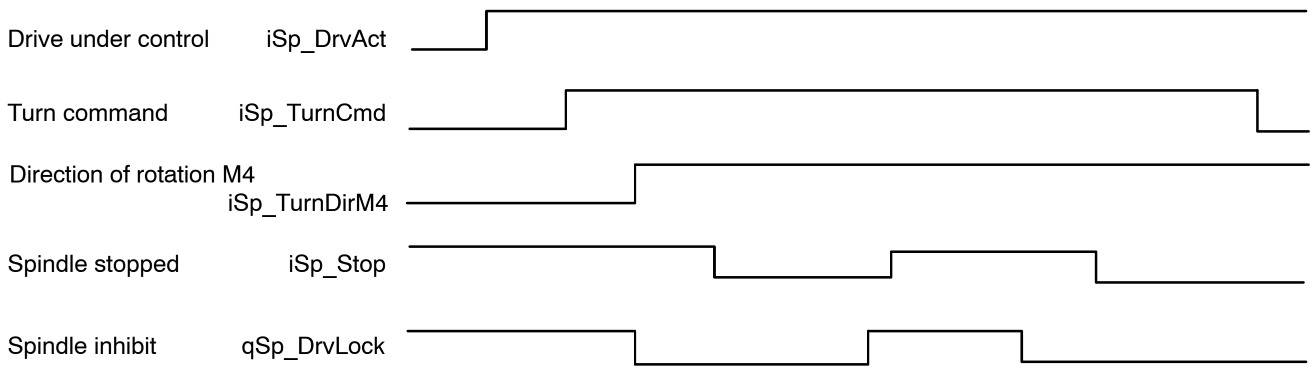
Signal level

1: The spindle cannot be turned. If a rotating movement is already active, the spindle will be stopped at the parameterized deceleration.

0: Spindle rotation is not inhibited. If a spindle command is initiated (via the NC part program or manually) and the appropriate gear range is active, the spindle can start rotating immediately. If a rotating spindle was brought to a standstill with Spindle inhibit , this spindle will now start accelerating to the initial speed.

Example:

The machine is switched on and the control loop has been closed. A rotary motion, e.g. M3 S1000, has been programmed via the NC part program.



The **Drive under control** signal informs the PLC sequential program that the control loop is closed. When the NC block in which the rotary movement was programmed is executed, the **Spindle command** signal is output.

As long as **Spindle inhibit** is set, the rotary movement will not be carried out; signal **Direction of rotation M4** is not set and signal **Spindle stopped** is set. The direction of rotation can be queried only if a spindle command has been set.

When **Spindle inhibit** is reset, the programmed rotary movement is executed, the **Direction of rotation M4** is set according to the movement and **Spindle stopped** is reset.

If **Spindle inhibit** is set again during the rotary movement, the rotary movement is stopped and **Spindle stopped** is set when the spindle has come to a standstill.

When **Spindle inhibit** is reset, the spindle accelerates to the previously applied speed and **Spindle stopped** is reset.

The Spindle **command** signal is reset when the spindle is stopped with the **M5** spindle stop function (NC part program or interface signal).

6.2.2 Input signals (NC –> PLC)

C axis is active iSp_CAxAct

This signal indicates whether the respective spindle is in C-axis mode (position interface) or in spindle mode (speed interface). Spindles which can be switched over to C-axis operation have a **normal axis interface** for C-axis operation **in addition** to the spindle interface. The C-axis is active signal indicates the relevant interface.

Signal level

1: The spindle is in C-axis mode: the axis interface is valid, the spindle interface is not served by the NC.

0: The spindle is in spindle mode: the spindle interface is served by the NC, the axis interface is not valid.

C axis switching iSp_CAxSwitch

This signal indicates that the spindle is currently in the switching phase between Spindle and C-axis operation.
The **Spindle command** signal is always output at the same time.

Signal level

1: The spindle is in the switching phase.

0: The spindle is not in the switching phase.

Turn command iSp_TurnCmd

The **Spindle command, Direction of rotation M4, Spindle stopped** and **Spindle in position** signals affect each other mutually.

Examples of their relationships are shown in the signal diagrams and explanations for the Spindle M3 manual, Spindle M4 manual, Spindle M5 manual (see page 6–8) and Spindle inhibit (see page 6–15) signals.

The **Direction of rotation M4** signal is output in conjunction with the **Spindle command**.

Signal level

1: The spindle is to carry out a rotary motion, which is input either **manually** (e.g. Spindle M3 manual) or via the **NC part program** (e.g. M3 S1500). A prerequisite is that the control loop is closed, i.e. the **Drive under control** signal must have been set.

0: Spindle rotation was stopped by a programmed M5 spindle stop function or by a Spindle M5 manual signal (qSp_Stop). In addition, the Spindle command signal is not set if **spindle orientation** or **positioning** or **gear range switching** is active.

Direction of rotation M4 iSp_TurnDirM4

This signal is effective only in conjunction with the **Spindle command** signal, i.e. the **Direction of rotation M4** and **Spindle command** signals must be linked by an AND element in the PLC sequential program.

In the absence of a spindle command signal, the Direction of rotation M4 signal is irrelevant.

No Direction of rotation M4 signal is output – or its status remains unchanged – for the functions Gear range switching, Idling speed and Spindle orientation.

Signal level

1: When Direction of rotation M4 and Spindle command are set, the spindle should carry out a rotary movement in the **counterclockwise direction (M4)**.

0: When Direction of rotation M4 not set and Spindle command is set, the spindle should carry out a rotary movement in the **clockwise direction (M3)**.

Spindle in position iSp_InPos

This signal is relevant only in conjunction with the **Spindle orientation** or **Spindle positioning** functions.
Prerequisite: The control loop is closed and torque is applied to the spindle (signal **Drive under control**is set).

Signal level

1: The spindle has been orientated or positioned and is located in the in-position window that is defined in the drive.

0: The spindle has not been oriented and/or is not located in the parameterized in-position window.

Position control active iSp_PosCtrl

Signal level

1: The spindle is activated via the position interface.

0: The spindle is activated via the speed interface.

Spindle control reset iSp_Reset

This signal indicates that the spindle has accepted the control reset task and that the spindle is again ready to accept new inputs for motion.

Speed reached iSp_ProgSpReach

Signal level

1: The spindle speed is identical with the programmed spindle speed and/or is within the speed tolerance range as defined for every spindle by the superposition of MP1040 00020 (Spindle speed window in rpm) and 1040 00021 (Spindle speed window in %).
If the speed is limited by programmed speed limits or gear range speed limits, this signal is output when the set limit is reached. In addition, the position of the override switch is taken into account.

0: The spindle speed is not within the speed tolerance range.

Spindle speed limited iSp_SpLim

This signal indicates that the programmed spindle speed is limited either by the active gear range or by the programmed spindle speed limit (SMin, SMax).

Signal level

1: The programmed spindle speed is limited.

0: The programmed spindle speed is not limited.

Spindle stopped iSp_Stop

This signal relates to the speed of the spindle.

Signal level

1: M5 has been executed (using the NC part program or interface), speed 0 (S0) has been set via the NC part program or the **Spindle inhibit** or **Override 0** signals have been set.

0: The spindle is rotating.

☞ **This signal is not generated for analog spindles!**

Spindle orientated iSp_OrientateFinish

Signals **Spindle orientation active** and **Spindle in position** have significance together with this signal. The functional relationship is set out in detail in the signal diagram on page 6–9.

Signal level

1: The Spindle orientation function has been completed and the spindle is in the oriented stop position.

0: The spindle orientation function has been completed but the spindle is no longer in its previous ”oriented stop” position.

Spindle orientation active iSp_OrientateAct

This signal is effective only in conjunction with the **Spindle orientation** function. The functional relationship is set out in detail in the signal diagram on page 6–9.

Signal level

1: The Spindle orientation function has been activated using the NC part program (e.g. by programming M19) or using the qSp_Orientate signal.

0: Spindle orientation is not activated.

Safety mode, bit 0 ... bit 3 iSp_SafOpMode_00 – 03

These signals are components of the safety technology for the IndraDrive series.
The control reads the selected mode of the drive (P-0-3215) and transfers it to the PLC using these outputs.

Safety function	Safety mode			
	Bit 3 (iAx_ SafOpMode_00)	Bit 2 (iAx_ SafOpMode_00)	Bit 1 (iAx_ SafOpMode_00)	Bit 0 (iAx_ SafOpMode_00)
Normal mode NO	0	0	0	0
Safe starting lockout (ASP)	0	0	0	1
Safe stop (SH) / Safe mode stop (SBH)	0	0	1	0
Safe movement (SBB1)	0	0	1	1
Safe movement (SBB2)	0	1	0	0
Safe movement (SBB3)	0	1	0	1
Safe movement (SBB4)	0	1	1	0
Safety technology not active	1	1	1	1

Safe position status iSp_SafStatePos

This signal is a component of the safety technology for the IndraDrive series.
A prerequirement for the ”Safe absolute position” safety function is that the drive uses this signal to report whether safe referencing has been carried out.

Signal level

1: The drive is safely referenced.

0: No safe referencing

Control status output iSp_SafCtrlOutputState

This signal is a component of the safety technology for the IndraDrive series.
The drive uses this signal to provide feedback about its safe status.

Signal level

1: Safe status of the drive

0: No safe status of the drive

Selection for GTS 1 – 4 iSp_Gear1Sel – 4Sel

Depending on the gear range selection (manual or via the NC part program), the preselected gear range is output.

The gear range can be selected directly by programming **M41 – M44** (gear range 1 – gear range 4) via the NC part program. The gear range is derived from the programmed **speed S ....** in conjunction with **automatic gear range recognition M40**.
If the speed ranges in the individual gear ranges overlap, the gear range with the higher spindle drive speed is output.

In the context of gear range switching, the **Idling speed reached** signal is of importance.

Behavior after power-up:

After the power-up or start-up of the control, the gear range which is reported as an **acknowledgement** by the PLC program via the relevant inputs is output. If no gear range is acknowledged, no gear range will be output either. In this case, no gear range is output until a gear range is selected manually or via the NC part program.
If **automatic gear range recognition M40** is active upon power-up, the gear range is output only after a **speed S...** has been programmed in conjunction with a **direction of rotation M3** or **M4.**

Signal edge

0 → 1: The change of gear is to be performed with this edge. In order to minimize the mechanical impact of the gear change, the **Spindle idling** function (if available in the spindle drive) is activated in the spindle drive via the SERCOS interface. As soon as the idling speed has been reached, the NC sets the **Idling speed reached** signal so that the mechanical switching procedure can be carried out. When the gear-switching operation has been executed, the current gear range must be signaled back to the relevant **GTS 1 – GTS 4 acknowledgement** interface. Activation of the respective parameter block via the SERCOS interface concludes the gear-switching operation.

1 → 0: The gear range selected previously is now deselected.

GTS change — iSp_GearChange

This signal is effective only in conjunction with the gear switchover function, triggered by **M functions M40, 41, 42**.

Signal level

1: A change of gear range is executed.

0: The change of gear range has been completed.

Idling speed reached — iSp_IdleSpeed

This signal is effective in conjunction with the gear switchover function and the feedback signal of the current gear range.
The Spindle override setting is not taken into account when the idling speed is output.

Signal level

1: The spindle speed corresponds to the idling speed. This signal is normally used to switch gear ranges.

0: The idling speed has not yet been reached.

Idle gear selection iSp_GearIdleSel

Depending on the gear range selection (manual or via the NC part program), the preselected gear range or idling setting is output.

Idling can be selected directly by programming the **M48** function in the NC part program. Neither a **M3** or **M4** direction of rotation, **Spindle orientation M19** nor a spindle speed **S...** may be programmed in conjunction with **M48**.

In this context, the **Idling speed reached** signal is also of importance.

Signal edge

0 → 1: This signal edge is used to switch the gear to the idle position. In order to minimize the mechanical impact of the gear change, the **Spindle idling** function (if available in the spindle drive) is activated in the spindle drive via the SERCOS interface.
As soon as the idling speed (possibly accompanied by spindle-gear meshing) is reached, the NC sets the **Idling speed reached** signal so that the mechanical switching operation can be carried out.
After the switching operation is complete, the idling position of the gear must be fed back to the **Idle gear acknowledgement** interface; gear switching is thus complete.

1 → 0: The gear switching operation in the idling position is deselected.

Spindle override 0% iSp_Override0

This signal indicates that the override of the spindle is set to 0% and that no Override 100% has been selected using the interface (qSp_Override100).

Signal level

1: 0% acts on the spindle.

0: A value not equal to 0% acts on the spindle.

Spindle override 100% iSp_Override100

This signal indicates that the override of the spindle is set to 100% and/or that Override 100% has been selected using the interface (qSp_Override100).

Signal level

1: A value of 100% acts on the spindle.

0: A value not equal to 100% acts on the spindle.

SCS-Signalstatus 0 ... 15 — iSp_ScsState00 – 15

The bit signals as configured in the cyclic telegram from the spindle drive are transferred to the NC in the signal status word (SERCOS parameter S-0-0144).

The status signals can be configured in parameter S-0-0026. Parameter S-0-0144 must be included in the configuration list of the drive telegram (S-0-0016) to transfer the signals in the cyclic telegram.

Configuration example:

- S-0-0026 = (330, 331,0, 333, 0, 335, 336, 337, 0, 0, 0, 0, 0, 0, 0, 0):
  Configuration of the signal status word
- S-0-0016 = (40, 144):
  Configuration of the drive telegram

The following status messages are then allocated to the individual signals:

iSp_ScsState00	: S-0-0330	Message $n_{act} = n_{comm}$
iSp_ScsState01	: S-0-0331	Message $n_{act} = 0$
iSp_ScsState01	: -	
iSp_ScsState01	: S-0-0333	Message $M_d \geq M_{dx}$
iSp_ScsState04	: -	
iSp_ScsState05	: S-0-0335	Message $n_{comm} > n_{limit}$
iSp_ScsState06	: S-0-0336	”In position” message
iSp_ScsState07	: S-0-0337	Message $P \geq P_x$
iSp_ScsState08	: -	
iSp_ScsState09	: -	
iSp_ScsState10	: -	
iSp_ScsState11	: -	
iSp_ScsState12	: -	
iSp_ScsState13	: -	
iSp_ScsState14	: -	
iSp_ScsState15	: -	

☞ **For a detailed description of all status messages available, please see the SERCOS interface Specifications.**

Customer output 1 – 8 — iSP_Custom1 – 8

Interface that can be configured on the NC side and can be assigned as required.
(This option is currently not yet active.)

No. of coupling bit 0 – bit 2 — iSp_CoupleIndex_00 – 02

When spindle coupling is active, the number of the coupling is output for each member of a group of coupled spindles using this signal.

Spindle is master iSp_Master

This signal indicates the master of a group of coupled spindles.

Signal level

1: The spindle is the master spindle.

0: The spindle is a slave spindle.

Coupling error iSp_CoupleErr

When spindle coupling is active (see iSp_CoupleIndex_00 – 02), this signal supplies information about the current status of the group of coupled spindles for master spindles (iSp_Master is set).

Signal level

1: The group of coupled spindles is in a fault status. This error can be cleared by a ”master spindle control reset” or by an ”overall control reset”.

0: The status of the group of coupled spindles is normal.

Synchronization 1 iSp_Synchr1

When spindle coupling is active (see iSp_CoupleIndex_00 – 02), this signal supplies the following information about the **angular offset** for slave spindles (iSp_Master is not set):

Signal level

1: The synchronous mode window is being complied with and a programmed angular offset has been executed.

0: The synchronous mode window is not being complied with or a programmed angular offset is just being executed.

Synchronization 2 iSp_Synchr2

When spindle coupling is active (see iSp_CoupleIndex_00 – 02), this signal supplies the following information about the **fault window** for slave spindles (iSp_Master is not set):

Signal level

1: The synchronous mode error window is being complied with.

0: The synchronous mode error window is not being complied with.

Error diagn. class 1 iSp_DrvErrClass1

This signal reports a fault of diagnostics class 1 (drive cutoff with interlock) and is a direct map of the corresponding signal in the SERCOS drive status word of the drive in question. The interlock can be canceled only by the command ”Reset diagnostics class 1”, which is set automatically by the NC upon a control reset.

The pending error can be evaluated manually using the NC operator interface or automatically by having the PLC directly read S-0-0011.

Parameter S-0-0011 comprises 2 bytes and has the following structure:

Bit 0:	–	
Bit 1:	Amplifier shut-off temperature	S-0-0203
Bit 2:	Motor shut-off temperature	S-0-0204
Bit 3:	–	
Bit 4:	Control voltage fault	
Bit 5:	Feedback error (encoder, measuring system)	
Bit 6:	Error in electronic commutation system	
Bit 7:	Overcurrent	
Bit 8:	Overvoltage	
Bit 9:	Undervoltage fault	
Bit 10:	Phase fault in power supply	
Bit 11:	Excessive controller deviation	S-0-0159
Bit 12:	Communication error	
Bit 13:	Position limit value exceeded	S-0-0049
Bit 14:	–	
Bit 15:	Manufacturer-specific error	S-0-0129

☞ **This signal is not generated for analog spindles (1040 00001).**

Signal level

1: A diagnostics class 1 error has occurred.

0: There is no diagnostics class 1 error.

☞ **Refer to the SERCOS Interface Specifications for more information on diagnostics class 1 errors.**

Change diagn. class 2 iSp_DrvChangeClass2

This signal reports a change of diagnostics class 2 (switch-off prewarning) and is a direct map of the corresponding signal in the SERCOS drive status word of the drive in question.
The bit is set when diagnostics class 2 is changed; it is reset after reading the status using parameter S-0-0012. Reading can occur manually using the NC operator interface or automatically by having the PLC directly read S-0-0011.

Parameter S-0-0012 comprises 2 bytes and contains the following **warnings**:

Bit 0:	–	–
Bit 1:	Amplifier overtemperature warning	S-0-0311
Bit 2:	Motor overtemperature warning	S-0-0312
Bit 3: through 14:	–	
Bit 15:	Manufacturer-specific warning	S-0-0181

The effect of switch-off prewarnings concerning the alteration bit can be suppressed on the screen for diagnostics class 2 (S-0-0097).

☞ **This signal is not generated for analog spindles (1040 00001).**

Change diagn. class 3 iSp_DrvChangeClass3

This signal reports a change of diagnostics class 3 (messages concerning operating statuses) and is a direct map of the corresponding signal in the SERCOS drive status word of the drive in question. The bit is set when diagnostics class 3 is changed; it is reset after reading the status using parameter S-0-0013. Reading can occur manually using the NC operator interface or automatically by having the PLC directly read S-0-0013.

Parameter S-0-0013 comprises 2 bytes and contains the following messages:

Bit 0:	$n_{act} = n_{comm}$	S-0-0330
Bit 1:	$n_{act} = 0$	S-0-0331
Bit 2:	$\lvert n_{act} \rvert < \lvert n_x \rvert$	S-0-0332
Bit 3:	$\lvert Md \rvert \geq \lvert Md_x \rvert$	S-0-0333
Bit 4:	$\lvert Md \rvert \geq \lvert Md_{lim} \rvert$	S-0-0334
Bit 5:	$\lvert n_{comm} \rvert > \lvert n_{lim} \rvert$	S-0-0335
Bit 6:	In-position	S-0-0336
Bit 7:	$\lvert P \rvert \geq \lvert P_x \rvert$	S-0-0337
Bit 8:	–	
Bit 9:	$\lvert n_{act} \rvert \geq$ min. spindle speed	S-0-0339
Bit 10:	$\lvert n_{act} \rvert \geq$ max. spindle speed	S-0-0340
Bit 11:	In-position rough	S-0-0341
Bit 12:	Target position reached	S-0-0342
Bit 13:	Interpolator halted	S-0-0343
Bit 14:	–	
Bit 15:	Manufacturer-specific operating states	S-0-0182

The effect of operating states concerning the alteration bit can be suppressed on the screen for diagnostics class 3 (S-0-0098).

☞ **This signal is not generated for analog spindles (1040 00001).**

Test mode iSp_DryRun

This signal indicates whether a spindle has been switched to test mode. In test mode, a spindle is still interpolated internally; however, the calculated command values are only shown on the display and not transmitted to the drive.

Test mode can be activated using a softkey, automatically for drives that are not connected (no SERCOS ring present) or using MP 1001 00010 ”Virtual drive”.

Enabled for power activation iSp_DrvPower

This signal is closely related to the signals **Start inhibitor, Drive on, Drive ready** and **Drive under control.**
The functional relationship is set out in detail in the signal diagram for the **Start inhibitor** signal (page 6–14).

Signal level

1: Initialization of the SERCOS interface, the spindle drive and the NC control is complete and no errors exist. This signal is a prerequisite for connecting the drive to power.

0: The power cannot be connected to the drive.

☞ **This signal is not generated for analog spindles (1040 00001).**

Drive ready iSp_DrvReady

This signal is closely related to the signals Start inhibitor, **Drive on, Enabled for power activation** and **Drive under control**.
The functional relationship is set out in detail in the signal diagram for the **Start inhibitor** signal (page 6–14) .

Signal level

1: Power is connected to the affected drive. The signal remains set until the drive is cut out.

0: The power is not connected to the drive.

☞ **This signal is not generated for analog spindles (1040 00001).**

Drive under control iSp_DrvAct

This signal is closely related to the signals **Start inhibitor, Drive on, Enabled for power activation** and **Drive ready**.
The functional relationship is set out in detail in the signal diagram for the Start inhibitor signal (page 6–14).

