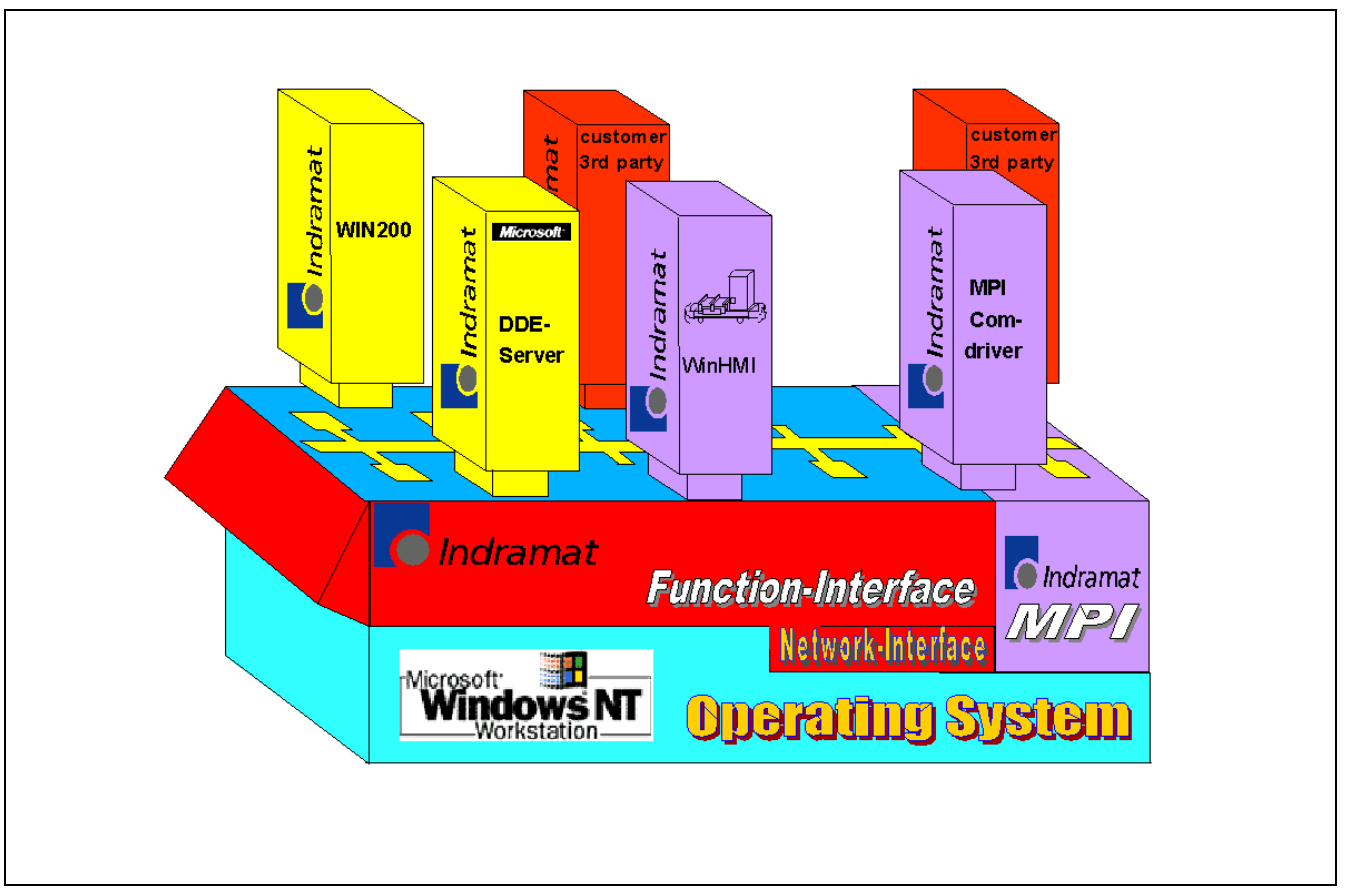
engineering
mannesmann
Rexroth



SYSTEM200
Function Interface
04VRS

Application Description

SYSTEM200

DOK-CONTRL-FUN*INT*V04-AW01-EN-P

Rexroth
Indramat

Title System 200 Function Interface 04VRS

Type of documentation Application Description

Documentation code DOK-CONTRL-FUN*INT*V04-AW01-EN-P

Internal file reference

- 120-0400-B313-01/EN

What is the purpose of this document?