☞ **This signal is not generated for analog spindles (1040 00001).**

Signal level

1: The control loop remains closed and torque is applied to the drive. The spindle can be rotated unless it is locked by any other function, e.g. **Spindle inhibit.**

0: The control loop is open and no torque is applied on the drive.

☞ **When changing from logic 1 to logic 0, spindle stop (M5) is triggered within the NC.**

Notes:

7 External spindle speed

If an external spindle is defined within the system, the actual spindle speed is transferred for display on the control panel by the PLC program. The PLC program writes the spindle speed value to an area within the PLC operands. This area is defined by MACODA parameters 2060 00002 [8] (operand type) and 2060 00003 [8] (operand address). Every spindle speed value uses 4 bytes.

If several external spindles are defined, their speeds are stored consecutively under this address.

The NC reads the speed to be displayed on the operator panel from this area.

External spindle speed

Notes:

8 Auxiliary functions

Auxiliary functions are programmed in the NC part program.

All **bit-coded** auxiliary functions programmed in **one** NC block are mapped in parallel via the APS to markers.

Parallel data transfer helps avoid machine downtimes.

Auxiliary function output is cyclic with each PLC cycle provided an auxiliary function has been programmed in the NC part program.

The following conventions apply to auxiliary functions:

- Every channel has several auxiliary functions. The designations (types) of these auxiliary functions (M, S, T, etc.) are defined in MACODA parameter 30100 0030.
- An auxiliary function can be mapped as a
  - **bit-coded auxiliary function**
  - **32-bit BCD-coded auxiliary function**
  - **64-bit BCD-coded auxiliary function**
  - **64-bit BCD-coded auxiliary function**
  - **64-bit BCD-coded auxiliary function**

  by the APS to an operand range (MACODA parameter 3010 00040).
- Several function types can be declared as bit-coded auxiliary functions for each channel (MACODA parameter 3010 00030), e.g. the auxiliary function with the designation **M** and the auxiliary function with the designation **H.**
- The start addresses of bit-coded auxiliary functions are set individually for each channel in MACODA parameter 2060 00009.

8.1 Bit-coded auxiliary functions

Auxiliary functions are most often required in the PLC sequential program in the form of individual signals. In this way, for example, the coolant is switched on/off, a clamp is opened/closed or a cylinder is extended/retracted.

Auxiliary functions are programmed in the NC part program in order to trigger these functions. The decoded auxiliary functions are mapped on an operand range (e.g. markers). The absolute size of this range is dependent on the number of auxiliary functions which are to be decoded as well as on the number of channels (MACODA parameter 9040 00001).

The operand type (e.g. marker) is defined in MACODA parameter 2060 00002[5].

The start address of the operand area is defined individually for each channel in MACODA parameter 2060 00009.

Example:

3 channels are defined in the NC; for each channel, 128 (MACODA parameter 3010 00010 [2]) decoded type **M** auxiliary functions are declared (MACODA parameter 301000030); the operand area is to be located at marker M200.0 for channel 0 (MACODA parameter 2060 00009[1]). The auxiliary functions of the other channels follow one after another (MACODA parameter 2060 00009[2] –> M240, MACODA parameter 2060 00009[3] –>M280).

Owing to these parameters, the marker field has the following structure:

Channel	M function	Marker field
0	M31 – M0	M203.7 – M200.0
	M128 – M96	M215.7 – M212.0
1	M31 – M0	M243.7 – M240.0
	M128 – M96	M255.7 – M252.0
2	M31 – M0	M283.7 – M280.0
	M128 – M96	M295.7 – M292.0

In this example, a 16-byte marker field is defined for every channel. In these marker fields, *e*very **bit-coded** auxiliary function occupies **one** marker.

Bit-coded auxiliary functions **may or may not require acknowledgement**.

MACODA parameter 3010 00020 defines which auxiliary function requires acknowledgement.

Acknowledgement is made by the PLC user program resetting the respective bit-coded auxiliary function.

As long as an auxiliary function which requires acknowledgement is not reset by the PLC user program, the following NC block will not be executed.

Auxiliary functions which do not require acknowledgement must likewise be reset by the PLC user program in order to recognize the changed signal level from logic 0 > to logic 1 when the auxiliary function is output again.

Bit-coded auxiliary functions are first and foremost channel-related, i.e. the assigned bit is set on the channel concerned. With M10, for example, the 11th bit of the channel on which the auxiliary function was output is set.

This way, even large auxiliary function indexes can be assigned to unused auxiliary functions in the lower area. Thus, M999 can address bit 109, e.g.

Depending on the application, operation of the PLC user program can be simplified by assigning a **global** effect to some bit-coded auxiliary functions. For this purpose, a specific auxiliary function of a random channel is assigned to the global auxiliary function in the marker field.

8.2 BCD-coded channel-independent auxiliary functions

Apart from the bit-coded format (type M, e.g.), auxiliary functions can be output also in BCD code with 32 or 64 bits (type S, e.g.).

Channel-independent BCD-coded auxiliary functions are not assigned to any specific channel and therefore have the same effect on every channel.

For every BCD-coded type of auxiliary function, an operand area is defined. Every operand area has either 8 bytes (32-bit BCD value) or 12 bytes (64-bit BCD value).

The address locations of these operand areas are defined in MACODA parameters 2060 00002 (operand type) and 2060 00003 (operand address).

Example:

The ”S” auxiliary function has been declared to be a 32bitcoded auxiliary function.

The start address of the marker field of the ”S” auxiliary function is to begin at the M20 marker byte.

Marker M23 reserved	Marker M22 reserved	Marker M21 reserved	Marker M20 control code
Marker M27 MSB	32-bit BCD value		Marker M24 LSB

The control code in marker byte M20 has the following meaning:

Marker M20
control code

Bit 0	Change signal
Bit 1	Sign
Bit 2	reserved
Bit 3	reserved
Bit 4	reserved
Bit 5	reserved
Bit 6	reserved
Bit 7	reserved

The example above shows that when a BCD-coded auxiliary function is output to the byte M24 through M27 marker field, also the M20.0 change signal is set. The PLC user program must acknowledge the output of the auxiliary function by resetting the change signal on marker M20.0.

If the auxiliary function has been programmed with a negative sign in the part program, the M20.1 marker will also be set.

All BCD-coded auxiliary functions have the same structure as shown in the example above. For 64-bit auxiliary functions, the total length of the corresponding marker fields is 12 bytes instead of 8 bytes.

Notes:

9 PROFIBUS-DP (DP master) interface

The PNC-P has a PROFIBUSDP interface that can be used as a **DP master interface** together with the iPCL.
This interface is designed for decentralized connection of peripheral PROFIBUS-DP modules.

9.1 Commissioning of PROFIBUS-DP interface

Installation steps for PNC-P with KNS:

1. Use the Rexroth configuration tool **WinDP** to generate a master parameter set (see the ”WinDP” documentation).
2. Use the ”load” function of the WinDP tool to store a master parameter file (e.g. bmlist.mxa) on the PC where you are running WinDP.

☞ **Save the name of this master parameter file in format 8.3 (evaluation by the NC is case-sensitive).**

3. Furthermore, the MACODA parameter must be adjusted to fit group 4080. Among other things, the file name of the master parameter set is entered here. For a description of these parameters, see the ”Configuration Parameters and MACODA Parameter Description” manual.
4. Subsequently, the PNC-P has to be started again using RUN.
   Possible error messages and information on possible error sources are displayed on the operating panel of the PNC-P.
   If there is a data exchange failure with connected DP slaves, an error message or a warning will be displayed on the operating panel.
   The diagnostics tools of WinDP (see ”WinDP” documentation) can be used for analysis of the error cause.
   The decentrally connected peripheral modules can only be accessed properly if no error message or warning indicating a data exchange failure is displayed any more on the operating panel.
5. You must adapt the PLC user program of the iPCL before you can commission the DP master. This adaptation is described in Section 9.2.

Installation steps for PNC-P with iPCL:

1. Use the Rexroth configuration tool **WinDP** to generate a master parameter set (see the ”WinDP” documentation).
2. Use the ”load” function of the WinDP tool to automatically store the master parameter set as a file in the PNC-P under the file name entered in MACODA (group 4080).

☞ **For a description of these parameters, see the ”Configuration Parameters and MACODA Parameter Description” manual.**

3. Subsequently, the PNC-P has to be started again using RUN.
   Possible error messages and information on possible error sources are displayed on the operating panel of the PNC-P.
   If there is a data exchange failure with connected DP slaves, an error message or a warning will be displayed on the operating panel.
   The diagnostics tools of WinDP (see ”WinDP” documentation) can be used for analysis of the error cause.
   The decentrally connected peripheral modules can only be accessed properly if no error message or warning indicating a data exchange failure is displayed any more on the operating panel.
4. You must adapt the PLC user program of the iPCL before you can commission the DP master. This adaptation is described in Section 9.2.

9.2 Determining the DP master status

Behavior of the iPCL:

In case of the iPCL, the DP master status is stored in the S 114 data word. The slave diagnostics are returned in system range S 240 – S 255. This can be used to determine all the faulty slaves (see the documentation ”iPCL System Description and Programming Manual”). These data are used to determine the state of the bus and allow for the PLC program to respond to possible errors.

For the evaluation in the PLC user program, see the following table, ”DP Master Status, Bits and Errors”:

Bit	Meaning
0	Bus master error
1	Classified slave diagnostics
2	System diagnostics
5	DP bus set to CLEAR
6	PG retains DP bus in STOP status (cannot happen)
7	DP bus master is active
8	Slave(s) not available
9	Slave(s) signal(s) configuration error
10	Slave(s) signal(s) static diagnostics
11	Slave(s) signal(s) extended diagnostics
12	Slave(s) not ready
13	Slave(s) invalid

Errors that are signaled by the DP master in the DP master status do not necessarily lead to an iPCL/ICL 700 STOP. Any error handling, if required, must be done in the PLC user program.

PROFIBUS-DP (DP master) interface

Notes:

10 Machine error and status display (MSD)

This function facilitates the display of a total of 2048 machine errors, warnings, messages or machine statuses on the NC control panel, thereby supporting rapid error search when malfunctions occur in the machine operation.

The respective messages are defined in plain text in one file (MSD file) and activated when the respective marker is set by the PLC sequential program.

10.1 MSD file and displays

Function and structure of the MZA file:

The same overall conditions apply to creating and handling the MSD file as to creating an NC part program.

The MZA file receives the name **”mzatexte.???”** where ”???” stands for the respective language to be used, e.g. 044 for English.

Every line in the MSD file must end with a ”**)**”.

A total of **2048 messages** can be defined. Every message is assigned a serial number (1...2048). There may be gaps in the list of serial numbers.

The various types of messages are distinguished as follows:

Machine errors	designation **I**
Machine warnings	designation **W**
Machine messages	designation **M**
Machine status	designation **Z**
Operating status	designation **B**

In addition to this classification by machine error, machine status and operating status , these messages can be classed by **groups** in the MSD file, to increase clarity in the display unit and facilitate error search when malfunctions occur. A group is defined by the **designation G**.

Designation I is used to designate an area of ”**fast signals**”, with representing a serial number from 1 to a max. of 2017 (in steps of 32). A total of 32 messages can be defined as fast signals. Programming ”**(I33)**” means that the fast signals are stored under serial numbers 33 – 64. These fast signals are checked for changes by the APS after each PLC cycle. This ensures that short-time changes present for at least one PLC cycle are transferred to the host and, if necessary, to the display unit. Up to 48 messages with their group designations can be shown on each screen of the display.

Display:

Machine errors, warnings and messages are displayed in the INFO dialog and in the status line of the OI. The status line displays the highest priority message. In addition, machine errors are displayed in the DIAGNOSTICS operating area in the MSD dialog function.
Operating status messages differ from machine status messages insofar as they are not only displayed in the MSD dialog but are, in addition, transferred like machine errors and warnings via the communication interface to the host computer.

Formatting and length of MSD texts:

- The MSD dialog displays a maximum of 38 characters of an MSD text. The character sequence ”\n” in the text has the effect that the text following this character sequence is not displayed.
- A maximum of 2 lines with 18 characters each can be displayed in the NC status line. The text following ”\n” is ignored.
- A maximum of 41 characters are displayed in the info list of the INFO dialog. The text following ”\n” is ignored.
- In the field ”selected info” of the INFO dialog, the MSD text can be displayed in several lines. 34 characters per line are allowed. An ”\n” in the MSD text results in a line break.

  Tip:
  2-part structure of the MSD text (separated by ”\n”:
  1. The text of the machine error, warning or message.
  2. Additional information (e.g. elimination text)
     The additional information would be displayed in the INFO dialog when the corresponding MSD message occurs.

  Example:
  (I20, emergency stop actuated from the magazine table\nsee error description\ngroup1)

Example of an MSD file structure:

```
(G1, protective devices)
(I1, EMERGENCY stop actuated)
(I2, touch guards actuated)
(I3, light barrier actuated)
(I4, door opened)
(G2, lubrication)
(I20, no oil pressure)
(I21, oil level too low)
(I22, oil cooler inactive)
(W29, oil contaminated)
(M30, oil slightly contaminated)
(Z31, oil has reached operating temperature range)
(Z32, oil level sufficient)
(G3, collision monitor)
(I97)
(I100, part 1 and 2 temporarily collided)
(Z110, inject adhesive)
```

Machine error and status display (MSD)

	Identifier	MSD dialog	Status line	INFO dialog	Host
Error	E	yes	yes	yes	yes
Warning	W	yes	no	yes	yes
Message	M	yes	no	yes	no
Status	Z	yes	no	no	no
Operating status	B	yes	no	no	yes
Group	G	yes	no	no	no

10.2 Activating messages

Machine errors, warnings, messages, statuses or operating statuses can be activated by setting defined bit signals or deleted by resetting the respective markers. The range of these bit signals is defined by MACODA parameters 2060 00003 [4] (operand type) and 2060 00002 [4] (operand index). **Marker area M4000.0 ... M4255.7** is currently reserved for a total of **2048** messages.

These markers are set or reset exclusively by the PLC sequential program.

Marker **M4000.0** is assigned to the message with serial number 1, marker **M4000.1** to the message with serial number 2.

Machine error and status display (MSD)

Notes:

11 Program modules for iPCL

11.1 General structure of program modules

The integrated program modules can be called up cyclically and with parameters set by the PLC sequential program.
Apart from setting the parameters required for a function (e.g. a program number for the external program selection), the user must also define the data area where the data is to be stored in each case.

Within this data area, there are a status word and an error code defined if an error occurs.

Thus, you must set two parameters for every program module call.
The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.
If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

☞ **The program examples shown in the context of APS program module descriptions show the PROFI software syntax. For PLC user programs written with WinSPS, this syntax must be adapted correspondingly.**

As a rule, these data areas, the addresses of which are defined in parameter P0, have the following structure:

	MSB bit 31		LSB 0
DW0	Error code (2 bytes)	Status (1 byte)	Job (1 byte)
DW1	Length of the total data area reserved (4 bytes)		
DW2	Channel number	Offset from the beginning of the data area to the 1st data feedback address	
DW3	Transfer parameter for the activated function, e.g. program number for ext. program selection. The length of this data area is dependent on the function concerned. Function. The entry is completed with the NUL character if input is made in the form of a character string.		
DWx	Data feedback, e.g. actual axis values, CPL variables, etc.		

DW: Designation of the operand (input, output, marker, entry in data module) as a double word with a length of 32 bits.

Double words DW0...DW2 appear in every program module data area but they are defined only depending on the respective function. If, e.g., no channel number is required, constant K-1 must be entered instead.

The same applies to entering the offset if there is no data feedback involved in a function.

Double word DW0

Double word DW0 contains the error code, status and job.

The **job** has a length of 1 byte and may contain the following functions:

Program modules for iPCL

MSB LSB

Bit 31 16	15 8	7 0
Error code	Status	Job

Bit 0 Start function (=1)
Bit 1 to be defined depending on the respective function
Bit 2 ”
Bit 3 ”
Bit 4 ”
Bit 5 ”
Bit 6 ”
Bit 7 ”

The **status** has a length of 1 byte and may contain the following functions:

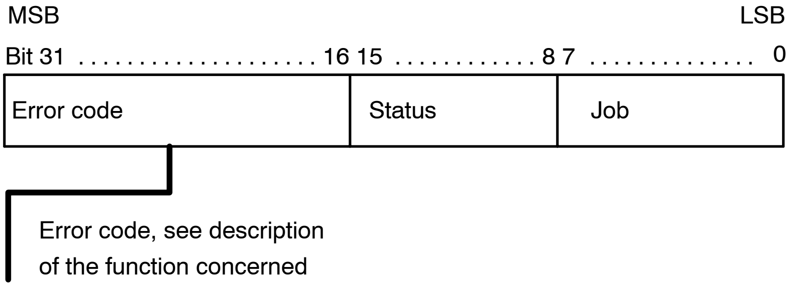
MSB LSB

Bit 31 16	15 8	7 0
Error code	Status	Job

Bit 0 to be defined depending on the respective function
Bit 1 ”
Bit 2 ”
Bit 3 ”
Bit 4 Function active (=1)
Bit 5 to be defined depending on the respective function
Bit 6 ”
Bit 7 Error (see also error code)

☞ **When the PLC user program is saved in the FEPROM, data ranges (e.g. data modules) that may contain data areas for client module calls are also saved. If the function active bit is set when the PLC user program is being saved, the function active status is saved, too, with the effect that the respective function is not carried out when the user program is loaded the next time.**
The same problem arises with data areas located in a retentive marker area or a data module because, when the NC is restarted, these areas are not automatically deleted by the NC.
To avoid this, data areas used for communication should be preassigned valid values by the PLC in a start-up module (OM5 or OM7) or in the 1st PLC cycle.

The **error code** has a length of 2 bytes:


MSB
LSB
Bit 31 .... 16 15 .... 8 7 0
Error code
Status
Job
Error code, see description of the function concerned

The type of error or status is represented in the error code in the form of a **binary-coded, signed number**.

There are two categories of errors and states:

1. Errors and statuses caused by incorrect parameter settings or an internal NC error.
   These messages are listed under **general errors and statuses** in the following.
2. Errors and statuses signaled as an acknowledgement upon the execution of a function.
   These messages are listed under **function-specific errors and statuses** in the following.

An error code is output whenever an error occurs in the execution of a function which prevents the proper execution of the respective function.

Errors may be caused by, among other things, incorrect parameterization, e.g. an invalid channel number. Also, system errors may occur, which will also be displayed with the respective error code.

Because some of the error messages are function-specific, they are explained in the description of the respective function.

Error codes are defined by Rexroth. They are updated as required in the context of new releases.

Double word DW1

Double word DW1 is used to define the size of the data area required to transfer or receive the data of the respective function.

The length of the data area is stated in bytes to be entered as an integer.

Example:
The length of the data area is to be 40 bytes. In double word DW1, constant K 28_H must be entered.

Double word DW2

In double word DW2, an offset must be entered which defines the length from the beginning of the data area down to the address where the data feedback is to be stored. In addition, a channel number must be defined here if the function to be executed requires a channel number.

Entries are made in binary code.

MSB LSB

Bit 31 .... 16 15 .... 8 7 0

Channel number	Offset from the beginning of the data area to the 1st data feedback address Data

The offset length is stated in bytes to be entered as an integer. If there is no data feedback involved in the respective function, constant K-1 must be entered in byte 0 and in byte 1 of double word DW2.

Example:

The data area begins with the start address at marker byte M200.0. The data feedback is to be stored from marker byte M232.0 on. Thus, the offset is 32 bytes.

Constant K 20_H must be entered in byte 0 and byte 1 of double word DW2.

The channel number must be entered as an integer. If no channel number is required for the function to be executed, constant K-1 must be entered in byte 2 and byte 3 of double word DW2.

Example:

The actual values of the axes on channel 1 are to be scanned.

Constant K B_H must be entered in byte 2 and byte 3 of double word DW2.

Double word DW3

From double word 3 on, the address area is used to define the parameters that must be transferred additionally for the respective function to be executed, e.g. for the **external program selection** module.

In the case of the external program selection module, the parameter required for the execution of this function is the part program number, and for CPL variable declarations, their permanent variable numbers are required, etc.

If no parameters need to be transferred for the execution of a function, no definition of double word DW3 is required. Instead, data feedback may be stored already from here on.

Double word DWx

This address is dynamic and is determined indirectly by the offset defined in double word DW2.

Data feedback, e.g. actual axis values or CPL variables, is stored in this area.

11.2 Read actual axis values program module (-B02AXPOS)

Function

The program module **read actual axis values (-B02AXPOS)** supplies the **actual axis values of all** axes as at the time of the request.

Accordingly, enough storage space must be reserved for this purpose in the data module or marker field.

The **actual axis value** feedback is scaled to 0.1m and represented as an integer with 4 bytes.

As regards cyclic calls of this module, please note that the number of calls should be limited to what is necessary to reduce the strain on system capacity. This can be achieved by, for example, timer programming, time-controlled modules or by calling the module only every nth cycle.

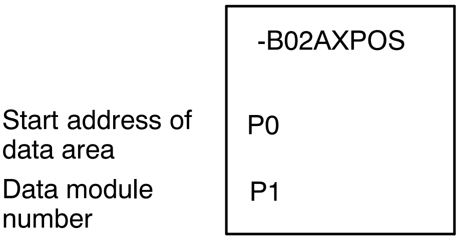
11.2.1 Module call

The module may be called once or repeatedly in a PLC cycle; calls may be unconditional or conditional.

If the module is called again before the data from the previous call is received, please note that a different data area must be activated.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, parameter P1 contains the data module number.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

Program modules for iPCL

In the case of program module **-B02AXPOS**, the data area, the address of which is defined in parameter P0, has the following structure:

MSB LSB

bit 31 0

DW0	Error code (2 bytes)	Status (1 byte)	Job (1 byte)
DW1	Length of the total data area reserved (4 bytes)		
DW2	Channel number	Offset from the beginning of the data area to the 1st data feedback address	
DW3	1^{st} actual axis data feedbackt		
. . . .	:		
DWn	last actual axis data feedback		

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle.

Start function may be reset by the PLC sequential program with the **function active** message.

If not reset by the PLC sequential program, **start function** is automatically reset when the function is finished.

Status bit 4, **function active**, is set. It remains set for one PLC cycle at least, or until the function has been executed.

Execution of this function is complete as soon as the data feedback is received. Then, status bit 4, **function active**, is reset.

This module provides actual data feedback of every axis, regardless of the channel number transferred.

To ensure compatibility also in the future, –1 should be entered for the channel number.

11.2.2 Configuration example

3 axes are defined in channel 1. The actual axis values are to be queried with the positive edge of PLC input I0.2.

The data area is to begin at marker 200.

Given these requirements, a **-B02AXPOS** program module call may be designed as follows:

```
AN   B    I0.2
A    B    M302.0
R    B    M302.0         ; resetting the edgetrigger marker
A    B    M302.7         ; evaluate answer ?
JPC       -checkRsp
A    B    I0.2           ; scan input 0.2
AN   B    M302.0         ; formation of positive edge
JPI       -end
S    B    M302.0         ; setting the edgetrigger marker
S    B    M302.7         ; set the evaluate answer marker
S    B    M200.0         ; setting start function
L    DW   K24,A          ; definition of parameters to be transferred
T    DW   A,M204         ; total length of data range = 24 bytes
L    W    K12,A          ; 12 bytes offset from beginning of data area to
T    W    A,M208         ; actual axis value of 1st axis,
L    W    K1,A           ; request for
T    W    A,M210         ; actual axis value of channel 1.
CM        -B02AXPOS,2    ;call module -B02AXPOS
P0        M200           ; the data area begins at marker 200
P1        K-1            ; not in a data module
JP        -end
          -checkRsp
L    DW   M200,A         ; job, status and error codes in reg.
A    B    A.12           ; wait while function is still active
JPC       -end
A    B    M302.7
R    B    M302.7         ; resetting of evaluate answer
A    B    A.15           ; error flag set ?
JPI       -end
                         ; error handling
EM                       ; processing of received data
          -end
EM
```

The actual axis value of the

1st axis is stored on markers	212...215,
1st axis is stored on markers	216...219,
3rd axis is stored on markers	220...223

11.2.3 Status messages

When the **read actual axis values** function has been called and an error or a status occurs which prevents the proper execution of this function, **bit 7** is set in **status.**

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF_H: Parameter errors caused by incorrect parameter input in the data area.
FFFE_H: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.
FFFD_H: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.
FFFC_H: Module **B01APSMN** has not been called.
FFFB_H: Incorrect offset: The offset is too large (> reserved area) or too small (overwriting of job data occurs).
FFF_AH: Currently, no NCS material is available; the function request could not be sent.
8000_H: NC-internal error, cannot be influenced by user.
7FFF_H: PxRos error, cannot be influenced by user.
7FFF_H: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

$\mathbf{1}_\text{H}$: Invalid channel number.

11.3 External program selection/deselection program module (–B03PANAB)

Function

The External program selection/deselection module facilitates the selection and deselection of an NC part program via the PLC sequential program.

The **program change without control reset** function allows selecting another program without a modal function reset. For example, another program can be selected while the spindle is still running if ”M5” in the init string is deleted. This applies on the condition that the program selected previously has been fully executed or has not been started yet.

☞ **The function ”program change without control reset” is applicable from NC software version V2.3.4 up.**

11.3.1 Module call

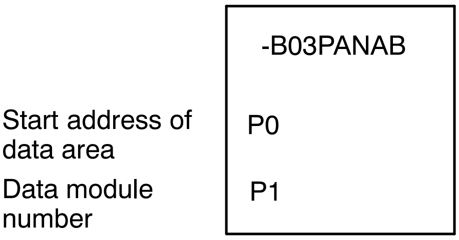
This module can be called unconditionally or conditionally.

If several channels are activated, the module can also be called several times per PLC cycle.

If the module is called repeatedly, please note that a different data area must be parameterized for each call.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

Program modules for iPCL

In the case of program module **-B03PANAB,** the data area, the address of which is defined in parameter P0, has the following structure:

MSB LSB

bit 31 0

<table>
<tr><td>DW0</td><td>Error code (2 bytes)</td><td>Status (1 byte)</td><td>Job (1 byte)</td></tr>
<tr><td>DW1</td><td colspan="3">Length of the total data area reserved (4 bytes)</td></tr>
<tr><td>DW2</td><td>Channel number</td><td colspan="2">Offset from the beginning of the data area to the 1st data feedback address</td></tr>
<tr><td>DW3</td><td colspan="3">Program number in BCD or binary code</td></tr>
<tr><td>DW4
.
.</td><td colspan="3">Program name, max. 30 ASCII characters, with a trailing NUL (00H) ASCII character for termination</td></tr>
<tr><td>.
.
.
.</td><td colspan="3">Start block, max. 9 ASCII characters, with a trailing NUL (00H) ASCII character for termination</td></tr>
<tr><td>.
DWx</td><td colspan="3">End block, max. 9 ASCII characters, with a trailing NUL (00H) ASCII character for termination</td></tr>
</table>

The job byte is structured as follows:

Bit	**Function**
0	Start function
1	Reserved
2	Program number is BCDcoded
3	Program selection with start block
4	Program selection with end block
5-7	Function number:
0:	Reserved
1:	Reserved
2:	Program deselection
3:	Reserved
4:	Program selection without automatic cancellation and modal function resetting
5:	Program selection with automatic cancellation and modal function resetting
6:	Program change without modal function resetting
7:	Reserved

An NC part program can be selected only if either automatic **or single block** operating mode is active.

The function is started when bit 0 is set to logic 1 in the job byte.

Bit 0 may be reset via the PLC sequential program with the **function active** message (bit 4 in status). Otherwise, the ”start function” signal is reset automatically when the function has been carried out.

Bit 4 in the status byte, **function active**, remains set until the function has been carried out, at least, however, until the subsequent call of the **-B01APSMN** program module.

As soon as a program has been selected/deselected, this function is completed and bit 4, **function active**, in the status byte is reset.

Programs can be selected by entering the program number in BCD or binary code or, optionally, by entering the program name with a max. of 30 ASCII characters.

If the program is to be selected by inputting the program number in BCD code, bit 2 must be set in the job and the NUL ASCII character must be set in byte 0 in DW3.

If a start block and an end block are defined, bit 3 and bit 4 must be set in the job.

For deselecting a program, only the program number or the program name need to be entered.

The option as to whether a program is to be selected or an active program is to be deselected is made by setting bit 6 or 7 in the job.

The size of the data area required for the entire function is defined in double word DW1.

Because the **-B03PANAB** module does not return any data, no offset needs to be defined in double word DW2.

Specification of the channel number (as an integer), likewise defined in double word DW2, is, however, absolutely necessary.

The program number is input in BCD or binary code in double word DW3 (provided that a part program with a sequence of numbers and a preceding P, such as Π1234, has been stored in the part program memory).

The **-B03PANAB** module converts the program number into ASCII characters, as the part programs in PNC-R are stored not numerically but according to character.

If a program to be selected is specified by its name in the form of an ASCII character string, please note that the program name is limited to 29 characters, with a trailing NUL ASCII character for termination. If the search path for selecting a program is to be different from the NC-internal search path defined in the MACODA parameters, the respective directory containing the program can be transferred together with the program name. However, the total length of 30 characters must not be exceeded. If you want, e.g., the **test.cpl** program always to be selected from the user FEPROM, the program name to be transferred must read.

If the part program is to be started at a certain block, the start block can be entered at the end of the program name, with a trailing NUL ASCII character for termination. The start block can be an NC block according to DIN 66025 (always begins with ”N”) or a CPL block (no ”N” in front of the block number).

The start block must thus be specified as a character string. The start block length is limited to 9 ASCII characters, with a trailing NUL ASCII character for termination.

If execution of a part program is to be stopped at some point prior to its completion (M30), an end block may be defined. The same rules as for defining start blocks apply to defining end blocks.

If a new program is to be selected while another one is still active, all you have to do is select the new program because function no. 5 or no. 6 is already specified in the job byte. The previous program is deselected automatically provided the NC is in **ready** or **stopped** status.

11.3.2 Configuration example

The NC part program with program name ”P1327” is to be activated in channel 1. Since the program name consists of the letter P and a series of numbers, the program can be selected by presetting a program number in BCD code.

The data area is to begin at byte 28 in data module DM1.

Module **-B03PANAB** could be called as follows:

```
AN   B    I0.3            ;
A    B    M303.0
R    B    M303.0          ; resetting the edge marker
CM        -DM01PANA       ; call data module DM1
A    B    M303.7          ; evaluate answer ?
JPC       -checkRsp
A    B    I0.3            ; query input 0.3
AN   B    M303.0          ; formation of positive edge
EMI
S    B    M303.0          ; setting the edgetrigger marker
S    B    M303.7          ;setting the evaluate answer marker
S    B    D28.2           ; program number in BCD code
S    B    D28.7           ; definition of program selection
S    B    D28.0           ;setting start function
L    DW   K60,A           ; definition of parameters to be transferred
T    W    A,D32           ; total length of data range = 60 bytes
L    W    K0,A            ; no offset required as no data
T    W    A,D36           ; are returned,
L    W    K1,A            ; program should be selected/deselected in
                            channel 1
T    W    A,D38           ; erfolgen,
L    DW   K1327H,A        ; enter program number to be input in
                            BCD code
T    DW   A,D40
L    BY   K0H,A           ; complete program number input
T    BY   A,D44           ;with trailing NUL ASCII character
CM        -B03PANAB,2     ;-B03PANAB module call
P0        K28             ; the data area begins at data byte 28
P1        K1              ; in data module DM1
EM
          -checkRsp
L    DW   D28,A           ; job, status and error codes in reg.
A    B    A.12            ; wait while function is still active
BEB
A    B    M303.7
R    B    M303.7          ;resetting of evaluate answer
A    B    A.15            ; error flag set
EMI                       ; error handling
EM
```

11.3.3 Status messages

If an error or status occurs after calling the ext. program selection/deselection function which prevents proper execution of this function, **bit** 7 is set in **status**.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_{H}$: Parameter errors caused by incorrect parameter input in the data area.
FFFE$_{H}$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.
FFFD$_{H}$: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.
FFFC$_{H}$: Module **B01APSMN** has not been called.
FFFB$_{H}$: Incorrect offset: The offset is too large (> reserved area) or too small (overwriting of job data occurs).
FFFA$_{H}$: Currently, no NCS material is available; the function request could not be sent.
8000$_{H}$: NC-internal error, cannot be influenced by user.
7FFF$_{H}$: PxRos error, cannot be influenced by user.
7FFF$_{H}$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

1: Invalid channel number

Errors or states occurring upon program selection:

2$_{H}$: Channel is busy.
3$_{H}$: Control reset is currently being executed on the channel.
4H: Program name is too long.
5H: Axes have not been referenced.
6H: Program cannot be opened.
7H: Error during program selection.
8H: Either the wrong operation mode or none at all has been selected.

Functionspecific errors and statuses occurring upon program deselection:

1$_{H}$: Invalid channel number.
3$_{H}$: Control reset is currently being executed on the channel.
0A$_{H}$: No program has been selected.
0B$_{H}$: Program is running and feed hold is not present.
19H: Syntax error in the selected program.

11.4 NC block input program module (-B04SATZV)

☞ **This program module is not available for KNS.**

Function

The **NC block input** module facilitates the input of an NC block via the PLC sequential program. This enables the PLC program, e.g., to transfer set-point values of auxiliary axes or auxiliary functions to the NC. NC block input can be made even while the program is running provided no axis addresses are input that would have an impact on the running machining process.

Program module **NC block input** allows any NC block to be preset using the PLC sequential program in modes **Manual data input** or **Automatic**. NC block input can be made even while the program is running provided no axis addresses are input that would have an impact on the running machining process. In this way, for example, command values for auxiliary axes or auxiliary functions can be transferred to the NC. Specifiying an NC block that affects the current processing, such as specifying the feed or programming a synchronous axis, leads to a syntax error.

A block can be preset in the same channel only after the previous block has been fully processed.

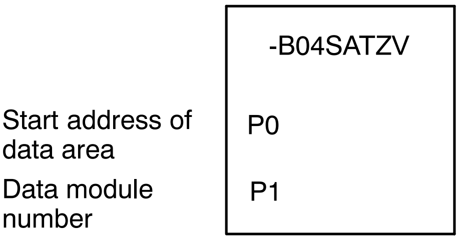
11.4.1 Module call

This module can be called unconditionally or conditionally.

If the module is called repeatedly, please note that a different data area must be parameterized for each call.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1. In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

In the case of program module **-B04SATZV,** the data area, the address of which is defined in parameter P0, has the following structure:

	MSB bit 31		LSB 0
DW0	Error code (2 bytes)	Status (1 byte)	Job (1 byte)
DW1	Length of the total data area reserved (4 bytes)		
DW2	Channel number	Offset from the beginning of the data area to the 1st data feedback address	
DW4 ... DWx	Max. length of NC block: 128 ASCII characters, with a trailing NUL ASCII character (00H) for termination		

The following functions can be activated in the job:

Bit	**Function**
0	Start function
1	Reserved

The function is started when bit 0 is set to logic 1 in the job byte.
Bit 4, **function active**, remains set in the status until the function has been carried out, at least, however, until the subsequent call of the **-B01APSMN** program module.

The start function is reset:

- automatically when the function is completed, or
- via the PLC sequential program, upon setting ”function active” (bit 4 in the status).

As soon as NC block input has been executed, this function is completed and bit 4, **function active**, in the status byte is reset. Execution of NC block input is complete when the respective block has become active.
If an asynchronous movement is input whose end point has not been reached yet, this ensures that the axis travel command is already set at the time the function is completed and ”InPos” can be reset.
To check whether the movement is complete, the ”Axis in position” (iAx_InPos) axis signals can be evaluated.

Because the **-B04SATZV** module does not return any data, no offset needs to be defined in double word DW2.
Specification of the channel number (as an integer), likewise defined in the double word DW2, is, however, absolutely necessary.

The NC block is input from double word DW3 on.
Please note that the NC block must end in a trailing NUL ASCII character (00H).

The NC executes the NC block immediately after receipt, without an additional NC start signal.

11.4.2 Configuration example

NC block **B123** is to be transferred in channel 1. The data area is to begin at byte 4 in data module DM4.

Module **-B04SATZV** could be called as follows:

```
AN   B    I0.4           ;
A    B    M304.0
R    B    M304.0         ; resetting the edge marker
CM   -DM4SatzV           ; call data module DM4
A    B    M304.7         ; evaluate answer ?
JPC       -checkRsp
A    B    I0.4           ; Abfrage Eingang 0.4,
AN   B    M304.0         ; formation of positive edge
JPI       -end
S    B    M304.0         ; setting the edgetrigger marker
S    B    M304.7         ; setting the evaluate answer marker
S    B    D4.0           ; setting start function;
L    DW   K128,A         ; total length of data range = 128 bytes
T    DW   A,D8
L    W    K0,A           ; no offset required as no data
T    W    A,D12          ; are returned,
L    W    K1,A           ; channel number 1
T    W    A,D14
L    DW   K'321B',A      ; note order of characters ”B123”!
T    DW   A,D16
L    BY   K0,A           ; end of string
T    BY   A,D20
CM        -B04SATZV,2    ; call module -B04SATZV
P0        K4             ; the data area begins at data byte 4
P1        K4             ; in data module DM4
EM
          -checkRsp
L    DW   D4,A           ; job, status and error codes in reg.
A    B    A.12           ; wait while function is still active
BEB
A    B    M304.7
R    B    M304.7         resetting of evaluate answer
A    B    A.15           ; error flag set
EMI                      ;   error handling;
EM
```

11.4.3 Status messages

If an error or status occurs after calling the NC block input function which prevents proper execution of this function, **bit 7** is set in **status**.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_H$: Parameter errors caused by incorrect parameter input in the data area.

FFFE$_H$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.

FFFD$_H$: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.

FFFC$_H$: Module **B01APSMN** has not been called.

FFFA$_H$: Currently, no NCS material is available; the function request could not be sent.

8000$_H$: NC-internal error, cannot be influenced by user.

7FFF$_H$: PxRos error, cannot be influenced by user.

7FFF$_H$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

1$_H$: Invalid channel number.

2$_H$: Channel is busy.

3$_H$: The channel is just being reset.

4$_H$: NC block is too long.

5$_H$: Axes have not been referenced.

8$_H$: Either the wrong operation mode or none at all has been selected.

19$_H$: In the case of an NC block input with automatic start, a runtime error, e.g. a syntax error, was detected.

11.5 Temperature compensation program module (-B05TKOMP)

Function

The **temperature compensation** program module allows the user to set a compensation value via the PLC sequential program to offset temperature impacts. The PLC program transfers the temperature compensation value for one axis at a time (as of V5.1.8 also for σεϖεραλ αξεσ at a time) with temperature compensation values becoming active immediately upon transfer. Compensation values are transferred on a 0.0001 mm scale and are applied irrespective of axis positions.

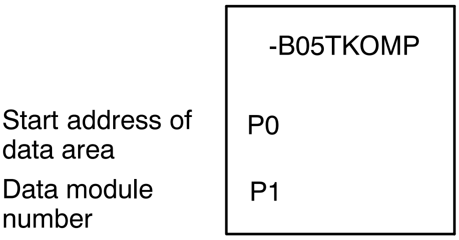
11.5.1 Module call

This module can be called unconditionally or conditionally.

If the module is called repeatedly, please note that a different data area must be parameterized for each call.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

In the case of program module **-B05TKOMP**, the data area, the address of which is defined in parameter P0, has the following structure:

	MSB bit 31		LSB 0
DW0	Error code (2 bytes)	Status (1 byte)	Job (1 byte)
DW1	Length of the total data area reserved (4 bytes)		
DW2	Channel number	Offset from the beginning of the data area to the 1st data feedback address	
DW3	Number of compensation values	Physical axis index	
DW4	1. st compensation value (scaled to 0.1m)		
:	:		
DWx	nth compensation value (scaled to 0.1m)		

The following functions can be activated in the **job**:

Bit	**Function**
0	Start function
1–3	Reserved
4–7	Function number

Permissible parameters:

0: temperature compensation input for one axis only (function available previous functions: up to V5.1.7)

1: temperature compensation input for one or several axes

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle.

Start function may be reset by the PLC sequential program with the **function active** message.
Otherwise, the **start function** signal is reset automatically when the function has been carried out.
Bit 4, **function active**, remains set in the status until the function has been carried out, at least, however, for the duration of the PLC cycle. When the NC has acknowledged the function, the function is completed and bit 4, **function active**, is reset in the status.

Because the **-B05TKOMP** module does not return any data, no **offset** needs to be defined in double word DW2.
Likewise, no **channel number** needs to be specified in double word DW2 because this function is axis-related.

The **physical axis index** and the νυμβερ **of compensation values** must be entered in double word DW3. The NC takes the temperature compensation values transferred to it into account as of the axis with the index ”axis index”. The axis index starts at 0 for the 1st axis.

☞ **If a temperature compensation value is entered for just one axis (function number 0), 0 must be entered for the number of compensation values. This allows for the module to be integrated in an existing PLC program without first having to make any changes in the PLC user program.**

The temperature compensation is immediately accepted by the NC.

11.5.2 Configuration example

To compensate the temperature impact, a compensation value of 100 m is to be input for the 4th axis.
The data area is to begin at byte 12 in data module DM5.

Module **-B05TKOMP** could be called as follows:

```
AN   B    I0.5
A    B    M305.0
R    B    M305.0          ; resetting the edge marker
CM        -DM5TKOMP       ; call data module DM5
A    B    M305.7          ; evaluate answer ?
JPC       -checkRsp
A    B    I0.5            ; query input 0.5
AN   B    M305.0          ; formation of positive edge
JPI       -end
S    B    M305.0          ; setting the edgetrigger marker
S    B    M305.7          ; setting the evaluate answer marker
S    B    D12.0           ; setting start function
L    DW   K20,A           ; definition of parameters to be transferred
T    DW   A,D16           ; Gesamtlänge des Datenber. 20 Byte,
L    W    K0,A            ; no offset required as no data
T    W    A,D20           ; are returned,
L    W    K-1,A           ; no channel No. because function is
                           axis-related
T    W    A,D22
L    DW   K4,A            ; for the 4th axis
T    DW   A,D24
L    W    K1000,A         ; 100 μ (scaling 0.1 μ)
T    W    A,D28
CM        -B05TKOMP,2     ; call module -B05TKOMP
P0        K12             ; the data area begins at data byte 12
P1        K5              ; in data module DM5
JP        -end
          -checkRsp
L    DW   D12,A           ; job, status and error codes in reg.
A    B    A.12            ; wait while function is still active
JPC       -end
A    B    M305.7
```

```
R      B      M305.7         ; resetting of evaluate answer
A      B      A.15           ; error flag set
                             ; error handling
JP            -end           ; processing of received data
-end
EM
```

11.5.3 Status messages

If an error or status occurs after calling the **temperature compensation** function which prevents proper execution of this function, **bit 7** is set in status.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_H$: Parameter errors caused by incorrect parameter input in the data area.

FFFE$_H$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.