This documentation serves

- to provide an overview of the functionality of the function interface
- to define application possibilities and
- to configure and develop user-defined user interfaces in in C/C++ and Visual Basic. Basic.

Course of modifications

Document identification of previous and present output	Release Date	Comments
109-0768-4187-01	01.97	New edition V16
109-0768-4187-02	03.98	Revision V03
120-0400-B313-01	09.99	Revision V04

Copyright © REXROTH INDRAMAT GmbH, 1999

Transmission as well as reproduction of this documentation, commercial use or communication of its contents will not be permitted without expressed written permission. Violation of these stipulations will require compensation. All rights reserved for the issuance of the patent or registered design. (DIN 34-1)

Validity All rights are reserved with respect to the content of this documentation and the availability of the product.

Published by REXROTH INDRAMAT GmbH
Bgm.-Dr.-Nebel-Str. 2 • D-97816 Lohr a. Main

Telephone 09352/40-0 • Tx 689421 • Fax 09352/40-4885

http://www.rexroth.com/indramat

Dept. ESM (JA)

Note This document has been printed on chlorine-free bleached paper.

Contents

0 New in Versions 04VRS 0-1
0.1 Version 04V03 0-1
0.2 Version 04V02 0-1
0.3 Version 04V01 0-2
0.4 Version 04V00 0-2
0.5 The data interface newsletter 0-3

1 General 1-1
1.1 Introduction 1-1
1.2 The function interface from the user's point of view 1-1

2 Structure and configuration examples 2-1
2.1 The structure of the function interface 2-1
Logic process 2-1
Communication process 2-2
BOF process 2-2
2.2 Configuration examples and linking possibilities 2-3
MPI link to Profibus FMS 2-3
Indramat user interface (BOF) and DDE server 2-4
Communication between a user program and Indramat devices 2-5
Communication between several user programs and Indramat devices 2-6
Function interface structure with configuration data 2-7

3 Programming 3-1
3.1 Guidelines 3-1
Software for development of user programs (PRO version) 3-2
Settings of the C++ development environment 3-4
3.2 Routines for log-on and log-off 3-6
Log-on routine „LogInIf" 3-6
Log-off routine „LogOutIf" 3-8
3.3 Data transfer and result evaluation routines 3-10
„DataTransfer" routine 3-11
„ReadGroupItem" routine 3-14
GetNumberOfGroups" routine 3-15
„GetNumberOfRows" routine 3-16
„GetNumberOfItems" routine 3-16
3.4 Routines for cyclic reading via pipes 3-17
„StartCyclicPipe" routine 3-17

„ReadCyclicPipe“ routine ... 3-18
„StopCyclicPipe“ routine ... 3-21
„SuspendCyclicPipe“ routine ... 3-22
„ResumeCyclicPipe“ routine ... 3-23
3.5 Access functions for working with SYS messages ... 3-24
„HookIfMsgList” routine ... 3-24
„GetIfMsg“ routine ... 3-25
„SetIfMsgConf“ routine ... 3-26
What SYS messages are there what must the reaction be? ... 3-27
3.6 Tips and tricks for working with the interface ... 3-30
Purging main memory with the „KILLTASK.EXE“ tool ... 3-31
Sending FI commands with the „VBDemo“ application ... 3-33
Output of diagnostic information ... 3-35
Windows NT keystroke combinations ... 3-37