FFFD$_H$: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.

FFFC$_H$: Module **B01APSMN** has not been called.

FFFA$_H$: Currently, no NCS material is available; the function request could not be sent.

8000$_H$: NC-internal error, cannot be influenced by user.

7FFF$_H$: PxRos error, cannot be influenced by user.

7FFF$_H$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

10H: Too many temperature compensation values (displayed with the axis number, if applicable)

18$_H$: Invalid axis number

11.6 Logic analyzer program module (-B06LGANA)

Function

The **logic analyzer** program module contains the entire logic for recording the logic analyzer signals within the diagnostics on the NC side.

If this diagnostics function is used, the module must be included in the PLC program.

11.6.1 Module call

☞ **The logic analyzer is called from within the APS modules and, therefore, must not be called from the user program. Otherwise, the NC would stop operations with a system error.**

11.7 Read MACODA parameters program module (-B07MPARA)

Function

The read **MACODA parameters (-B07MPARA)** program module returns the values of a MACODA parameter (block).

The memory space required in the data module or marker field must be defined sufficiently large for all individual parameters to be stored there.

Because this module does not convert value formats, real values are returned as real numbers.

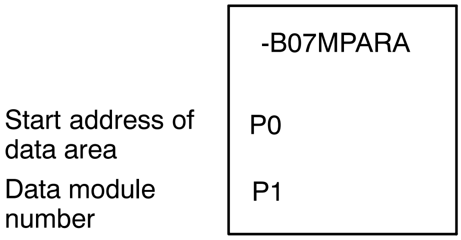
11.7.1 Module call

The module may be called once or repeatedly in a PLC cycle; calls may be unconditional or conditional.

If the module is called again before the data from the previous call is received, please note that a different data area must be activated.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

In the case of program module **-B07MPARA**, the data area, the address of which is defined in parameter P0, has the following structure:

MSB LSB

bit 31 0

DW0	Error code (2 bytes)	Status (1 byte)	Job (1 byte)
DW1	Length of the total data area reserved (4 bytes)		
DW2	Channel number	Offset from the beginning of the data area to the 1st data feedback address	
DW3		No. of MACODA parameter (block)	
DW4		1st individual parameter returned	
. . . . . .		:	
DWx		last individual parameter returned	

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle.

Start function may be reset by the PLC sequential program with the **function active** message.

If not reset by the PLC sequential program, **start function** is automatically reset when the function is finished.

Bit 4, **function active**, remains set in the status until the function has been carried out, at least, however, for the duration of the PLC cycle.

Execution of this function is complete as soon as the data feedback is received. Then, status bit 4, **function active**, is reset.

Because the **-B07MPARA** module does return data, an offset in double word DW2 must be defined, indicating the beginning of the returned data.

No channel number needs to be specified in double word DW2 because this function is not channelrelated.

The number of the MACODA parameter (block) to be read must be entered in double word DW3.

11.7.2 Configuration example

The values of MACODA parameter 2060 00001 are to be scanned upon the formation of the positive edge at PLC input I0.7.

The data area is to begin at marker 700.

Given these requirements, a **-B07MPARA** program module call may be designed as follows:

```
AN   B    I0.7
A    B    M307.0
R    B    M307.0         ; resetting the edge marker
A    B    M307.7         ; evaluate answer ?
JPC       -checkRsp
A    B    I0.7           ; query input 0.7
AN   B    M307.0         ; formation of positive edge
EMI
S    B    M307.0         ; setting the edgetrigger marker
S    B    M307.0         ; setting the evaluate answer marker
S    B    M700.0         ; setting start function
L    DW   K24,A
L    DW   K24,A          ; total length of data range = 24 bytes
L    W    K16,A          ; 16 bytes offset from beginning of data area to
T    W    A,M708         ; 1st individual parameter
L    W    K-1,A          ; no channel number
T    W    A,M710
L    DW   K206000001,A   ; requesting MACODA parameter
T    DW   A,M712         ; with the number 2060 00001
CM        -B07MPARA,2    ; call module -B07MPARA
P0        M700           ; the data area begins at M700
P1        K-1            ; not in a data module
EM
          -checkRsp
L    DW   M700,A         ; job, status and error codes in reg. A
A    B    A.12           ; wait while function is still active
BEB
A    B    M307.7
R    B    M307.7         ; resetting of evaluate answer
A    B    A.15           ; error flag set
JPI       -FktOk
                         ;    error handling
EM
          -FktOk         ;    processing of received data
EM
```

The individual parameters returned are stored as follows:

1: individual parameter on markers	716...719,
2. individual parameter on markers	720...723,
3. individual parameter on markers	724...727

11.7.3 Status messages

If, after calling the **Read MACODA parameters** function, an error or status which prevents proper execution of this function occurs, **bit 7** is set in **Status**.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_H$: Parameter errors caused by incorrect parameter input in the data area.

FFFE$_H$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.

FFFD$_H$: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.

FFFC$_H$: Module B01APSMN has not been called.

FFFA$_H$: Currently, no NCS material is available; the function request could not be sent.

8000$_H$: NC-internal error, cannot be influenced by user.

7FFF$_H$: PxRos error, cannot be influenced by user.

7FFF$_H$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

72: No MACODA parameter (block) defined

11.8 Open/close serial interface program module (-B10SEROP)

Function

The **open/close serial interface** program module facilitates the opening and closing of a serial interface of the NC.
For opening a serial interface, the interface parameters to be used to initialize the selected interface may be entered optionally. If no such parameters are entered, the interface parameters used in the previous opening process or those defined in MACODA are applied.

☞ **The PNC-P does not support accesses to the serial interface, i.e. this module must not be used with the PNC-P.**

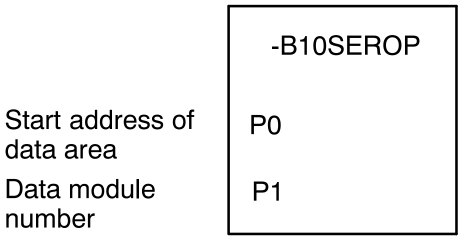
11.8.1 Module call

The module may be called once or repeatedly in a PLC cycle; calls may be unconditional or conditional.

If the module is called again before the data from the previous call is received, please note that a different data area must be activated.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

In the case of program module **-B10SEROP** program module, the data area, the address of which is defined in parameter P0, has the following structure:

MSB LSB

Bit 31 . . . . 16 15 0

<table>
<tr><td>Error code (2 bytes)</td><td>Status (1 byte)</td><td>Job (1 byte)</td></tr>
<tr><td colspan="3">Length of the total data area reserved (4 bytes)</td></tr>
<tr><td>Channel number</td><td colspan="2">Offset</td></tr>
<tr><td>Driver</td><td colspan="2">Interface</td></tr>
<tr><td colspan="3">Baud rate</td></tr>
<tr><td>Log. interface parameters</td><td>Data format</td><td>No. of data bits</td></tr>
<tr><td colspan="3">Abort string (4 bytes)</td></tr>
<tr><td colspan="3">Read-in timeout [s]</td></tr>
<tr><td colspan="3">Read-out timeout [s]</td></tr>
<tr><td colspan="3">Identifier</td></tr>
<tr><td colspan="3">:</td></tr>
<tr><td colspan="3">Data</td></tr>
</table>

The following functions can be activated in the job:

Bit	**Function**
0	Start function
1-3	Reserved
4-7	Function number:
1:	Opening the interface with unchanged parameters
2:	Opening the interface with new interface parameters
3:	Closing the interface

Start function

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle. Start function may be reset by the PLC sequential program with the **function active** message. Otherwise, the **start function** signal is reset automatically when the function has been carried out.

Status

Bit 4, **function active**, remains set in the status until the function has been carried out, at least, however, for the duration of one PLC cycle. Sofern die Schnittstelle geöffnet bzw. geschlossen wurde, ist die Funktion beendet, und im Status wird das Bit 4 **Funktion aktiv** zurückgesetzt.

Size of data area

In order to prevent accidental overwriting of data, the user enters here the maximum number of bytes expected to occur with this function.

Offset

The offset is the space between the beginning of the area (job) and the address where the received data identification is to be stored or, resp., where the identification for closing the interface is stored.

Channel no.

This parameter is irrelevant to this module.

Interface no.

The interface number can be used to select one of the serial interfaces for the function.

The following values are permitted:

1 - Interface X32 on the BOR module
2 - Interface X31 on the BOR module
4 - Interface X35 on the osa master module

Driver number

Only the TTY driver can be selected at present for data exchange via a serial interface (driver number = 0).

☞ **The following parameters are required only if also the interface parameters are to be transferred in the process of opening an interface. These parameters are irrelevant to any other function.**

Baud rate

Baud rate at which the interface is to operate.

The following values are permitted:

110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 Baud

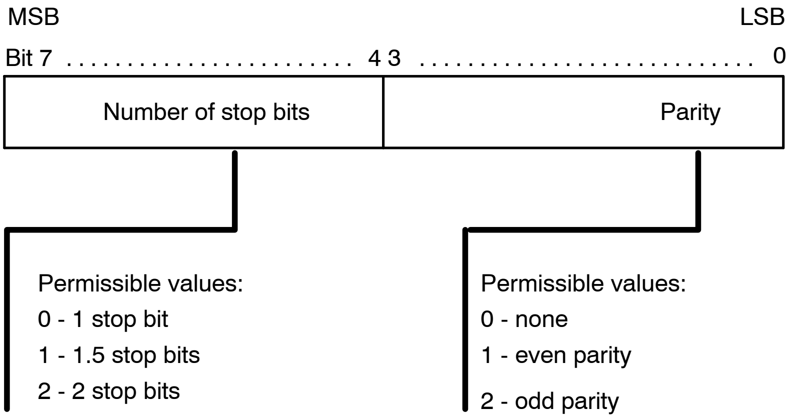
Number of data bits

This parameter defines the number of data bits assigned to every packet received.

The following values are permitted:
5, 6, 7, 8

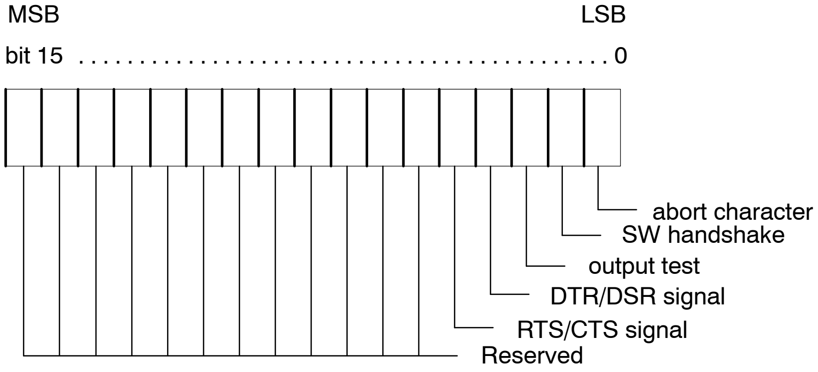
Data format (parity, number of stop bits)

This parameter is used to define the parity and the number of stop bits.



Logical interface parameters

Serial interface performance can be controlled by setting the logic interface parameters. Setting the respective bit will activate the function.



Abort character: When the respective control character (< 20H or > 7fH) is received, the loading process is aborted and the characters read in by then are transferred.

SW handshake (XON/XOFF): The interface is controlled by software handshake.

Output test: For test purposes, all control characters are output as displayable characters (e.g., ”<0x24>” for the output of a <CR> character at the line end)

DTR/DSR signals: For hardware handshakes, the DSR and DTR signal lines are used for data transfer control.

RTS/CTS signals: For hardware handshakes, the RTS and CTS signal lines are used for data transfer control.

Abort character

Up to 4 characters may be defined to control data traffic; when one is received, the reading process is aborted and the characters read in by then are transferred to the user. When they are received, the loading process is aborted and the characters read in by then are transferred to the user together with the abort character being the last character received.

Readin timeout

The check time for data readin must be entered in seconds. If a timeout occurs in the course of a readin process, the characters readin by then are transferred to the user.

Readout timeout

The check time for data readout must be entered in seconds. If a timeout occurs in the course of a readout process, the characters readout by then are transferred to the user.

Identifier

The function returns an identifier in acknowledgement of the successful execution of the interface opening process. This identifier must be entered for all interface functions (readin, readout, closing). The memory area where this identifier is stored results from the start address of the data area transferred and the offset entered.

11.8.2 Configuration example

Example:

Interface X32 on the BOR module is to be opened with the following parameters:

9600 bauds, 8 data bits, even parity and 2 stop bits. Data flow is to be controlled by software handshake (XON/XOFF):

The function is started at the increasing flank of input E10.0.
The data area with the job parameters begins at byte 12 in data module DM10.

Module -B10SEROP.PAH could be called as follows:

```
AN   B    I10.0
A    B    M310.0
R    B    M310.0          ; resetting the edge marker
CM        -DM10serOp      ; call data module DM10
A    B    M310.7          ; evaluate answer ?
JPC       -checkRsp
A    B    I10.0           ; query input 10.0
AN   B    M310.0          ; formation of positive edge
EMI
S    B    M310.0          ; setting the edgetrigger marker
S    B    M310.7          ; setting the evaluate answer marker
L    BY   K2,A            ; open with input of the interface-
                          ; parameters
SLL  BY   A,4             ;function no. is entered in bit 4 in the job
O    BY   K1,A            ; setting start function
T    BY   A,D12           ; save job byte
L    DW   K64,A           ; definition of parameters to be transferred
T    DW   A,D16           ; total length of data range = 64 bytes
L    W    K36,A           ; offset from beginning of data area
                           Identifier
T    W    A,D20           ;
L    W    K-1,A           ; no channel-specific function
T    W    A,D22
L    W    K4,A            ; interface X35 on the osa master module
T    W    A,D24
L    W    K0,A            ; TTY-driver
T    W    A,D26
L    DW   K9600,A         ; 9600 bauds
T    W    A,D28
L    BY   K8,A            ; 8 data bits
T    BY   A,D32
L    BY   K1,A            ; even parity
L    BY   K2,B            ; 2 stop bits
SLL  BY   B,4             ; No. of stop bits in upper half-byte
O    BY   B,A
T    BY   A,D33
L    W    K2,A            ; SW-Handshake (XON/XOFF)
```

```
T    W    A,D34
L    DW   KFFFFFFFFH,A  ; no abort characters
T    DW   A,D36
L    DW   K2,A          ; readin timeout = 2 s
T    DW   A,D40
L    DW   K2,A          ; readout timeout = 2 s
T    DW   A,D44
CM        -B10SEROP,2   ; call module B10SEROP.PAH
P0   K12                ; the data area begins at data byte 12
P1   K10                ; in data module DM10
EM
     -checkRsp
L    DW   D12,A         ; job, status and error codes in reg.A
A    B    A.12          ; wait while function is still active
BEB
A    B    M310.7
R    B    M310.7        ; resetting of evaluate answer
A    B    A.15          ; error flag set
EMI                     ; error handling
EM
```

11.8.3 Status messages

If, after calling the **Open/close serial interface function**, an error or status which prevents proper execution of this function occurs, **bit 7** is set in **Status**.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_H$: Parameter errors caused by incorrect parameter input in the data area.
FFFE$_H$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.
FFFD$_H$: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.
FFFC$_H$: Module **B01APSMN** has not been called.
FFFB$_H$: Incorrect offset: The offset is too large (> reserved area) or too small (overwriting of job data occurs).
FFFA$_H$: Currently, no NCS material is available; the function request could not be sent.
8000$_H$: NC-internal error, cannot be influenced by user.
7FFF$_H$: PxRos error, cannot be influenced by user.
7FFF$_H$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

2: Invalid job parameters
3: Invalid identifier
7: Device driver does not exist
8: Invalid interface number
9: Interface is already open (possibly by the operator via the operator interface or by the DNC)
10: Changing the interface parameters was canceled when the interface was closed.
12: There is no operational equipment available at present to which this function could be applied
35: Other error within the input/output logic

11.9 Data read-in/read-out program module (-B11SERIO)

Function

The **data read-in/read-out** program module allows data to be read in and read out via a serial NC interface.

☞ **The PNC-P does not support accesses to the serial interface, i.e. this module must not be used with the PNC-P.**

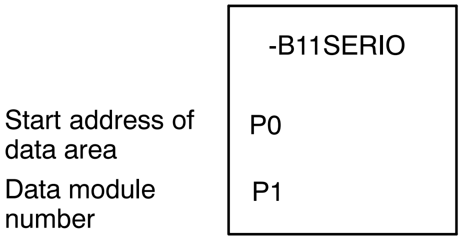
11.9.1 Module call

The module may be called once or repeatedly in a PLC cycle; calls may be unconditional or conditional.

If the module is called again before the data from the previous call is received, please note that a different data area must be activated.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

In the case of program module **-B11SERIO**, the data area, the address of which is defined in parameter P0, has the following structure:

MSB LSB

Bit 31 .... 16 15 0

Error code (2 bytes)	Status (1 byte)	Job (1 byte)
Length of the total data area reserved (4 bytes)		
Channel number	Offset	
Identifier		
Volume of data to be read in / read out (in bytes)		
Data to be read out / volume of data to be read in		
Data read-in		

The following functions can be activated in the job:

Bit	**Function**
0	Start function
1-3	Reserved
4-7	Function number:

1:	Readingin data at the serial interface
2:	Readingout data at the serial interface
3:	Cancellation of data readin
4:	Cancellation of data readout

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle.

Start function may be reset by the PLC sequential program with the **function active** message.

If not reset by the PLC sequential program, **start function** is automatically reset when the function is finished.

Status

Bit 4, **function active**, remains set in the status until the function has been carried out, at least, however, for the duration of the PLC cycle.

When the data has been loaded or output or the respective transaction has been cancelled, the function is completed and bit 4, **function active**, is reset in the status.

Size of data area

In order to prevent accidental overwriting of data, enter here the maximum number of bytes you want to provide for this function.

Offset

The offset is the space between the beginning of the area (job) and the address where the received data is to be stored or, resp., where the data to be read out is stored.

Channel no.

This parameter is irrelevant to this module.

Identifier

The identifier which was returned when the interface was opened.

Volume of data to be read in / read out

Data volume in bytes.

Output data / number of characters read in

When data is read in, the number of data bytes readin is stored here.

When data is to be read out to the interface, data is transferred from this point on.

Received data

When data is read in, the data bytes received are stored here.

11.9.2 Configuration example

A total of 64 characters are to be loaded from the serial interface.

The identifier that was returned when the interface was opened was entered in D48 in data module DM10. The data area containing the job parameters is to begin at byte 0 in data module DM11 and the function is to be started with the rising edge on input I11.0. Module -B11SERIO.PAH could be called as follows:

```
AN   B    I11.0
A    B    M311.0
R    B    M311.0         ; resetting the edge marker
CM        -DM11serIO     ; call data module DM11
A    B    M311.7         ; evaluate answer ?
JPC       -checkRsp
A    B    I11.0          ; query input 11.0
AN   B    M311.0         ; formation of positive edge
EMI
S    B    M311.0         ; setting the edgetrigger marker
S    B    M311.7         ; setting the evaluate answer marker
L    BY   K1,A           ; data readin
SLL  BY   A,4            ;function no. is entered in bit 4 in the job
O    BY   K1,A           ; setting start function
T    BY   A,D0           ; save job byte
L    DW   K88,A          ; total length of data range = 88 bytes
T    DW   A,D4
L    W    K20,A          ; offset from beginning of data area to identifier
T    W    A,D8           ;
L    W    K-1,A          ; no channel-specific function
CX   DM10
BX   DB10                ; 2. Datenbaustein öffnen
L    DW   DX48,A         ; read identifier from DM10
T    DW   A,D12
L    DW   K64,A          ; request readin of 64 characters
T    DW   A,D16
CM        -B11SERIO,2    ; call module B11SERIO.PAH
P0   K0                  ; the data area begins at data byte 0
P1   K11                 ; in data module DM11
EM
     -checkRsp
L    DW   D0,A           ; job, status and error codes in reg.
A    B    A.12           ; wait while function is still active
BEB
A    B    M311.7
R    B    M311.7         ; resetting of evaluate answer
A    B    A.15           ; error flag set
JPC  -error
                         ; processing of the data received
EM
     -error              ; error handling
EM
```

11.9.3 Status messages

If an error or status occurs after calling the **data read-in/read-out** function which prevents proper execution of this function, **bit** 7 is set in στατυσ.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_{H}$: Parameter errors caused by incorrect parameter input in the data area.

FFFE$_{H}$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.

FFFD$_{H}$: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.

FFFC$_{H}$: Module **B01APSMN** has not been called.

FFFB$_{H}$: Incorrect offset: The offset is too large (> reserved area) or too small (overwriting of job data occurs).

FFFA$_{H}$: Currently, no NCS material is available; the function request could not be sent.

8000$_{H}$: NC-internal error, cannot be influenced by user.

7FFF$_{H}$: PxRos error, cannot be influenced by user.

7FFF$_{H}$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

2: Invalid job parameters

3: Invalid identifier

10: Data readin/readout was canceled when the interface was closed.

12: There is no operational equipment available at present to which this function could be applied

20: Timeout. Data readin before the timeout has occurred is transferred to the user. Thus, the user can examine the timeout process by comparing the data volume set in the job parameters and the number of characters read-in.

21: Overrun-error in UART

22: Parity-error in UART

23: Framing-error in UART

35: Other error within the input/output logic

11.10 Handwheel data program module (-B14HANDW)

Function

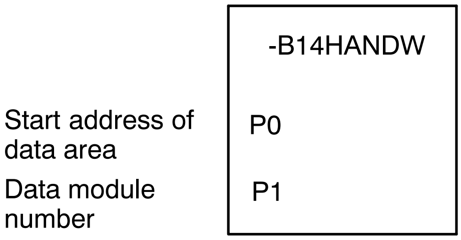
For handwheels in connection with PROFIBUS-DP, the **handwheel data** program module transmits the current value of the handwheel, including the handwheel number, to the NC.
Since the data are transmitted via PROFIBUS-DP, the data from the input area have to be transferred to the NC with the aid of this APS module.

11.10.1 Module call

The module may be called once or repeatedly in a PLC cycle; calls may be unconditional or conditional.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, parameter P1 contains the data module number. In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K–1 must be entered in parameter P1.

This function is activated when bit 0, start function, is set in the job byte and remains set for at least one PLC cycle.
Bit 0 may be reset via the PLC sequential program with the function active message. Otherwise, the start function signal is reset automatically when the function has been carried out.

In the case of program module B14HANDW, the data area, the address of which is defined in parameter P0, has the following structure:

MSB LSB

Bit 31 . . . . 16 15 0

<table>
<tr><td>Error code (2 bytes)</td><td>Status (1 byte)</td><td>Job (1 byte)</td></tr>
<tr><td colspan="3">Length of the total data area reserved (4 bytes)</td></tr>
<tr><td>Channel number</td><td colspan="2">Offset</td></tr>
<tr><td>Input value of the handwheel</td><td colspan="2">Handwheel number</td></tr>
</table>

Job

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle.
Bit 0 may be reset via the PLC sequential program in dependence on the **function active** message. Without resetting via the PLC sequential program, **start function** is reset automatically when the function has been carried out.

Status

The value of status indicates the current condition of the job:

Bit	**Status**
0-3	Reserved
4	function active
5-6	Reserved
7	error flag

As long as bit 4 function active is set, the job has not been completed. If an error occurs during execution of a job, the module sets bit 7 error flag.

Error code

If an error occurs, the module transfers the error code indicating the cause of the error (see page 11–45).

Size of data area

To prevent data from being overwritten accidentally, the maximum number of bytes, earmarked for this function, are transmitted here.

Offset channel no.

Not relevant for this module.

Handwheel number

Transmitting the number of the handwheel for which the data are transferred: (Permissible range: 1 – 2)

Input value of the handwheel

Current value in increments.

11.10.2 Configuration example

Handwheel 1 is active and the values of the handwheel are positioned starting at input 56.
The data area is to begin at marker 1400.
The module is called cyclically when the handwheel mode is active.

Module -B14HANDW.PAH could be called as follows:

```
A     B     –Handr          ; handwheel mode active ?
JPCI        –end

S     B     M1400.0         ; setting of start function
L     DW    K24,A           ; definition of parameters to be transferred
T     DW    A,M1404         ; total length of data area = 24 bytes
L     W     K–1,A           ; dummy-offset
T     W     A,M1408
L     W     K–1,A           ; dummy channel number
T     W     A,M1410
L     W     K1,A            ; handwheel number 1
T     W     A,M1412
L     W     E56,A           ; value of the handwheel
T     W     A,M1414
CM    –B14HANDW             ; B14HANDW module call
P0    M1400                 ; the data area begins at marker 1400
P1    K–1                   ; not in a data module

; check whether an error has occurred
; the error check is performed directly after the  call because data are trans-
ferred
A     B     A.15            ; error flag set ?
JPCI        –end
; error handling

–end
EM
```

11.10.3 Status messages

If an error or status occurs after calling the function which prevents proper execution of this function, **bit 7** is set in **status** and the type of error or status is represented in the form of a binary-coded number in the **error code**.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_H$: Parameter errors caused by incorrect parameter input in the data area.

FFFE$_H$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.

FFFC$_H$: Module **B01APSMN has not been called.**

FFFA$_H$: Currently, no NCS material is available; the function request could not be sent.

8000$_H$: NC-internal error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

1 : Invalid handwheel number.

11.11 Read/Input SERCOS ident. number program module (-B15IDTNR)

Function

The program module **read/input SERCOS ident. number** allows reading and inputting ident. numbers from and to the drive modules. The format of values to be read or input is a 4byte, unsigned integer, i.e. the format of values as saved in the drive modules. Any conversion into another format must be done in the user program. When a value is read, the function also supplies the attribute that describes the ID number. When a value is input to the drives, the attribute must be included.

As of V5.1.8 also the values (for all axes) of an ident no. transferred from the cyclic message to the NC can be read. For an ident. number of an axis not listed in the cyclic message, the value 80000000_H is returned for the respective axis.

11.11.1 Module call

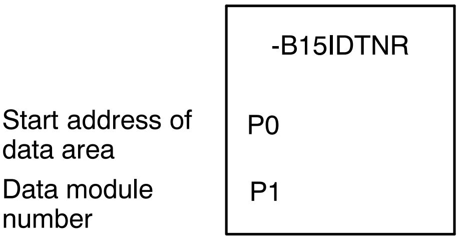
This module can be called unconditionally or conditionally.

If the module is called repeatedly, please note that a different data area must be parameterized for each call.

☞ **Repeated module calls with requests for data transfer directly on the service channel (noncyclical data) may strain the capacity of the service channel! Therefore, a 500 ms interval between two module calls is recommended!**

Module parameters

The module is called with the two parameters P0 and P1.

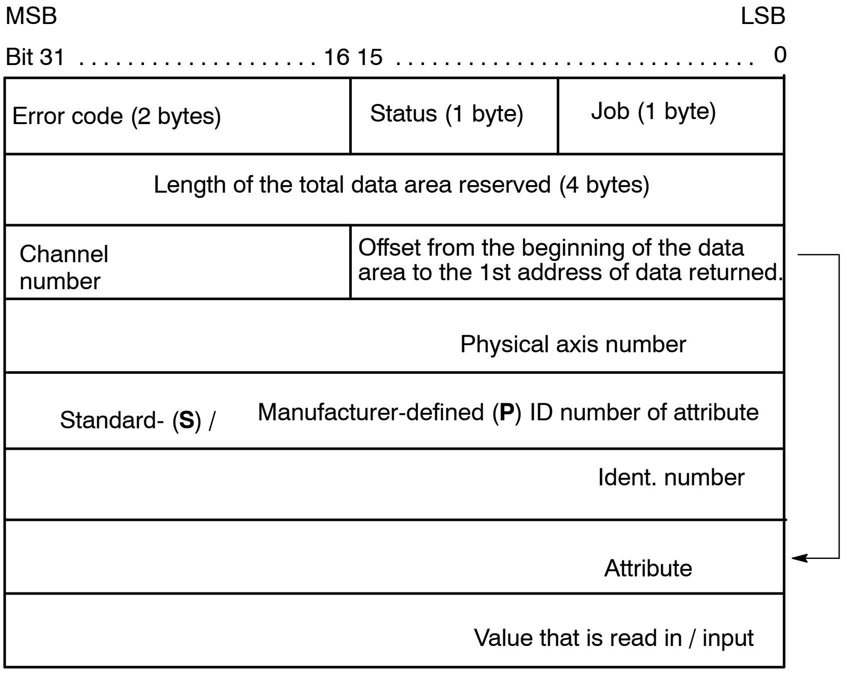


The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

In the case of program module **-B15IDTNR**, the data area, the address of which is defined in parameter P0, has the following structure:



MSB LSB

Bit 31 . . . . 16 15 0

Error code (2 bytes)	Status (1 byte)	Job (1 byte)
Length of the total data area reserved (4 bytes)		
Channel number	Offset from the beginning of the data area to the 1st address of data returned.	
Physical axis number		
Standard- (**S**) / Manufacturer-defined (**P**) ID number of attribute		
Ident. number		
Attribute		
Value that is read in / input		

Reading from the cyclic telegram:

As of version 5.2.2, the NC can read the values of ident. number (as specified in the cyclic message transmitted from the drives to the NC) without any **transmission delay time**.

In the case of program module **-B15IDTNR**, the data area for read-outs from the **cyclic** message, the address of which is defined in parameter P0, has the following structure:

MSB LSB

Bit 31 . . . . 16 15 0

Error code (2 bytes)	Status (1 byte)	Job (1 byte)
Length of the total data area reserved (4 bytes)		
Channel number	Offset from the beginning of the data area to the 1st address of data returned.	
Physical axis number		
Standard **(S)** / manufacturer-dependent **(P)**		
Ident. number		
Ident. number value of 1st physical axis		
Ident. number value of 1st physical axis		
:		
Ident. number value of nth physical axis		

The **job byte** is structured as follows:

Bit	**Function**
0	Start function
1-3	Reserved
4-7	Function number:

1: Read the attribute and the current value of the ident. no.
2: Input a new ident. no. value.
3: Read the ident. no. attribute only.
4: Read an ident no. from the cyclic message. Starting at the address defined by the specified offset, readouts from the cyclic message provide the values of all axis ident. numbers transferred. For ident. numbers not listed in the cyclic message, the value 80000000$_{hex}$ is returned for the respective axis. The minimum interval for value readouts is 100 ms.
5: Same as above, 4, but the values are provided **without** any transmission time delay.

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle.

Start function may be reset by the PLC sequential program with the **function active** message. If not reset by the PLC sequential program, **start function** is automatically reset when the function is finished.

Bit 4, **function active**, remains set in the status until the function has been carried out, at least, however, for the duration of the PLC cycle.

When the data has been returned or input, the function is completed and bit 4, **function active**, is reset in the status.

Offset

The offset is the space between the beginning of the area (job) and the address where the attribute and the current value of the ident. no. are to be stored, or, resp., where the attribute and the new value of the ident. no. are currently stored.
With read-outs of ident. numbers from the cyclic telegram, this is where the values (starting with the 1st axis) are stored.

Channel no.

This parameter is irrelevant to this module.

Physical axis number

Physical axis number of the axis whose ident. no. is to be read or input. This parameter is irrelevant in the context of read-outs from the cyclic message.

Standard / manufacturer-defined ID No.

To distinguish between standard and manufacturer-dependent ident. numbers to be read out or input, ASCII characters **S** or **P** must be entered here.

Ident. no.

The ident. no. whose value is to be read or input.

Attribute

Attribute of the returned ident. no.

Value

Current or new value of the specified ident. no.

ID number value of nth physical axis

Current or new value of the ident. no. read out from the cyclic message.

☞ **To reduce high system loads due to communication, ident. numbers should be read out from cyclic messages using function ”4” if the respective application allows this.**

11.11.2 Configuration example

The current value of ident. no. **S-0-0108, Feedrate Override** of physical axis 1 is to be set at 50%. The data area containing the job parameters is to begin at byte 0 in data module DM15 and the function is to be started with the rising edge on input I15.0. The attribute of the respective ident. no. has already been read out and stored beginning with D24 of the data module. The attribute of the corresponding ID number has already been read and stored starting at D24 of the data module.

Module –B15IDTNR.PAH could be called as follows:

```
A    B    M315.0
R    B    M315.0
R    B    M310.0        ; resetting the edge marker
CM        -DM15IDTNR    ; call data module DM15
A    B    M315.7        ; evaluate answer ?
JPC       -checkRsp
A    B    I15.0         ; query input 15.0
AN   B    M315.0        ; formation of positive edge
EMI
S    B    M315.0        ; setting the edgetrigger marker
S    B    M315.7        ; setting the evaluate answer marker
L    BY   K2,A          ; shifting the function no. of ident. no. input
SLL  BY   A,4           ; to the upper half of the byte
O    BY   K1,A          ; set Start function
T    BY   A,D0          ; save job byte
L    DW   K32,A         ; total length of data range = 32 bytes
T    DW   A,D4
L    W    K24,A         ; 24byte offset to attribute
T    W    A,D8          ; of the ident. number
L    W    K-1,A         ; channel no. is irrelevant
CX   DM10
L    DW   K1,A          ; physical axis number 1
T    DW   A,D12
L    W    K' S',A       ; standard ident. number (note
T    W    A,D18         ; sequence of characters!)
L    DW   K108,A        ; ident. no. 108
T    W    A,D16
L    DW   K5000,A       ; 50% (unit in 0.01%)
T    W    A,D28
CM        -B15IDTNR,2   ; module call B15IDTNR.PAH
P0   K0                 ; the data area begins at data byte 0
P1   K15                ; in data module DM15
          -checksp
L    DW   D0,A          ; job, status and error codes in reg.
A    B    A.12          ; wait while function is still active
BEB
A    B    M315.7
R    B    M315.7        ; resetting of evaluate answer
A    B    A.15          ; error flag set
EMI                     ; function completed without errors
```

```
                -error
                                        ; error handling
                                        ; e.g. repeat the job if the service channel
                                        ; to the drive was occupied
                                        ; (error No. 9H).
EM
```

11.11.3 Status messages

If an error or status occurs after calling the **read/input SERCOS ident. number** function which prevents proper execution of this function, **bit 7** is set in **status**.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

$FFFF_H$: Parameter errors caused by incorrect parameter input in the data area.
$FFFE_H$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.
$FFFD_H$: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.
$FFFC_H$: Module **B01APSMN** has not been called.
$FFFA_H$: Currently, no NCS material is available; the function request could not be sent.
8000H: NC-internal error, cannot be influenced by user.
$7FFF_H$: PxRos error, cannot be influenced by user.
$7FFF_H$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

3_H: NCS channel currently busy
6_H: Unknown ident. no.
9_H: Access conflict on service channel
E_H: Handshake timeout
F_H: Busy timeout
16_H: Transmitted datum too short
17_H: Transmitted datum too long
18_H: Element cannot be changed
19_H: Element cannot be changed at the moment (different phase)
$1A_H$: Data smaller than min. value
$1B_H$: Data greater than max. value
$1C_H$: Invalid operating datum (e.g. bit combination)
$1D_H$: SERCOS ring currently unavailable (run-up)
$1e_H$: Incorrect attribute received
$1F_H$: Invalid or non-SERCOS (dummy) system axis
21_H: Data length exceeds service container size

Program modules for iPCL

28_H:	Max. number of simultaneously active commands exceeded
29_H:	Command change bit timeout
32_H	Conversion error (conversion to SERCOS representation or vice versa)
$1F4_H$:	System axis not available as drive
$1F5_H$:	Configuration error (system axis no. and axis no. on ring are inconsistent)

11.12 Edit tool list program module (-B20WZLST)

Function

With the **edit tool list (-B20WZLST)** module, you can request lists of tool data records or delete lists of tool data records.

- **Searching for tool data records**
  To search for tool data records, you may use up to 7 search criteria to be linked by logic AND operators. If less than 7 search criteria are used, 0 must be entered in the search word for any search criteria remaining unused.
  With this module you can also search for the next data record in the memory location that follows a reference memory location you have specified. If no such data record is found, this module returns **--1** for the next memory location number. If no search for the next memory location is to be conducted, -1 must be entered for the reference sector.
- **Deleting tool data records**
  To delete data records, you may specify a list of data records to be deleted. This module will then delete each of these data records by overwriting their contents with 0, with the exception of the sector and location numbers.
  The space between the beginning of the whole data area and the start index address must be entered in the offset. This allows for the same data area to be used both for requesting and deleting a list of tool data records.Thus, the very same tool data list requested on the basis of specific search criteria can then be used to delete these data records.
  With the start index, the user also transfers the index of the 1st data record on the list to be deleted as well as the number of data records to be deleted.

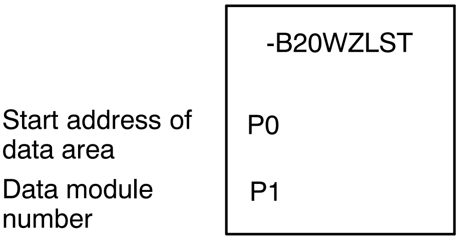
11.12.1 Module call

The module may be called once or repeatedly in a PLC cycle; calls may be unconditional or conditional.

If the module is called again before the data from the previous call is received, please note that a different data area must be activated.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

In the case of program module **-B20WZLST**, the data area, the address of which is defined in parameter P0, has the following structure:

MSB LSB

Bit 31 . . . . 16 15 0

Error code (2 bytes)	Status (1 byte)	Job (1 byte)
Length of the total data area reserved (4 bytes)		
Channel number	Offset	
Table no.		
Start sector		
Start-Platz		
Reference location		
Reference location		
1st search term		
1st search content		
1st search mask		
2nd search term		
2nd search content		
2nd search mask		
:		
7th search term		
7th search content		
7th search mask		
only for search word=6 (query_string) character string (max. 32 bytes)		
Number of data records	Next memory location/start index	
1st data block: loc. No.	1st data block: sector No.	
2nd data block: loc. No.	2nd data block: sector No.	
3rd data block: loc. No.	3rd data block: sector No.	
:	:	
last data block: loc. No.	last data block: sector No.	

The following functions can be activated in the job:

Bit	**Function**	
0	Start function	
1	Reserved	
2-7	Function number:	
	1:	Requesting a tool list, with the tool table to be searched from its very beginning
	2:	Requesting a tool list, with the tool table to be searched from the specified start sector/start location on
	3:	Deleting data records in accordance with the list
	4-7:	Reserved

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle.

Start function may be reset by the PLC sequential program with the **function active** message.

If not reset by the PLC sequential program, **start function** is automatically reset when the function is finished.

Bit 4, **function active**, remains set in the status until the function has been carried out, at least, however, for the duration of the PLC cycle.

Execution of this function is complete as soon as the data feedback is received. Then, status bit 4, **function active**, is reset.

Search term

The following values may be entered for selecting a search word:

0	Ignore search term	
1	Sector	
2	Tool number	(query_int 1)
3	Duplo number	(query_int 2)
4		(query_int 3)
5		(query_int 4)
6	Tool name	(query_string)
7	Tool identification	(bitfield)

11.12.2 Configuration example

A search is to be conducted in the list of all tools with tool number 123 located in the magazine (sector 2) and whose tool life has expired.

The area containing the job data is to begin at byte 0 in data module DM20.

The function is to be started with the rising edge on input **I20.0.**

Module **-B20WZLST** could be called as follows:

```
AN   B    I20.0
A    B    M320.0
R    B    M320.0          ; resetting the edge marker
CM        -DM20WzgL       ; call data module DM20
A    B    M320.7          ; evaluate answer ?
JPC       -checkRsp
A    B    I20.0           ; query input 20.0
AN   B    M320.0          ; formation of positive edge
JPI       -end
S    B    M320.0          ; setting the edgetrigger marker
S    B    M320.7          ; setting the evaluate answer marker
S    B    D0.0            ; setting start function
                          ; definition of parameters to be transferred
L    BY K1,A              ; function number for search from beginning
                          of table
SLL  BY A,2               ; shifting the function no. to the respective
                          ; location in the job byte
O    BY   K1,A            ; start function
T    BY   A,D0
L    DW   K256,A          ; total length of data range = 256 bytes
T    DW   A,D4
L    W    K116,A          ; offset = 148 bytes
T    W    A,D8,
L    W    K-1,A           ; no channel-specific function
CX   DM10
L    DW   K1,A            Table No. 1
T    DW   A,D12
L    DW   K-1,A           ; no start sector
T    DW   A,D16
L    DW   K-1,A           ; no start location
T    DW   A,D20
L    DW   K-1,A           ; no reference sector
T    DW   A,D24
L    DW   K-1,A           ; no reference location
T    W    A,D28
L    DW   K1,A            ;1st search word is ”sector”
T    DW   A,D32
L    DW   K2,A            ; search is limited to sector 2 (tool magazine)
T    DW   A,D36
L    DW   K0,A            ; no mask, as search word not ”identifier”
T    DW   A,D40
L    DW   K2,A            ; 2nd search word is ”tool number”
```

```
T     DW   A,D44
L     DW   K123,A           ; search for tool number 123
T     DW   A,D48
L     DW   K0,A             ; no mask, as search word not ”identifier”
T     DW   A,D52
L     DW   K7,A             ; 3rd search word is ”identifier”
T     DW   A,D56
L     DW   K4H,A            ; search for tools identified as ”defective”
                            ; ”defekt”
T     DW   A,D60
L     DW   K4,A             ; defective tool identifier bit only
T     DW   A,D64
L     DW   K0,A             ; 4th search word is ”not used”
T     DW   A,D68
L     DW   K0,A             ; 5th search word is ”not used”
T     DW   A,D80
L     DW   K0,A             ; 6th search word is ”not used”
T     DW   A,D92
L     DW   K0,A             ; 7th search word is ”not used”
T     DW   A,D104
CM         -B20WZLST,2      ; call module -B20WZLST
P0         K0               ; the data area begins at data byte 0
P1         K20              ; in data module DM20
JP         -end
           -checkRsp
L     DW   D0,A             ; job, status and error codes in reg.
A     B    A.12             ; wait while function is still active
JPC        -end
A     B    M320.7
R     B    M320.7           ; resetting of evaluate answer
A     B    A.15             ; error flag set
JPI        -FktOk
                            ; error handling
JP         -end
           -FktOk
                            ; processing of received data
           -end
EM
```

Program modules for iPCL

11.12.3 Status messages

If an error or status occurs after calling the edit tool list function which prevents proper execution of this function, **bit 7** is set in **status**.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_H$: Parameter errors caused by incorrect parameter input in the data area.
FFFE$_H$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.
FFFD$_H$: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.
FFFC$_H$: Module **B01APSMN** has not been called.
FFFB$_H$: Incorrect offset: The offset is too large (> reserved area) or too small (overwriting of job data occurs).
FFFA$_H$: Currently, no NCS material is available; the function request could not be sent.
8000$_H$: NC-internal error, cannot be influenced by user.
7FFF$_H$: PxRos error, cannot be influenced by user.
7FFF$_H$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

68: Tool table is empty
72: Data record not found
108: The last data record was read with inhibit and not reenabled
115: The last data record was not read with inhibit
1002: Invalid table no.