4 Windows NT and function interface installation ... 4-1

4.1 The Windows NT operating system ... 4-1
Multitasking and Windows NT ... 4-1
Windows NT Task Manager ... 4-2
4.2 NT system properties settings ... 4-3
Features ... 4-3
Properties of date/time ... 4-4
Environment ... 4-5
Paging the kernel mode drivers ... 4-6
Idle activity ... 4-7
4.3 Installing the Windows NT kernel-mode driver ... 4-8
4.4 Installing the function interface ... 4-9
4.5 Directory and file structure of the function interface ... 4-18
Contents of the „INDRAMAT.INI“ file ... 4-18
Example entries in the „INDRAMT.INI“ file ... 4-19
Contents of the „IND_DEV.INI“ file ... 4-19
Example entries in the „IND_DEV.INI“ file ... 4-22
Contents of the system directory „[LW]:\Winnt\System32“ ... 4-23
Contents of the driver directory „[Drive]:\Winnt\System32\Drivers“ ... 4-23
Contents of the „C:\IND_BASE“ directory ... 4-23
Contents of the „[Drive]:\...\DOCUMENTATION“ directory ... 4-24
Contents of the „[Drive]:\...\SAMPLE\FI\VBDEMO“ directory ... 4-24
Contents of the „[Drive]:\...\SAMPLE\FI\VCDEMO“ directory ... 4-24
Contents of the „[Drive]:\...\IND_DLL“ directory ... 4-25
Contents of the „[Drive]:\...\IND_DRV“ directory ... 4-25
Example entries in the „VERSION.DAT“ file ... 4-26
Contents of the „[Drive]:\...\IND_DRV\IF_DLL“ directory ... 4-27

5 Structure of function interface commands ... 5-1

5.1 Elements of the FI command ... 5-1
Identifier ... 5-1

Selector ... 5-3
Data code ... 5-3
5.2 Data tables ... 5-4
General parameters ... 5-4
Axis meanings ... 5-5
Axis types ... 5-5

6 Function interface commands ... 6-1

6.1 FI commands for the MPCX device group ... 6-1
Output of the device configuration: CCP ... 6-1
Removing function interface jobs: DFJ ... 6-8
Information about function interface jobs: IFJ ... 6-9
Activated language of the Indramat user interface: LNG ... 6-12
Far configuration parameter: FCP ... 6-13
Far Device configuration parameter: FDC ... 6-16
Far PC configuration parameter: FPC ... 6-18
6.2 FI commands for the MTCX device group ... 6-21
Active acceleration value: AAC ... 6-21
Active angle dimension (RAD/DEG): AAD ... 6-22
Actual axis speed (spindle speed): AAS ... 6-23
Active NC block: ABI ... 6-24
Active cutting speed of the reference spindle: ACS ... 6-25
Active D-correction number: ADN ... 6-26
Active event monitoring: AEM ... 6-27
Active tool cutter number: AEN ... 6-28
Active feed rate override: AFO ... 6-29
Actual feed rate: AFR ... 6-30
Active G functions: AGF ... 6-31
Active M functions: AMF ... 6-32
Active mechanism messages: AMM ... 6-32
Active machine parameter index: API ... 6-42
Active part program message in the NC program (part program message and NC block number): APM ... 6-45
Active part program message in the NC program (NC block number only): APN ... 6-46
Actual position value of an axis: APO ... 6-46
Active NC part program number: APP ... 6-48
Actual rapid traversing override: ARO ... 6-49
Selected reference spindle: ASF ... 6-50
Actual spindle gear: ASG ... 6-51
Active system fault messages: ASM ... 6-52
Actual NC block number: ASN ... 6-59
Actual spindle override: ASO ... 6-59
Actual spindle speed: ASS ... 6-60
Active tool number: ATN ... 6-61
Active zero offset bank: AZB ... 6-62
Position setpoint of an axis: CPO ... 6-63
Control Reset: CRT ... 6-65