11.13 Edit tool data program module (-B21WZDAT)

Function

The **edit tool data (-B21WZDAT)** program module offers various functions for reading and changing tool data.

The various functions can be selected by entering the respective function number in the job byte.

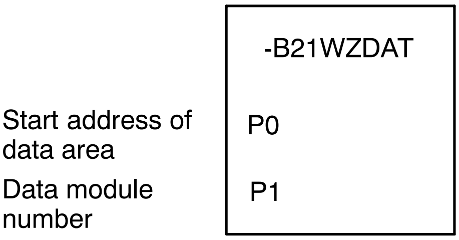
11.13.1 Module call

The module may be called once or repeatedly in a PLC cycle; calls may be unconditional or conditional.

If the module is called again before the data from the previous call is received, please note that a different data area must be activated.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

In the case of program module **-B21WZDAT**, the data area, the address of which is defined in parameter P0, has the following structure:

MSB LSB

Bit 31 . . . . 16 15 0

<table>
<tr><td>Error code (2 bytes)</td><td>Status (1 byte)</td><td>Job (1 byte)</td></tr>
<tr><td colspan="3">Length of the total data area reserved (4 bytes)</td></tr>
<tr><td>Channel number</td><td colspan="2">Offset</td></tr>
<tr><td colspan="3">Table number (source)</td></tr>
<tr><td colspan="3">File name (with functions 8 or 9 only) / sector number (source)</td></tr>
<tr><td colspan="3">Location number (source)</td></tr>
<tr><td colspan="3">Table number (destination, with functions 5 or 6 only)</td></tr>
<tr><td colspan="3">Sector number (destination, with functions 5 or 6 only)</td></tr>
<tr><td colspan="3">Location number (destination, with functions 5 or 6 only)</td></tr>
<tr><td>Location number</td><td colspan="2">Sector number</td></tr>
<tr><td colspan="3">Tool number (query_int 1)</td></tr>
<tr><td colspan="3">Duplo number (query_int 2)</td></tr>
<tr><td colspan="3">(query_int 3)</td></tr>
<tr><td colspan="3">(query_int 4)</td></tr>
<tr><td colspan="3">Tool identification (bit field)</td></tr>
<tr><td colspan="3">Tool name (query_string)
32 characters</td></tr>
<tr><td colspan="3">Tool data (data_int)
40 values of 4 bytes</td></tr>
<tr><td colspan="3">32character string (data_string)</td></tr>
</table>

The job byte is structured as follows:

Bit	**Function**
0	Start function
1	Reserved
2-7	Function number:

1: Reading a tool data record without inhibiting further access to it.

2: Reading a tool data record while inhibiting further access to it. This function must be called if tool data is to be saved in the data base after being edited. Until access to the inhibited data record is reenabled, the PLC user program is denied access also to any other data records stored in the data base.

3: Saving a tool data block. Using this function, this data block is re-enabled at the same time so that access to this data block is again permitted. Access to this data block is again permitted.

4: At the same time, this function reenables access to this function

5: Exchanging two data blocks within the database.
Here, the contents of two data blocks in the database are exchanged.

6: Copying the contents of a data record (source record) to another data record (destination record).

7: Deleting the contents of a data record (source record).

8: Saving the tool table in an ASCII file. Together with the file name under which the tool table content is to be saved, you may specify the whole path name, which must not exceed 50 characters (incl. end identifier NUL (00H). To ensure that the file can be read in also from the operator interface, it must be saved under /database/tooltab1.dat. For backup purposes, this file can be output also on a serial interface.

9: Reading in the contents of the tool table from an ASCII file. With this function you can read in an ASCII file created by operating interface input or with function **8**.

☞ **The function described above overwrites the contents of the current tool table.**

10: Deleting all data records of the tool table. This function deletes the entire table. Then the control must be restarted; the table is regenerated at this time. After the restart, modifications to the configuration file for the tool table become effective.

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle.

Start function may be reset by the PLC sequential program with the **function active** message.

If not reset by the PLC sequential program, **start function** is automatically reset when the function is finished.

Bit 4, **function active**, remains set in the status until the function has been carried out, at least, however, for the duration of the PLC cycle.

Execution of this function is complete as soon as the data feedback is received. Then, status bit 4, **function active**, is reset.

11.13.2 Configuration example

The data of the tool in location 5, sector 2, table 1 is to be read while any further access to this data is to be inhibited.
The area containing the job data is to begin at byte 8 in data module DM21.
The function is to be started with the rising edge on input **I21.0**.

Module **-B21WZDAT** could be called as follows:

```
AN   B    I21.0
A    B    M321.0
R    B    M321.0           ; resetting the edgetrigger marker
CM   -DM21WzgDb            ; call data module DM21
A    B    M321.7           ; evaluate answer ?
JPC       -checkRsp
A    B    I21.0            ; scan input 21.0
AN   B    M321.0           ; formation of positive edge
JPI       -end
S    B    M321.0           ; setting the edgetrigger marker
S    B    M321.7           ; set the evaluate answer marker
S    B    D8.0             ; setting start function
                           ; definition of parameters to be transferred
L    BY   K1,A             ; function number for reading a data record
SLL BY    A,2              ; shifting the function no. to the respective
                             location in the job byte
O    BY   K1,A             ; start function
T    BY   A,D8
L    DW   K292,A           ; total length of data range = 292 bytes
T    DW   A,D12            ; (36 bytes job data + 256 bytes
                             location no.
L    W    K36,A            ; offset = 36 bytes
T    W    A,D16,
L    W    K-1,A            ; no channel-specific function
T    W    A,D18
L    DW   K1,A             ; Table No. 1
T    DW   A,D20
L    DW   K2,A             ; sector 2
T    DW   A,D24
L    DW   K5,A             ; location 5
T    W    A,D28
L    DW   K-1,A            ; no destination table number
T    DW   A,D32
L    DW   K-1,A            ; no destination sector number
T    DW   A,D36
L    DW   K-1,A            ; no destination location number
```

```
T     DW   A,D40
CM         -B21WZDAT,2   ; call module -B21WZDAT
P0         K8            ; the data area begins at data byte 8
P1         K21           ; in data module DM21
JP         -end
           -checkRsp
L     DW   D8,A          ; job, status and error codes in reg.
A     B    A.12          ; wait while function is still active
JPC        -end
A     B    M321.7
R     B    M321.7        ; resetting of evaluate answer
A     B    A.15          ; error flag set
JPI        -FktOk
                         ; error handling
JP         -end
           -FktOk
                         ; processing of received data
           -end
EM
```

11.13.3 Status messages

If an error or status occurs after calling the **edit tool data** function which prevents proper execution of this function, **bit 7** is set in στατυσ.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF_{H}: Parameter errors caused by incorrect parameter input in the data area.
FFFE_{H}: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.
FFFD_{H}: Data overflow, the reserved data area is used to capacity, not enough space to store all the data.
FFFC_{H}: Module **B01APSMN** has not been called.
Incorrect offset: The offset is too large (> reserved area) or too small (overwriting of job data occurs).
FFFA_{H}: Currently, no NCS material is available; the function request could not be sent.
8000_{H}:: NC-internal error, cannot be influenced by user.

FFFB_{H}:
7FFF_{H}: PxRos error, cannot be influenced by user.
7FFF_{H}: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

68: The specified tool table is empty.
72: The data record specified in the job data could not be found.
108: The previous data record was inhibited and has not yet been reenabled.
115: The data record was not inhibited before it was saved to the database or reenabled.
1001: Invalid table no.

Invalid table no. The following error codes can appear only with functions 8 or 9:

1010: The transferred file name is invalid.
1011: The file could not be opened.
1012: The file could not be closed.
1013: Error when reading the file.
1014: Error when writing the file.
1016: The file data are inconsistent.

11.14 External tool compensation program module (-B22WZKOR)

Function

The **external tool compensation** program module allows input of compensation values of compensation groups 1 and 2 (1^{st} and 2^{nd} external tool compensation).

1st compensation group:

The PLC program transfers the values for one of a maximum of 8 paired compensation values of ”radius and length compensation” on an NC channel.

2**nd compensation group** (as of software version V4.4.1):

The PLC program transfers the values for up to 8 possible compensation value sets of ”radius compensation, up to 3 length compensation values and tool edge position” within an NC channel.

☞ **As of software version V4.4.1, several compensation inputs of the 1^{st} or 2^{nd} compensation group can be combined in one module call.**

The units of these values are to be defined in MACODA parameter 9020 00010 and must be scaled to 0.0001 for transfer.

Example: MACODA parameter 9020 00010 is set to 0 (unit = mm). Entries of compensation values must be scaled to 0.1m.

The external tool compensations of the 1_{st} and 2_{nd} compensation groups are activated and deactivated with the following G functions:

Compensation group	Transfer parameter	Activation with	Deactivation with

1	D = radius compensation
	H = length compensation
		G145,
		G245..G845
			G146

2	R = radius compensation L_1, L_2, L_3 = Length compensation or offset parameters parameter SL = tool edge position	G147, G247..G847	G148

Compensation values become active only in the next program block to be prepared.
For compensation values to become active immediately upon their input, NC block preparation must be suspended (e.g., by **WAIT or block transfer inhibit**) or NC blocks already prepared must be prepared again (by **cancel distance to go**).

Program modules for iPCL

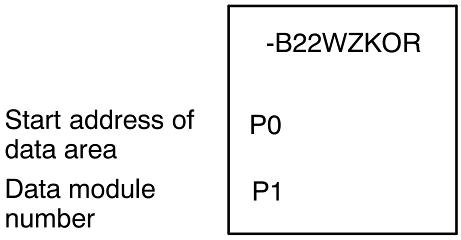
11.14.1 Module call

This module can be called unconditionally or conditionally.

If the module is called repeatedly, please note that a different data area must be parameterized for each call.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

In the case of program module **-B22WZKOR**, the data area, the address of which is defined in parameter P0, has the following structure as of V4.4.1:

MSB LSB

Bit 31 . . . . 16 15 0

Error code (2 bytes)	Status (1 byte)	Job (1 byte)
Length of the total data area reserved (4 bytes)		
Channel number	Offset	
Type of compensation	Number of compensations	
1. st compensation value (scaled to 0.0001)		
..		
..		
n^th compensation value (scaled to 0.0001)		

The **job byte** is structured as follows:

Bit	Function
0	Start function
1-2	Tool compensation group Permissible values: 0 Input as previously (up to V4.4.0) of one radius compensation value and one length compensation value in DW3 and DW4. 1 Input in 1st compensation group; several compensations can be input with one call. 2 Input in 2st compensation group; several compensations can be input with one call.
3	Reserved
4-7	Compensation index, permissible values: 1 - 8

The function is started when bit 0 is set to logic 1 in the job byte. Compensation values are transferred to the NC immediately. The **function active** bit remains set until the NC has acknowledged the compensation input.

Because module **-B22WZKOR** does not return any data, no **offset** needs to be defined in double word DW2.

The NC channel on which the compensation values are to become active is specified by entering the respective **channel number** in double word DW2. The structure in which compensation values are transferred depends on which compensation group is specified in the job byte:

Compensation group 0:

For existing PLC programs, the data range is transferred in its former structure (up to V4.1.0) by specifying compensation group 0. Thus, existing PLC programs remain operable without any changes required.

In this case, the compensation values are entered in double words **DW3** and **DW4**. As was the case in the past, inputs can be made in compensation group 1 only.

MSB LSB

Bit 31 . . . . 16 15 0

Error code (2 bytes)	Status (1 byte)	Job (1 byte)
Length of the total data area reserved (4 bytes)		
Channel number	Offset	
Radius compensation value (scaled to 0.0001)		
Length compensation value (scaled to 0.0001)		

Compensation group 1 or 2:

In this case, the **number of compensations** is entered in the low-order word and the **type of compensation** in the high-order word of DW3.

Bit	**Function**
0-15	Number of compensations
16-31	Type of compensation Permissible values: 1 Compensation with 1 length compensation value 2 Compensation with 1 radius and 1 length compensation value 3 Compensation with 1 radius and 2 length compensation values and with tool edge position 4 Compensation with 1 radius and 3 length compensation values and with tool edge position

☞ **Für die 1. Korrekturgruppe ist derzeit nur der Korrekturtyp 2 (1 Radius- und 1 Längenkorrekturwert) zulässig.**

Compensation values, which must be entered with 0.0001 scaling, follow from DW4 onwards.

The number of compensations refers to the type of compensation and varies between 1, 2, 4 or 5 values.

Example: For 1 compensation of compensation type 4, a total of 5 compensation values must be entered, which include ”1 radius and 3 length compensation values and the tool edge position”.

For compensations involving several length compensation values, inputs of the individual **length compensation** values must be made in the following **order**:

1. **L3**: Length compensation along the axis that is specified as the L3 length compensation axis in MACODA parameter 7050 00420, assignment compensation – logical axis.
2. **L1**: Length compensation along the axis that is specified as the L1 length compensation axis in MACODA parameter 7050 00420, assignment compensation – logical axis.
3. **L2:** Length compensation along the axis that is specified as the L2 length compensation axis in MACODA parameter 7050 00420, assignment compensation – logical axis.

11.14.2 Configuration example

A radius compensation of 0.1234 mm and a length compensation of 100 mm is to be input as the 2nd correction (**G245**) in channel 1.
The function is to be started with the rising edge on input **I22.0.**

The data area with the job parameters begins at byte 12 in data module DM22.

Module **-B22WZKOR** could be called as follows:

```
AN   B    I22.0
A    B    M322.0
R    B    M322.0          ; resetting the edge marker
CM        -DM22WZKOR      ; call data module DM22
A    B    M322.7          ; evaluate answer ?
JPC       -checkRsp
A    B    I22.0           ; query input 22.0
AN   B    M322.0          ; formation of positive edge
EMI
S    B    M322.0          ; setting the edgetrigger marker
S    B    M322.7          ; setting the evaluate answer marker
L    BY   K2,A            ; 2nd compensation (G245)
SLL  BY   A,4             ; compensation index begins at bit 4 in job
O    BY   K1,A            ; setting start function
T    BY   A,D12           ; save job byte
L    DW   K20,A           ; definition of parameters to be transferred
T    DW   A,D16           ; Gesamtlänge des Datenber. 20 Byte,
L    W    K0,A            ; no offset required as no data
T    W    A,D20           ; are returned,
L    W    K1,A            ; compensation values on channel 1
T    W    A,D22
L    DW   K123456,A       ; radius compensation value = 12.3456 mm
T    DW   A,D24
L    DW   K1000000,A      ; length compensation value = 100 mm
T    W    A,D28
CM        -B22WZKOR,2     ; call module -B22WZKOR
P0        K12             ; the data area begins at data byte 12
P1        K22             ; in data module DM22
EM
-checkRsp
L    DW   D12,A           ; job, status and error codes in reg. A
A    B    A.12            ; wait while function is still active
BEB
A    B    M322.7
R    B    M322.7          ; resetting of evaluate answer
A    B    A.15            ; error flag set
EMI
                          ;   error handling
EM
```

11.14.3 Status messages

If an error or status occurs after calling the **external tool compensation** function which prevents proper execution of this function, **bit 7** is set in **status**.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_\text{H}$: Parameter errors caused by incorrect parameter input in the data area.
FFFE$_\text{H}$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.
FFFC$_\text{H}$: Module **B01APSMN** has not been called.
FFFA$_\text{H}$: Currently, no NCS material is available; the function request could not be sent.
8000$_\text{H}$: NC-internal error, cannot be influenced by user.
7FFF$_\text{H}$: PxRos error, cannot be influenced by user.
7FFF$_\text{H}$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

1$_\text{H}$: Invalid channel no.
13$_\text{H}$:: Invalid correction index
14$_\text{H}$: Max. number of corrections exceeded (may be displayed with the correction index).
15$_\text{H}$: Invalid type of correction
16$_\text{H}$: Invalid tool edge position
17$_\text{H}$: Invalid compensation group.

11.15 External zero shift program module (-B23XTNPV)

Function

The external zero shift program module allows inputs of axis zero shift values on an NC channel. The PLC program transfers these values for the specified number of axes and axis zero shift groups. Up to 3 axis zero shifts can be specified for each axis; they are assigned according to the sequence of axes, beginning with the 1st physical axis. The number of axis zero shift values as specified for each axis must be entered.

Use G functions **G160**, **G260** or **G360** to select one of the three axis zero shifts that can be made for each axis. To deactivate the external axis zero shift function, use **G167**.

A newly input axis zero shift value becomes active only with the **next program block** to be processed ιν **block preparation**. For axis zero shift values to become active immediately upon their input, block preparation must be suspended (e.g., by **WAIT** or **block transfer inhibit**) or program blocks already prepared must be prepared again (by **cancel distance to go**).

As of version V5.2.2, you may assign external axis zero shift values to axes also under their **physical axis** numbers (as opposed to their logical axis numbers). The selection as to whether the logical or the physical axis number is to be transferred is made by setting the appropriate function code in the job byte. If **physical axis** numbers are used, transfer of values is axis-related, with the axis index being followed by up to 3 axis zero shift values.

CAUTION
Axis zero shift values of axes not included in the message are set to 0 on the channel where the values are transferred!

Axis zero shift values are transferred with 0.0001 scaling in the unit as specified in MACODA parameter 9020 00010.

Example: MACODA parameter 9020 00010 is set to 0 (unit = mm). Entries of axis zero shift values must be scaled to 0.1m for transfer.

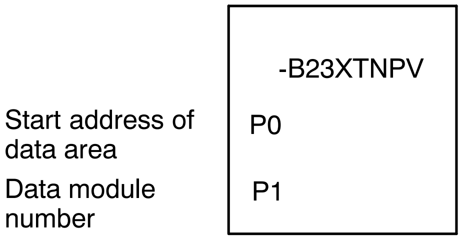
11.15.1 Module call

This module can be called unconditionally or conditionally.

If the module is called repeatedly, please note that a different data area must be parameterized for each call.

Module parameters

The module is called with the two parameters P0 and P1.



The start address (in bytes) of the data area where the function, status word and error message parameters to be transferred are to be stored must be entered in parameter P0.

If the data area is located in a data module, the data module number must be entered in parameter P1.

In cases where the data area is not defined in a data module but, e.g., in the marker field, constant K-1 must be entered in parameter P1.

If **logical axis numbers** are used in program module **–B23XTNPV**, the data range whose address is defined in parameter P0 has the following structure:

MSB LSB

Bit 31 . . . . 16 15 0

Error code (2 bytes)	Status (1 byte)	Job (1 byte)
Length of the total data area reserved (4 bytes)		
Channel number	Offset	
		Number of axes
		Number of axis zero shift (ZS) values for each axis
		1st axis ZS value of the 1st axis (scaled to 0.0001)
		:
		1st axis ZS value of the nth axis (scaled to 0.0001)
		2st axis ZS value of the 1st axis (scaled to 0.0001)
		:
		2st axis ZS value of the nth axis (scaled to 0.0001)
		3st axis ZS value of the 1st axis (scaled to 0.0001)
		:
		3st axis ZS value of the nth axis (scaled to 0.0001)

In the case of program module **-B23XTNPV** with **physical axis** numbers used, the data area, the address of which is defined in parameter P0, has the following structure:

MSB LSB

Bit 31 . . . . 16 15 0

Error code (2 bytes)	Status (1 byte)	Job (1 byte)
Length of the total data area reserved (4 bytes)		
Channel number	Offset to the 1st address of Channel number	
Number of axes		
Number of axis zero shift (ZS) values for each axis		
physical axis number		
1st $_{axis}$ ZS value (scaled to 0.0001)		
2st $_{axis}$ ZS value (scaled to 0.0001)		
3st $_{axis}$ ZS value (scaled to 0.0001)		
physical axis number		
1st $_{axis}$ ZS value ...		
:		
physical axis number		
1st $_{axis}$ ZS value (scaled to 0.0001**)**		
2st $_{axis}$ ZS value (scaled to 0.0001)		
3st $_{axis}$ ZS value (scaled to 0.0001)		

The job byte is structured as follows:

Bit	**Function**
0	Start function
1-7	Reserved
4-7	Function number: 0: Axis ZS value input with logical axis numbers (for structure of data area, see above) 1: Axis ZS value input with physical axis numbers (for structure of data area, see above)

This function is activated when bit 0, **start function**, is set in the job byte and remains set for at least one PLC cycle.Start function may be reset by the PLC sequential program with the **function active** message. If not reset by the PLC sequential program, **start function** is automatically reset when the function is finished.

Bit 4, **function active**, remains set in the status until the function has been carried out, at least, however, for the duration of the PLC cycle. Execution of this function is complete as soon as the data feedback is received. Then, status bit 4, **function active**, is reset.

Offset

Because the **-B23XTNPV** module does not return any data, no offset needs to be defined in double word DW2.

Channel number

The NC channel on which the axis zero shift values are to become active is specified by entering the respective channel number in double word DW2.

Number of axes

Contains the number of axes for which the external axis zero shifts are transferred.

Number of axis zero shift (ZS) values for each axis

Number of axis zero shift values entered for each axis.

Permissible values:

1: Values for the 1st axis zero shift (G160)
2 Values for the 1st and 2nd axis zero shift (G160,G260)
3: Values for the 1st, 2nd and 3rd axis zero shift (G160,G260,G360) (G160,G260,G360)

Axis zero offset values (logical axis number)

Contains the value by which the axis zero point of a single **logical axis** (with logical axis number) is to be offset. The value is transferred so that it is scaled to 0.0001 of the unit from MACODA parameter 9020 00010.