Device axis configuration parameter: DAC ... 6-67
Device configuration parameter: DCP ... 6-70
D-correction register: DCR ... 6-73
Identification string of NC/PLC data: DIS ... 6-76
Distance to go: DTG ... 6-77
Read device group: DTY ... 6-77
Position end value: EPO ... 6-78
Global process parameter configuration: GPC ... 6-80
Global process parameter: GPP ... 6-84
Module assignment of a process: MAP ... 6-87
Device data of the module configuration: MCD ... 6-88
Module data of the module configuration: MCM ... 6-89
Process data of the module configuration: MCP ... 6-90
SFC data of the module configuration: MCS ... 6-91
Maximum feed rate override: MFO ... 6-92
Maximum feed rate: MFR ... 6-94
Maximum rapid traversing override: MRO ... 6-95
Maximum spindle override: MSO ... 6-96
Maximum spindle speed: MSS ... 6-97
User machine data: MTD ... 6-98
Status of NC events: NEV ... 6-99
Selecting the NC memory: NMM ... 6-101
NC parameter: NPA ... 6-102
NC download commands: NPI, IPP, PPD, PPN, NPC, NPD ... 6-103
Selecting the NC program in the active NC memory: NPS ... 6-112
Next selected tool number: NTN ... 6-112
Reading and writing NC variables: NVS ... 6-114
Optimal position distance of axes: OPD ... 6-116
Programmed feed rate: PFR ... 6-118
Read an NC block: PPS ... 6-119
Programmed spindle speed: PSS ... 6-120
Read and write PLC variables: PVS ... 6-121
Servo lag of an axis: SLA ... 6-123
Sercos parameters: SPA1 ... 6-125
Change over active Sercos phase: SPH ... 6-128
Selected NC part program: SPP ... 6-129
Torque: TQE ... 6-130
Data of the zero offset table: ZOD ... 6-132
Sercos errors ... 6-137
6.3 FI commands for the MSCX device ... 6-139
Reading and writing Sercos parameters: SPA1 ... 6-139
Read and write the Sercos phase: SPH ... 6-143
Determining the current SERCANS error: ASE ... 6-144
Clearing a SERCANS error: CSE ... 6-145
Global SERCANS errors ... 6-146
Sercos errors ... 6-147

6.4 FI commands for the MISX device group ... 6-149
Active mechanism messages: AMM ... 6-149
Active system fault messages: ASM ... 6-157
Control Reset: CRT ... 6-163
Identification string of PLC data: DIS ... 6-164
Read device group: DTY ... 6-164
Module assignment of a process: MAP ... 6-165
Device data of the module configuration: MCD ... 6-166
Module data of the module configuration: MCM ... 6-167
Process data of the module configuration: MCP ... 6-168
SFC data of the module configuration: MCS ... 6-169
Read and write PLC variables: PVS ... 6-170
6.5 FI commands for the MTAX device group ... 6-173
Actual position value of an axis: APO ... 6-173
Active system fault messages: ASM ... 6-174
Read CMOS RAM integer parameter: CMI ... 6-180
Write CMOS RAM integer parameter: CMI ... 6-180
Read CMOS RAM floating point parameter: CMF ... 6-181
Write CMOS RAM floating point parameter: CMF ... 6-182
Read CMOS RAM ASCII parameter: CMA ... 6-183
Write CMOS RAM ASCII parameter: CMA ... 6-183
Control Reset: CRT ... 6-184
Device Configuration parameter: DCP ... 6-185
Identification string of NC/PLC data: DIS ... 6-188
Distance to go: DTG ... 6-188
Read device group: DTY ... 6-188
Module assignment of a process: MAP ... 6-189
Device data of the module configuration: MCD ... 6-190
Module data of the module configuration: MCM ... 6-191
Process data of the module configuration: MCP ... 6-192
SFC data of the module configuration: MCS ... 6-193
Read and write PLC variables: PVS ... 6-195

7 Error codes ... 7-1

7.1 General error result line ... 7-1
7.2 Error codes from 200 to 999 ... 7-2
7.3 Error codes from 1000 to 1999 ... 7-11
7.4 Error codes from 2000 to 2999 ... 7-12
7.5 Error codes from 4000 to 4999 ... 7-14
7.6 Error codes from 5000 to 5999 ... 7-19
7.7 Error codes from 6000 to 6999 ... 7-20
7.8 Error codes from 7000 to 7999 ... 7-21
7.9 Error codes from 8000 to 8999 ... 7-23
7.10 10000 error codes ... 7-24
7.11 100000 error codes ... 7-24
7.12 110000 error codes ... 7-25

8 Frequently asked questions: FAQ 8-1

8.1 Function interface FAQs 8-1

8.2 Windows NT FAQs 8-2

9 Glossary 9-1

10 Index 10-1

11 Kundenbetreuungsstellen – Sales & Service Facilities 11-1

0 New in Versions 04VRS

0.1 Version 04V03

General information

- This version is still under development!

Version identification, Indramat software components

Entries in the „C:\IND_BASE\INDRAMAT.INI“ file:

- IfDllMode = 04.20
- IfVersion = 04V03

Software components which contain the function interface by default:

- All Indramat System 200 user interfaces of Version 18V06.

FI commands Changes/additions

- FI commands „XYZ“ have been implemented as „XYZ1“ with newly formatted output: DCR1. Do not use the FI command „DCR“ any more!
- New - speed-optimised - FI command „GPP“ for reading the global process parameter device group (MTCX and MTRX).
- New FI command „CCP5“ for output of the configuration data of the device that is addressed via the specified device address (MPCX device group).
- New FI command „NPS“ for selecting the NC program in the active NC memory for execution (MTCX and MTRX device groups).
- New FI command „NMM“ for selecting the NC memory for NC program execution (MTCX and MTRX device groups).

0.2 Version 04V02

General information

- New chapter entitled „Frequently Asked Questions (FAQ)“.
- In Service Pack 2 of the user interface 18V05, essentially the shrinkage of resources has been remedied.

Version identification, Indramat software components

Entries in the „C:\IND_BASE\INDRAMAT.INI“ file:

- IfDllMode = 04.10
- IfVersion = 04V02

Software components which contain the function interface by default:

- All Indramat System 200 user interfaces of Version 18V05 with Service Pack 2.

FI commands Changes/additions

- Module commands MCD1, MCM1 and MCS1 enabled for MISX device group.
- Module commands MCD1, MCM1, MCP1, MCS1, MAP1 enabled for MTAX device group
- Addition of CR_APO2, CR_DTG2, CR_CMA, CW_CMA, CR_CMI, CW_CMI, CR_CMF and CW_CMF for the MTAX device group.

Basic processes Changes/additions

- Resource shrinkage in the logic process remedied.
- Expansion from 15 to max. 255 group requests in the case of cyclic requests (see „Routines for cyclic reading via pipes“).

0.3 Version 04V01

General information

- Inclusion of the PRO version as a software option in the installation program.

Version identification, Indramat software components

Entries in the „C:\IND_BASE\INDRAMAT.INI“ file:

- IfDllMode = 04.10
- IfVersion = 04V01

Software components which contain the function interface by default:

- All Indramat System 200 user interfaces of Version 18V05.

FI commands Changes/additions

- Additions to the device-independent access functions.
- New FI command „CRT“ for triggering a controller reset for the MTCX, MTRX and MISX device groups.

Basic processes Changes/additions

- Error correction of the telegram optimiser (correction of timeout detection).
- New SYS message „MSG_PC__ALIVE“ in the PC network.

0.4 Version 04V00

In comparison with the previous 03VRS versions, fundamental changes have been implemented in this version:

General information

- Delivery of a Visual Basic example link to the function interface (application including source codes).
- Delivery of the paper documentation as online help in the Windows NT/95 Help format.
- Provision of an installation program for the function interface.

Version identification, Indramat software components

Entries in the „C:\IND_BASE\INDRAMAT.INI“ file:

- IfDllMode = 04.00
- IfVersion = 04V00

Software components which contain the function interface by default:

- All Indramat System 200 user interfaces of Version 18V04 with Service Pack 2.