Physical axis number

Index of the physical axis.

1st – 3rd axis zero shift value (physical axis number)

Up to 3 axis zero shift values by which the axis zero point of a single **physical axis** (with physical axis number) is to be shifted. The value is transferred so that it is scaled to 0.0001 of the unit from MACODA parameter 9020 00010.

11.15.2 Configuration example

The following **axis** zero shifts are to be carried out on channel 1:
the 1st axis by 10mm,
the 2nd axis by –30mm and
the 3rd axis by 50mm

and, for **G260**, the axis zero point
the 1st axis by 20 mm,
the 2nd axis by –60 mm and
the 3rd axis by 100 mm.

The area containing the job parameters begins at byte 0 in data module DM23. The function is to be started with the rising edge on input I23.0.

Module -B23XTNPV.PAH could be called as follows:

```
AN   B    I23.0
A    B    M323.0
R    B    M323.0          ; resetting the edge marker
CM        -DM23xtNpv      ; call data module DM23
A    B    M323.7          ; evaluate answer ?
JPC       -checkRsp
A    B    I23.0           ; query input 23.0
AN   B    M323.0          ; formation of positive edge
EMI
S    B    M323.0          ; setting the edgetrigger marker
S    B    M323.7          ; setting the evaluate answer marker
L    BY   K1,A            ; data readin
T    BY   A,D0            ; save job byte
L    DW   K44,A           ; total length of data range = 28 bytes
T    DW   A,D4
L    W    K0,A            ; offset irrelevant
T    W    A,D8            ; are returned,
L    W    K1,A            ; channel no. 1
CX   DM10
L    DW   K3,A            axis ZO values for 1st, 2nd and 3rd axis
T    DW   A,D12
L    DW   K2,A            2nd axis ZO values for each axis (G160,G260)
T    DW   A,D16
L    DW   K100000,A       ; with G160,10mm for 1st axis
T    DW   A,D20
L    DW   K-300000,A      ; with G160, -30mm 2nd axis
T    DW   A,D24
L    DW   K500000,A       ; with G160,50mm for 3st axis
T    W    A,D28
L    DW   K200000,A       ; with G260,20mm for 1st axis
T    DW   A,D32
L    DW   K-600000,A      ; with G260, -60mm 2nd axis
T    DW   A,D36
L    DW   K1000000,A      ; with G260,10mm for 3st axis
T    DW   A,D40
CM        -B23XTNPV,2     ; call module B23XTNPV.PAH
P0   K0                   ; the data area begins at data byte 0
```

```
P1    K23                    ; in data module DM23
      -checksp
L     DW    D0,A             ; job, status and error codes in reg.
A     B     A.12             ; wait while function is still active
BEB
A     B     M323.7
R     B     M323.7           ; resetting of evaluate answer
A     B     A.15             ; error flag set
JPC         -error
                             ; function completed without errors
EM
      -error
                             ; error handling
EM
```

11.15.3 Status messages

If an error or status occurs after calling the **external zero shift** function which prevents proper execution of this function, **bit 7** is set in στατυσ.

In addition, the error code indicates the error type or, resp., the status in binary-coded format.

There are two categories of errors and statuses:

General errors and statuses occurring upon a function call that are caused by incorrect parameter settings or an internal NC error:

FFFF$_H$: Parameter errors caused by incorrect parameter input in the data area.

FFFE$_H$: Operand overflow, the operand data exceeds the limit of the reserved data area, e.g. the highest marker to be addressed.

FFFC$_H$: Module **B01APSMN** has not been called.

FFFA$_H$: Currently, no NCS material is available; the function request could not be sent.

8000$_H$: NC-internal error, cannot be influenced by user.

7FFF$_H$: PxRos error, cannot be influenced by user.

7FFF$_H$: NCS error, cannot be influenced by user.

Function-specific errors or statuses occurring with the acknowledgement that the function has been executed:

1$_H$: Invalid channel no.

10$_H$: Number of axes is greater than the number of configured axes

11$_H$: Number of axis ZS shifts is greater than the maximum permissible number

11.16 Conversion Integer –> ASCII program module (-B97ITOA)

Function

The **conversion Integer** –> **ASCII** module converts an integer into a character string ending on the string delimiter ASCII NUL (00hex).

11.16.1 Module call

Module parameters

The data area containing the job data is defined by parameters P0 and P1 when the module is called.

	-B97ITOA
4-byte integer value	P0
Offset	P1
Data module number	P2

P1 contains the operand address or, resp., the offset if the address is within the data module; P2 contains the number of the data module. If the address is not in a data module, the value to be transferred in P2 must be 1.

☞ **For the conversion to function properly, please make sure that there is enough space for all the characters (max. 11 characters).**

Example 1:

Marker 20 contains the integer to be converted into a character string. This character string is to be saved from marker 200 onwards:

```
CM    -B97ITOA,3          ;
P0    M20                 ; integer value P1 M200
P1    M200                ; Target address for character string
P2    K–1                 ; not in DM
```

Example 2:

Marker 20 contains the integer to be converted into a character string. This character string is to be saved from data word 40 onwards in data module 2:

```
CM    -B97ITOA,3          ;
P0    M20                 ; integer value P1 M200
P1    K40                 ; Target address for character string starting
                            at D40
P2    K2                  ; in DM2
```

11.17 Read system date/time program module (-B98DATTM)

Function

The read system date/time module returns the time, the date and the day of the week.

All this data is returned in the form of byte values.

11.17.1 Module call

Module parameters

The data area containing the job data is defined by parameters P0 through P6 when the module is called.

Meaning of the parameter values:

	-B98DATTM
Seconds	P0
Minutes	P1
Hours	P2
Day	P3
Month	P4
Year	P5
Day of the week	P6

Parameter ranges:

Year	0 – 99
Day of the week	0: Sunday 1: Monday 2: Tuesday 3: Wednesday 4: Thursday 5: Friday 6: Saturday

12 Machine operating panel

☞ **For machine operating panels using KNS, see ”PLC development environment” manual.**

12.1 Machine operating panel with PROFIBUS-DP

The machine user panel with PROFIBUS-DP can be used for the PNC-P.

The logic statuses of all keys, switches and potentiometers as well the data of a connected handwheel are directly mapped in the I/O area of the PROFIBUS-DP which is configurable via WinDP.

CAUTION

Errors and warnings relating to the operating panel bus, if, e.g. a machine operating panel connection is interrupted, will not automatically result in opening the Ready2 contact.

It therefore has to be ensured in the PLC sequential program that functions activated via the machine operating panel do not lead to hazards to persons or machines in case of an error.

Since all input data of the machine operating panels are set to zero in case of error, safety is ensured if either normally closed contacts (n.c.c.) or normally open contacts (n.o.c.) are used depending on the function type.

12.1.1 Signal transmission

The data are located on the inputs and outputs configured via WinDP. The offsets start at zero.

Example:

If the operating panel is positioned on I200/O200, S1 is assigned to input I203.6.

Machine operating panel

Assignment:

S1	n.o.c.	I203.6		H1	O201.0
S2	n.o.c.	I203.7		H2	O201.1
S3	n.o.c.	I204.5		H3	O201.2
S4	n.o.c.	I204.6		H4	O201.3
S5	n.o.c.	I204.7		H5	O201.4
S6	n.o.c.	I207.5		H6	O201.5
S7	n.o.c.	I207.6		H7	O201.6
S8	n.o.c.	I207.7		H8	O201.7
S9	n.o.c.	I206.6		H9	O200.0
S10	n.o.c.	I206.7		H10	O200.1
S11	n.o.c.	I205.0		H11	O200.2
S12	n.o.c.	I205.1		H12	O200.3
S13	n.o.c.	I205.2		H13	O200.4
S14	n.o.c.	I205.3		H14	O200.5
S15	n.o.c.	I205.4		H15	O200.6
S16	n.o.c.	I205.5		H16	O200.7
S17	n.o.c.	I205.6		H17	O203.0
S18	n.o.c.	I205.7		H18	O203.1
S19	n.o.c.	I206.4		H19	O203.2
S20	n.o.c.	I206.5		H20	O203.3
S21	n.o.c.	I201.0	n.c.c. I200.0	H21	O203.4
S22	n.o.c.	I201.1	n.c.c. I200.1	H22	O203.5
S23	n.o.c.	I201.2	n.c.c. I200.2	H23	O203.6
S24	n.o.c.	I201.3	n.c.c. I200.3	H24	O203.7
S25	n.o.c.	I201.4	n.c.c. I200.4	H25	O202.0
S26	n.o.c.	I201.5	n.c.c. I200.5	H26	O202.1
S27	n.o.c.	I201.6	n.c.c. I200.6	H27	O202.2
S28	n.o.c.	I201.7	n.c.c. I200.7	H28	O202.3
S29	n.o.c.	I202.0	n.c.c. I203.0	H29	O202.4
S30	n.o.c.	I202.1	n.c.c. I203.1	H30	O202.5
S31	n.o.c.	I203.2	n.c.c. I202.2	H31	O202.6
S32	n.o.c.	I203.3	n.c.c. I202.3	H32	O202.7
S33	n.o.c.	I203.4	n.c.c. I202.4		
S34	n.o.c.	I203.5	n.c.c. I202.5		
SS1	Rotary switch		I206.0 ... I206.3		
SS2	Rotary switch		I204.0 ... I204.4		
SS3	Rotary switch		I207.0 ... I207.4		
ST5	Pin2 .. Pin6	I208.7 ... I208.3			
ST3	Pin2	I208.0			
	Pin3	I208.2			
	Pin6	I209.1			
	Pin7	I209.3			
	Pin8	I209.5			
	Pin9	I209.7			
	Pin15	I208.1			
	Pin18	I209.0			
	Pin19	I209.2			
	Pin20	I209.4			
	Pin21	I209.6			

Machine operating panel

Machine operating panel with PROFIBUS-DP

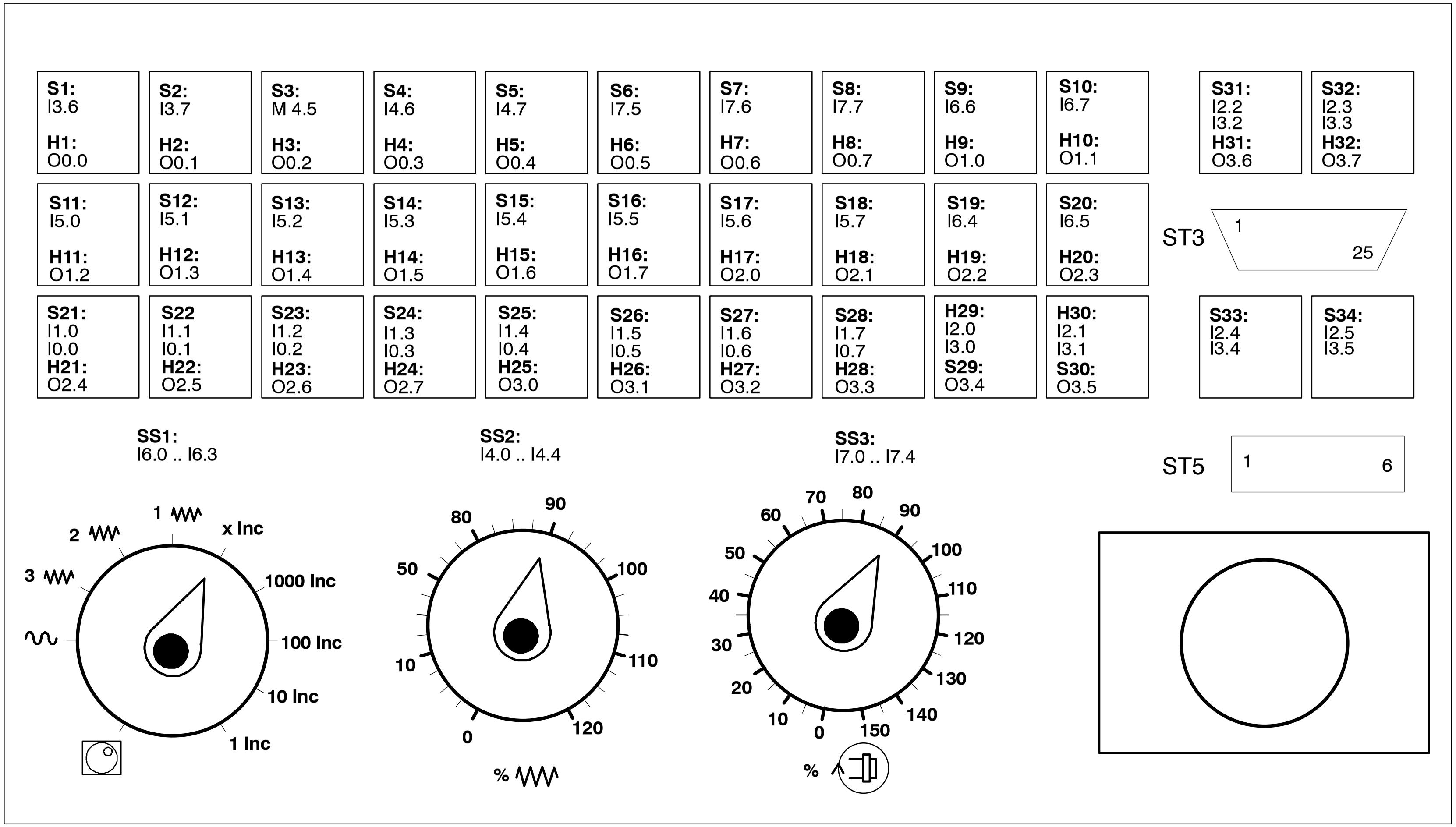
The addresses of the I/Os are relative to the start of the data area

Assignment ST3:

Pin 2	I8.0
Pin 3	I8.2
Pin 6	I9.1
Pin 7	I9.3
Pin 8	I9.5
Pin 9	I9.7
Pin15	I8.1
Pin18	I9.0
Pin19	I9.2
Pin20	I9.4
Pin21	I9.6

Assignment ST5:

Pin 2	I8.7
Pin 3	I8.6
Pin 4	I8.5
Pin 5	I8.4
Pin 6	I8.3



Connecting the handwheel

The machine user panel with PROFIBUS-DP can be used for the PNC-P.

Apart from the digital signals of the incremental measurement encoder, switching signals (inputs) can also be transferred to the PLC.

Switching signals are transmitted through plug connector ST3:

Pin 2	I 8.0
Pin 3	I 8.2
Pin 6	I 9.1
Pin 7	I 9.3
Pin 8	I 9.5
Pin 9	I 9.7
Pin15	I 8.1
Pin18	I 9.0
Pin19	I 9.2
Pin20	I 9.4
Pin21	I 9.6

A Appendix

A.1 Overview of Interface Signals

A.1.A Global output signals (PLC –> NC)

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
0.0	qGen_Reset	System control reset	1.0	qGen_StrokeInhibit	Stroke inhibit
0.1	qGen_EditInhibit	Edit inhibit	1.1	qGen_StrokeReserv	Stroke reservation
0.2	–	res.	1.2	qGen_StrokeRel	Stroke on
0.3	–	res.	1.3	–	res.
0.4	–	res.	1.4	–	res.
0.5	–	res.	1.5	–	res.
0.6	–	res.	1.6	–	res.
0.7	–	res.	1.7	–	res.
2.0	–	res.	3.0	–	res.
2.1	–	res.	3.1	–	res.
2.2	–	res.	3.2	–	res.
2.3	–	res.	3.3	–	res.
2.4	–	res.	3.4	–	res.
2.5	–	res.	3.5	–	res.
2.6	–	res.	3.6	–	res.
2.7	–	res.	3.7	–	res.

A.1.B Global input signals (NC –> PLC)

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
0.0	–	res.	1.0	iGen_StrokeIntend	Stroke intended
0.1	–	res.	1.1	iGen_NoStroke	Stroke is not running
0.2	–	res.	1.2	–	res.
0.3	–	res.	1.3	–	res.
0.4	–	res.	1.4	–	res.
0.5	–	res.	1.5	–	res.
0.6	–	res.	1.6	–	res.
0.7	–	res.	1.7	–	res.
2.0	–	res.	3.0	–	res.
2.1	–	res.	3.1	–	res.
2.2	–	res.	3.2	–	res.
2.3	–	res.	3.3	–	res.
2.4	–	res.	3.4	–	res.
2.5	–	res.	3.5	–	res.
2.6	–	res.	3.6	–	res.
2.7	–	res.	3.7	–	res.

A.1.C Channel–related output signals (PLC –> NC)

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
0.0	qCh_OpModeSel_00	Sel. mode, bit 0	1.0	qCh_OpModePlc	PLC mode
0.1	qCh_OpModeSel_01	Sel. mode, bit 1	1.1	qCh_Restart	Automatic restart
0.2	qCh_OpModeSel_02	Sel. mode, bit 2	1.2	qCh_NCStart	NC start
0.3	qCh_OpModeSel_03	Sel. mode, bit 3	1.3	qCh_TransferLock	Block transfer inhibit
0.4	–	res.	1.4	qCh_FeedHold	Feed hold
0.5	–	res.	1.5	qCh_FeedStop	Feed inhibit
0.6	–	res.	1.6	–	res.
0.7	–	res.	1.7	qCh_ReSelOff	Autom. reselection off
2.0	qCh_CtrlReset	Control reset	3.0	qCh_ASub1	Asynchr.-subrout. 1
2.1	–	res.	3.1	qCh_ASub2	Asynchr.-subrout. 2
2.2	qCh_CancDist	Cancel distance to go	3.2	qCh_ASub3	Asynchr.-subrout. 3
2.3	qCh_NextBlk	Switch to next block	3.3	qCh_ASub4	Asynchr.-subrout. 4
2.4	–	res.	3.4	qCh_ASub5	Asynchr.-subrout. 5
2.5	–	res.	3.5	qCh_ASub6	Asynchr.-subrout. 6
2.6	qCh_RetCont	Restarting	3.6	qCh_ASub7	Asynchr.-subrout. 7
2.7	qCh_Retract	Rapid retraction	3.7	qCh_ASub8	Asynchr.-subrout. 8
4.0	qCh_JogPlusWcs	WCS manual +	5.0	qCh_BlkSlash	Block skip
4.1	qCh_JogMinusWcs	WCS manual –	5.1	qCh_OptStop	Optional stop
4.2	–	res.	5.2	qCh_OptJump	Conditional jump
4.3	–	res.	5.3	–	res.
4.4	–	res.	5.4	qCh_RedRap	Limit rapid travel
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	–	res.
4.7	–	res.	5.7	qCh_Override100	Override 100%
6.0	qCh_Override_00	Override bit 0	7.0	qCh_Override_08	Override bit 8
6.1	qCh_Override_01	Override bit 1	7.1	qCh_Override_09	Override bit 9
6.2	qCh_Override_02	Override bit 2	7.2	qCh_Override_10	Override bit 10)
6.3	qCh_Override_03	Override bit 3	7.3	qCh_Override_11	Override bit 11
6.4	qCh_Override_04	Override bit 4	7.4	qCh_Override_12	Override bit 12
6.5	qCh_Override_05	Override bit 5	7.5	qCh_Override_13	Override bit 13
6.6	qCh_Override_06	Override bit 6	7.6	qCh_Override_14	Override bit 14
6.7	qCh_Override_07	Override bit 7	7.7	qCh_Override_15	Override bit 15
8.0	qCh_Custom1	Customer input 1	9.0	qCh_OnlCorrWcs	Online correction enable
8.1	qCh_Custom2	Customer input 2	9.1	qCh_OnlCorrWcsDir	Online correction direction
8.2	qCh_Custom3	Customer input 3	9.2	–	res.
8.3	qCh_Custom4	Customer input 4	9.3	–	res.
8.4	qCh_Custom5	Customer input 5	9.4	–	res.
8.5	qCh_Custom6	Customer input 6	9.5	–	res.
8.6	qCh_Custom7	Customer input 7	9.6	qCh_CoordCoupleOff	Terminate coupling
8.7	qCh_Custom8	Customer input 8	9.7	qCh_TangTRotRel	G131 release

Appendix

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
10.0	–	res.	11.0	–	res.
10.1	–	res.	11.1	–	res.
10.2	–	res.	11.2	–	res.
10.3	–	res.	11.3	–	res.
10.4	–	res.	11.4	–	res.
10.5	–	res.	11.5	–	res.
10.6	–	res.	11.6	–	res.
10.7	–	res.	11.7	–	res.
12.0	–	res.	13.0	–	res.
12.1	–	res.	13.1	–	res.
12.2	–	res.	13.2	–	res.
12.3	–	res.	13.3	–	res.
12.4	–	res.	13.4	–	res.
12.5	–	res.	13.5	–	res.
12.6	–	res.	13.6	–	res.
12.7	–	res.	13.7	–	res.

A.1.D Channel–related input signals (NC –> PLC)

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
0.0	iCh_OpMode_00	Sel. mode, bit 0	1.0	iCh_DryRun	Test mode
0.1	iCh_OpMode_01	Sel. mode, bit 1	1.1	iCh_NCReady	NC ready
0.2	iCh_OpMode_02	Sel. mode, bit 2	1.2	iCh_ProgRun	Program running
0.3	iCh_OpMode_03	Sel. mode, bit 3	1.3	iCh_TransferLockAct	Block transfer inhibit active
0.4		res.	1.4	iCh_FeedHoldAct	Feed hold active
0.5		res.	1.5	–	res.
0.6		res.	1.6	iCh_ProgStopM0	Program stop M0
0.7		res.	1.7	iCh_ProgStopM30	Program end M30
2.0	iCh_Reset	Channel reset	3.0	iCh_ASub1	Asynchr.-subrout. 1
2.1	–	res.	3.1	iCh_ASub2	Asynchr.-subrout. 2
2.2	–	res.	3.2	iCh_ASub3	Asynchr.-subrout. 3
2.3	–	res.	3.3	iCh_ASub4	Asynchr.-subrout. 4
2.4	iCh_RemoveFinish	Remove finished	3.4	iCh_ASub5	Asynchr.-subrout. 5
2.5	iCh_ReadyReEnter	Ready for reentry	3.5	iCh_ASub6	Asynchr.-subrout. 6
2.6	iCh_ReEnterAct	Re-entry active	3.6	iCh_ASub7	Asynchr.-subrout. 7
2.7	–	res.	3.7	iCh_ASub8	Asynchr.-subrout. 8
4.0	iCh_State_00	Channel status bit 0	5.0	iCh_BlkSlash	Activate block skip
4.1	iCh_State_01	Channel status bit 1	5.1	iCh_OptStop	Optional stop activated
4.2	iCh_State_02	Channel status bit 2	5.2	–	res.
4.3	iCh_State_03	Channel status bit 3	5.3	–	res.
4.4	iCh_State_04	Channel status bit 4	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	iCh_Override0	Override 0%
4.7	–	res.	5.7	iCh_Override100	Override 100%

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
6.0	iCh_Cpl01	CPL customer output 1	7.0	iCh_Cpl09	CPL customer output 9
6.1	iCh_Cpl02	CPL customer output 2	7.1	iCh_Cpl10	CPL customer output 10
6.2	iCh_Cpl03	CPL customer output 3	7.2	iCh_Cpl11	CPL customer output 11
6.3	iCh_Cpl04	CPL customer output 4	7.3	iCh_Cpl12	CPL customer output 12
6.4	iCh_Cpl05	CPL customer output 5	7.4	iCh_Cpl13	CPL customer output 13
6.5	iCh_Cpl06	CPL customer output 6	7.5	iCh_Cpl14	CPL customer output 14
6.6	iCh_Cpl07	CPL customer output 7	7.6	iCh_Cpl15	CPL customer output 15
6.7	iCh_Cpl08	CPL customer output 8	7.7	iCh_Cpl16	CPL customer output 16
8.0	iCh_Custom1	Customer output 1	9.0	iCh_G0Act	Rapid traverse active
8.1	iCh_Custom2	Customer output 2	9.1	iCh_InPosAct	In-pos range 2 active
8.2	iCh_Custom3	Customer output 3	9.2	iCh_G41G141Act	G41/G141 active
8.3	iCh_Custom4	Customer output 4	9.3	iCh_G42G142Act	G42/G142 active
8.4	iCh_Custom5	Customer output 5	9.4	–	res.
8.5	iCh_Custom6	Customer output 6	9.5	–	res.
8.6	iCh_Custom7	Customer output 7	9.6	iCh_CoordCoupleAct	Coord. coupling active
8.7	iCh_Custom8	Customer output 8	9.7	iCh_TangTRotCmd	Tool turn (TangTool)
10.0	iCh_ActFunc01	G70 active	11.0	iCh_ActFunc09	Tool compensation active bit 0
10.1	iCh_ActFunc02	Feed 100% active	11.1	iCh_ActFunc10	Tool compensation active bit 1
10.2	iCh_ActFunc03	not assigned	11.2	iCh_ActFunc11	Tool compensation active bit 2
10.3	iCh_ActFunc04	not assigned	11.3	iCh_ActFunc12	Tool compensation active bit 3
10.4	iCh_ActFunc05	G92 active	11.4	iCh_ActFunc13	Tool compensation active bit 4
10.5	iCh_ActFunc06	Thread cycle active	11.5	iCh_ActFunc14	Tool compensation active bit 5
10.6	iCh_ActFunc07	Tapping without compensating chuck active	11.6	iCh_ActFunc15	not assigned
10.7	iCh_ActFunc08	Thread cutting active	11.7	iCh_ActFunc16	not assigned
12.0	iCh_ActFunc17	not assigned	13.0	–	res.
12.1	iCh_ActFunc18	G96 active	13.1	–	res.
12.2	iCh_ActFunc19	not assigned	13.2	–	res.
12.3	iCh_ActFunc20	not assigned	13.3	–	res.
12.4	iCh_ActFunc21	not assigned	13.4	–	res.
12.5	iCh_ActFunc22	not assigned	13.5	–	res.
12.6	iCh_ActFunc23	not assigned	13.6	–	res.
12.7	iCh_ActFunc24	not assigned	13.7	–	res.

A.1.E Axis–related output signals (PLC –> NC)

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
0.0	qAx_OpModeSel_00	Bit 0 axis operating mode	1.0	qAx_TrvLim_00	Bit 0 limit switch range
0.1	qAx_OpModeSel_01	Bit 1 axis operating mode	1.1	qAx_TrvLim_01	Bit 1 limit switch range
0.2	qAx_JogPlus	Manual +	1.2	qAx_SwLimOff	Suppress limit switches
0.3	qAx_JogMinus	Manual –	1.3	–	res.
0.4	qAx_JogInch	Inch incr. step	1.4	–	res.
0.5	qAx_JogDia	Diameter incr. step	1.5	–	res.
0.6	qAx_NextNotch	Next grid position	1.6	–	res.
0.7	qAx_Reset	Basic axis setting	1.7	qAx_FxStopRel	Cancel fixed stop
2.0	qAx_SafOpModeSel	Mode selection	3.0	qAx_HandwSel_00	Bit 0 handwheel sel.
2.1	–	res.	3.1	qAx_HandwSel_01	Bit 1 handwheel sel.
2.2	qAx_SafAgreeButton	Consent key	3.2	qAx_HandwDir	Handwheel direction of rotation
2.3	qAx_SafSwitch1	Safety sw. 1 (S1)	3.3	–	res.
2.4	–	res.	3.4	–	res.
2.5	qAx_SafCheckInputState	Safety status insp. input	3.5	–	res.
2.6	qAx_SafSignalState	Safety signal status	3.6	–	res.
2.7	–	res.	3.7	–	res.
4.0	qAx_ManFeed_00	Bit 0 manual feed	5.0	–	res.
4.1	qAx_ManFeed_01	Bit 1 manual feed	5.1	–	res.
4.2	qAx_ManFeed_02	Bit 2 manual feed	5.2	–	res.
4.3	qAx_ManFeed_03	Bit 3 manual feed	5.3	–	res.
4.4	–	res.	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	–	res.
4.7	–	res.	5.7	qAx_Override100	Axis override 100%
6.0	qAx_Override_00	Override bit 0	7.0	qAx_Override_08	Override bit 8
6.1	qAx_Override_01	Override bit 1	7.1	qAx_Override_09	Override bit 9
6.2	qAx_Override_02	Override bit 2	7.2	qAx_Override_10	Override bit 10
6.3	qAx_Override_03	Override bit 3	7.3	qAx_Override_11	Override bit 11
6.4	qAx_Override_04	Override bit 4	7.4	qAx_Override_12	Override bit 12
6.5	qAx_Override_05	Override bit 5	7.5	qAx_Override_13	Override bit 13
6.6	qAx_Override_06	Override bit 6	7.6	qAx_Override_14	Override bit 14
6.7	qAx_Override_07	Override bit 7	7.7	qAx_Override_15	Override bit 15
8.0	qAx_Custom1	Customer input 1	9.0	–	res.
8.1	qAx_Custom2	Customer input 2	9.1	–	res.
8.2	qAx_Custom3	Customer input 3	9.2	–	res.
8.3	qAx_Custom4	Customer input 4	9.3	–	res.
8.4	qAx_Custom5	Customer input 5	9.4	–	res.
8.5	qAx_Custom6	Customer input 6	9.5	–	res.
8.6	qAx_Custom7	Customer input 7	9.6	–	res.
8.7	qAx_Custom8	Customer input 8	9.7	–	res.

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
10.0	qAx_TrqErrOff	Supress standstill error	11.0	qAx_Discharge	Axis discharged
10.1	qAx_LagErrOff	Suppress coupling error	11.1	qAx_FrzIpoPos	Hold command position
10.2	qAx_MasterPos	Gantry in master position	11.2	–	res.
10.3	–	res.	11.3	qAx_TrqLim	Torque reduction
10.4	–	res.	11.4	–	res.
10.5	–	res.	11.5	–	res.
10.6	–	res.	11.6	qAx_DrvOn	Drive on
10.7	–	res.	11.7	qAx_DrvLock	Feed inhibit

A.1.F Axis–related input signals (NC –> PLC)

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
0.0	iAx_RefKnow	Reference point known	1.0	iAx_DistCtrl	Axis near endpoint
0.1	iAx_RefReached	Reference point reached	1.1	–	res.
0.2	iAx_TrvCmd	Travel command	1.2	–	res.
0.3	iAx_TrvDirNeg	Negative travel direction	1.3	–	res.
0.4	iAx_Run	Axis running	1.4	–	res.
0.5	iAx_InPos	Axis in position	1.5	–	res.
0.6	iAx_NotchPos	Axis in grid position	1.6	iAx_FxStopReached	Fixed stop reached
0.7	iAx_Reset	Axis homed	1.7	iAx_FxStopAct	Fixed stop active
2.0	iAx_SafOpMode_00	Bit 0 safety mode	3.0	iAx_PosSwitch1	Position switch point 1
2.1	iAx_SafOpMode_01	Bit 1 safety mode	3.1	iAx_PosSwitch2	Position switch point 2
2.2	iAx_SafOpMode_02	Bit 2 safety mode	3.2	iAx_PosSwitch3	Position switch point 3
2.3	iAx_SafOpMode_03	Bit ^{3} safety mode	3.3	iAx_PosSwitch4	Position switch point 4
2.4	iAx_SafStatePos	Safe position status	3.4	iAx_PosSwitch5	Position switch point 5
2.5	iAx_ SafCtrlOutputState	Control status output	3.5	iAx_PosSwitch6	Position switch point 6
2.6	–	res.	3.6	iAx_PosSwitch7	Position switch point 7
2.7	–	res.	3.7	iAx_PosSwitch8	Position switch point 8
4.0	iAx_ChIndex_00	Channel number bit 0	5.0	–	res.
4.1	iAx_ChIndex_01	Channel number bit 1	5.1	–	res.
4.2	iAx_ChIndex_02	Channel number bit 2	5.2	–	res.
4.3	iAx_ChIndex_03	Channel number bit 3	5.3	–	res.
4.4	–	res.	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	iAx_Override0	Axis override 0 %
4.7	–	res.	5.7	iAx_Override100	Axis override 100%
6.0	iAx_ScsState00	SCS signal status 0	7.0	iAx_ScsState08	SCS signal status 8
6.1	iAx_ScsState01	SCS signal status 1	7.1	iAx_ScsState09	SCS signal status 9
6.2	iAx_ScsState02	SCS signal status 2	7.2	iAx_ScsState10	SCS signal status 10
6.3	iAx_ScsState03	SCS signal status 3	7.3	iAx_ScsState11	SCS signal status 11
6.4	iAx_ScsState04	SCS signal status 4	7.4	iAx_ScsState12	SCS signal status 12
6.5	iAx_ScsState05	SCS signal status 5	7.5	iAx_ScsState13	SCS signal status 13
6.6	iAx_ScsState06	SCS signal status 6	7.6	iAx_ScsState14	SCS signal status 14
6.7	iAx_ScsState07	SCS signal status 7	7.7	iAx_ScsState15	SCS signal status 15

Appendix

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
8.0	iAx_Custom1	Customer output 1	9.0	iAx_MasterAxIndex_00	Index of master axis bit 0
8.1	iAx_Custom2	Customer output 2	9.1	iAx_MasterAxIndex_01	Index of master axis bit 1
8.2	iAx_Custom3	Customer output 3	9.2	iAx_MasterAxIndex_02	Index of master axis bit 2
8.3	iAx_Custom4	Customer output 4	9.3	iAx_MasterAxIndex_03	Index of master axis bit 3
8.4	iAx_Custom5	Customer output 5	9.4	iAx_MasterAxIndex_04	Index of master axis bit 4
8.5	iAx_Custom6	Customer output 6	9.5	–	res.
8.6	iAx_Custom7	Customer output 7	9.6	–	res.
8.7	iAx_Custom8	Customer output 8	9.7	–	res.
10.0	iAx_TrqExceed	Standstill error	11.0	iAx_DrvErrClass1	Error diagn. class 1
10.1	iAx_CoupleLag	Tracking error	11.1	iAx_DrvChangeClass2	Change diagn. class 2
10.2	–	res.	11.2	iAx_DrvChangeClass3	Change diagn. class 3
10.3	–	res.	11.3	iAx_TrqLim	Torque reduced
10.4	–	res.	11.4	iAx_DryRun	Test mode
10.5	–	res.	11.5	iAx_DrvPower	Enabled for power activation
10.6	–	res.	11.6	iAx_DrvReady	Drive ready
10.7	–	res.	11.7	iAx_DrvAct	Drive under control

A.1.G Spindle–related output signals (PLC –> NC)

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
0.0	qSp_CAxOn	C axis on	1.0	qSp_TurnCW	Spindle M3 manual
0.1	qSp_CAxOff	C axis off	1.1	qSp_TurnCCW	Spindle M4 manual
0.2	qSp_JogPlus	Spindle jog M3	1.2	qSp_Stop	Spindle M5 manual
0.3	qSp_JogMinus	Spindle jog M4	1.3	qSp_Orientate	Spindle M19 manual
0.4	–	res.	1.4	–	res.
0.5	–	res.	1.5	–	res.
0.6	–	res.	1.6	–	res.
0.7	qSp_Reset	Spindle control reset	1.7	–	res.
2.0	qSp_SafOpModeSel	Mode selection	3.0	qSp_Gear1Act	GTS 1 acknowledgement
2.1	–	res.	3.1	qSp_Gear2Act	GTS 2 acknowledgement
2.2	qSp_SafAgreeButton	Consent key	3.2	qSp_Gear3Act	GTS 3 acknowledgement
2.3	qSp_SafSwitch1	Safety sw. 1 (S1)	3.3	qSp_Gear4Act	GTS 4 acknowledgement
2.4	–	res.	3.4	–	res.
2.5	qSp_ SafCheckInputState	Safety status insp. input	3.5	–	res.
2.6	qSp_SafSignalState	Safety signal status	3.6	–	res.
2.7	–	res.	3.7	qSp_GearIdleAct	Idling acknowledgement
4.0	qSp_ManSpeed_00	Spindle speed jog bit 0	5.0	–	res.
4.1	qSp_ManSpeed_01	Spindle speed jog bit 1	5.1	–	res.
4.2	qSp_ManSpeed_02	Spindle speed jog bit 2	5.2	–	res.
4.3	–	res.	5.3	–	res.
4.4	–	res.	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	–	res.
4.7	–	res.	5.7	qSp_Override100	Spindle override 100%
6.0	qSp_Override_00	Override bit 0	7.0	qSp_Override_08	Override bit 8
6.1	qSp_Override_01	Override bit 1	7.1	qSp_Override_09	Override bit 9
6.2	qSp_Override_02	Override bit 2	7.2	qSp_Override_10	Override bit 10
6.3	qSp_Override_03	Override bit 3	7.3	qSp_Override_11	Override bit 11
6.4	qSp_Override_04	Override bit 4	7.4	qSp_Override_12	Override bit 12
6.5	qSp_Override_05	Override bit 5	7.5	qSp_Override_13	Override bit 13
6.6	qSp_Override_06	Override bit 6	7.6	qSp_Override_14	Override bit 14
6.7	qSp_Override_07	Override bit 7	7.7	qSp_Override_15	Override bit 15
8.0	qSp_Custom1	Customer input 1	9.0	–	res.
8.1	qSp_Custom2	Customer input 2	9.1	–	res.
8.2	qSp_Custom3	Customer input 3	9.2	–	res.
8.3	qSp_Custom4	Customer input 4	9.3	–	res.
8.4	qSp_Custom5	Customer input 5	9.4	–	res.
8.5	qSp_Custom6	Customer input 6	9.5	–	res.
8.6	qSp_Custom7	Customer input 7	9.6	–	res.
8.7	qSp_Custom8	Customer input 8	9.7	–	res.

Bit	Symbol. addr.	PLC output signal	Bit	Symbol. addr.	PLC output signal
10.0	–	res.	11.0	–	res.
10.1	–	res.	11.1	–	res.
10.2	–	res.	11.2	–	res.
10.3	–	res.	11.3	–	res.
10.4	–	res.	11.4	–	res.
10.5	–	res.	11.5	–	res.
10.6	–	res.	11.6	qSp_DrvOn	Drive on
10.7	–	res.	11.7	qSp_DrvLock	Spindle lock

A.1.H Spindle–related input signals (NC –> PLC)

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
0.0	iSp_CAxAct	C axis is active	1.0	iSp_ProgSpReach	Speed reached
0.1	iSp_CAxSwitch	C axis switching	1.1	iSp_SpLim	Spindle speed limited
0.2	iSp_TurnCmd	Turn command	1.2	iSp_Stop	Spindle stopped
0.3	iSp_TurnDirM4	Direction of rotation M4	1.3	iSp_OrientateFinish	Spindle orientated
0.4	–	res.	1.4	iSp_OrientateAct	Spindle orientation active
0.5	iSp_InPos	Spindle in position	1.5	–	res.
0.6	iSp_PosCtrl	Position control active	1.6	–	res.
0.7	iSp_Reset	Spindle control reset	1.7	–	res.
2.0	iSp_SafOpMode_00	Bit 0 safety mode	3.0	iSp_Gear1Sel	GTS 1 selection
2.1	iSp_SafOpMode_01	Bit 1 safety mode	3.1	iSp_Gear2Sel	GTS 2 selection
2.2	iSp_SafOpMode_02	Bit 2 safety mode	3.2	iSp_Gear3Sel	GTS 3 selection
2.3	iSp_SafOpMode_03	Bit ^{3} safety mode	3.3	iSp_Gear4Sel	GTS 4 selection
2.4	iSp_SafStatePos	Safe position status	3.4	iSp_GearChange	GTS change
2.5	iSp_ SafCtrlOutputState	Control status output	3.5	iSp_IdleSpeed	Idling speed reached
2.6	–	res.	3.6	–	res.
2.7	–	res.	3.7	iSp_GearIdleSel	Idle gear selection
4.0	–	res.	5.0	–	res.
4.1	–	res.	5.1	–	res.
4.2	–	res.	5.2	–	res.
4.3	–	res.	5.3	–	res.
4.4	–	res.	5.4	–	res.
4.5	–	res.	5.5	–	res.
4.6	–	res.	5.6	iSp_Override0	Spindle override 0 %
4.7	–	res.	5.7	iSp_Override100	Spindle override 100 %
6.0	iSp_ScsState00	SCS signal status 0	7.0	iSp_ScsState08	SCS signal status 8
6.1	iSp_ScsState01	SCS signal status 1	7.1	iSp_ScsState09	SCS signal status 9
6.2	iSp_ScsState02	SCS signal status 2	7.2	iSp_ScsState10	SCS signal status 10
6.3	iSp_ScsState03	SCS signal status 3	7.3	iSp_ScsState11	SCS signal status 11
6.4	iSp_ScsState04	SCS signal status 4	7.4	iSp_ScsState12	SCS signal status 12
6.5	iSp_ScsState05	SCS signal status 5	7.5	iSp_ScsState13	SCS signal status 13
6.6	iSp_ScsState06	SCS signal status 6	7.6	iSp_ScsState14	SCS signal status 14
6.7	iSp_ScsState07	SCS signal status 7	7.7	iSp_ScsState15	SCS signal status 15

Bit	Symbol. addr.	PLC input signal	Bit	Symbol. addr.	PLC input signal
8.0	iSp_Custom1	Customer output 1	9.0	iSp_CoupleIndex_00	No. of coupling bit 0
8.1	iSp_Custom2	Customer output 2	9.1	iSp_CoupleIndex_01	No. of coupling bit 1
8.2	iSp_Custom3	Customer output 3	9.2	iSp_CoupleIndex_02	No. of coupling bit 2
8.3	iSp_Custom4	Customer output 4	9.3	–	res.
8.4	iSp_Custom5	Customer output 5	9.4	–	res.
8.5	iSp_Custom6	Customer output 6	9.5	–	res.
8.6	iSp_Custom7	Customer output 7	9.6	–	res.
8.7	iSp_Custom8	Customer output 8	9.7	–	res.
10.0	iSp_Master	Spindle is master	11.0	iSp_DrvErrClass1	Error diagn. class 1
10.1	iSp_CoupleErr	Coupling error	11.1	iSp_DrvChangeClass2	Change diagn. class 2
10.2	iSp_Synchr1	Synchronization 1	11.2	iSp_DrvChangeClass3	Change diagn. class 3
10.3	iSp_Synchr2	Synchronization 2	11.3	–	res.
10.4	–	res.	11.4	iSp_DryRun	Test mode
10.5	–	res.	11.5	iSp_DrvPower	Enabled for power activation
10.6	–	res.	11.6	iSp_DrvReady	Drive ready
10.7	–	res.	11.7	iSp_DrvAct	Drive under control

Appendix

A.2 Index

Rexroth
Bosch Group

Bosch Rexroth AG
Electric Drives and Controls
P.O. Box 13 57
97803 Lohr, Germany
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr, Germany
Phone +49 (0)93 52-40-50 60
Fax +49 (0)93 52-40-49 41
service.svc@boschrexroth.de
www.boschrexroth.com

R911307969

Printed in Germany
DOK-PNC***-PLC*INT*V10-PR01-EN-P