FI commands Changes/additions

- FI commands „XYZ“ have been implemented as „XYZ1“ with newly formatted output: AAC1, AAS1, ADN1, ,AFO1, APO1, ARO1, ASO1, AZB1, MFO1, MRO1, MSO1. Do not use the FI commands „XYZ“ any more!
- FI command replaced by the commands ASMx, AMMx.
- New functions for the BOF/GBO for calling up WIN32 applications.
- New functions for a WIN32 application at the function interface for calling up the BOF/GBO.
- Expansion of the data structures for BOF/GBO. Support of the Indramat TRANS200 device type (MTRX device group).

Note: The Indramat devices TRANS200-P and TRANS200-R (MTRX device group) are still in the development status and can therefore not yet be ordered.

- Extended function calls for device configuration.
- Message for activating or deactivating a PC in a PC network.
- Expansions of the device-independent access functions.

Basic processes Changes/additions

- Error correction of the telegram optimiser (correction of timeout detection).
- Additions in the PLC data optimiser.
- Enlarged input buffer for the telegram optimiser.
- Revision of the internal interface.
- Error correction in data provision by means of the „ReadGroupItem“ routine.
- Error remedying of the communication channel.
- Error corrections in the internal DLL interfaces.
- Correction of INDIF200.DLL (correction of the binary result in the case of spindle data).
- Changes in LogOutIf(), with regard to the selective KILLTASK
- Revision of the COMVIEW interface to WIN200.
- Shifting of the new SYS message to the „INDIF000.H“ file.

0.5 The data interface newsletter

We will inform you by email about news and updates of the Indramat products MPI and the function interface.

Send an email message with the message text **subscribe** to:

owner-ml_datainterface@proxy.indramat.de

To log out of the list, proceed in the same way, but with the message text: **unsubscribe**.

Note: Your email address is not passed on to third parties.

1 General

1.1 Introduction

The Indramat function interface is a standard data interface of Indramat devices for user programs based on the windows NT platform.

Requirements

In relation to free access to available NC and PLC data, there is a need to provide a data interface that is as open as possible, at low cost and which is easy to handle. The principal requirements consist of being able to gain access to CNC/PLC data with a large scope of functions and a high access and reaction speed.

Objective

The Indramat function interface has precisely this objective, i.e. it permits access to all necessary controller data via a compact and functional interface.
This makes it possible for customers to create their own user interfaces completely in the Visual C++ or Visual Basic language. Therefore, a powerful interface is available to users with which they can communicate with Indramat hardware and user interfaces by way of mnemonic function calls. Thus, the function interface provides a universal solution for data communication.

Validity

This description is valid for the following versions:

- BOF/GBO: 18Vxx
- Function interface: 04Vxx
- Windows NT Workstation/Server: 4.0
- Visual C++: 5.0
- Visual Basic: 5.0

1.2 The function interface from the user's point of view

The function interface is a client/server interface and, in a library, (DLL) it provides the user with services for communication. The services, i.e. the functions of the DLL, perform the communication tasks that are needed to read isolated data, to read cyclic data, to cyclically ready data groups, to write data, and for the purpose of process communication with Indramat user interfaces and hardware. The function interface can communicate with a maximum of seven mutually independent user programs (clients). For example a user program may be a customer's specific user interface, an Indramat DDE server, or a simple communication driver to another data interface. Towards the hardware end, the function interface supports up to eight parallel communication channels and, in doing so, a communication channel establishes a connection to one of a maximum of 16 data terminal devices. Here, the physical communication address can be a serial interface (RS232/RS485), a dual port RAM or a shared memory area.

2 Structure and configuration examples

2.1 The structure of the function interface

As an overall component, the function interface consists of the three basic processes:

- Logic process
- Communication process
- BOF process.

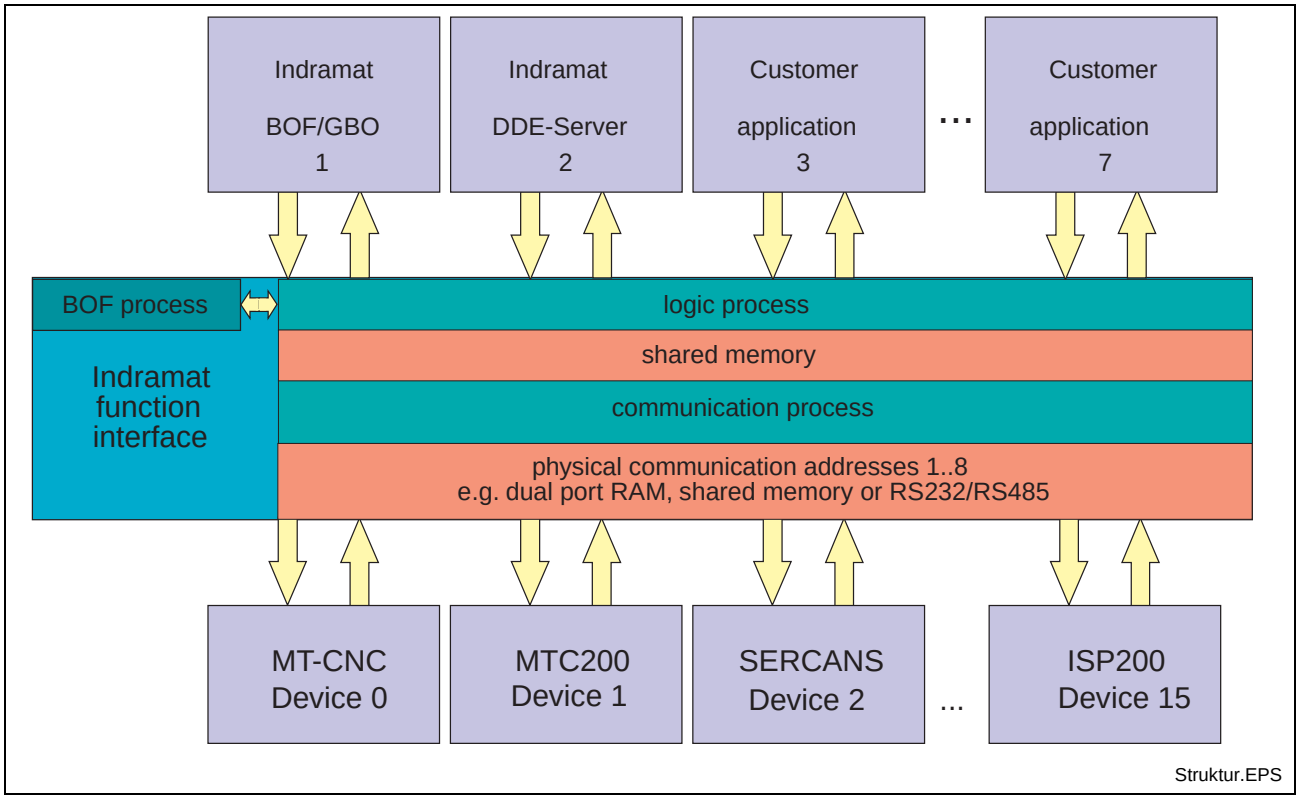


Fig. 2-1: Structural overview of the function interface

Logic process

The logic process provides the user program with the actual data interface including the services described in the previous chapter. To do this, it opens a logic channel (LOG channel) for each logged-on user program. The logic process is also a data interface to all defined devices and to the management and status device data that is managed by the BOF process. From the point of view of the user program (client), the logic process is the server. Towards the other side, the logic process establishes the link to the communication process via a shared memory. Data is distributed to the individual logic channels via this link.

Note: A maximum of seven LOG channels can be opened for function interface applications. The chapter entitled „Programming“ describes how the data interface to the logic process has to be handled and how data accesses by the user program to the function interface are possible.

Communication process

The communication process executes the requests of the various logic channels, generates communication with the devices with a time breakdown and initialises all configured devices after starting. The communication process and the logic process therefore permit data access to the respective Indramat devices (MT-CNC, MTC200-P, ISP200-P and SERCANS etc.).

On the one hand it exchanges Indramat telegrams with the logic process via the shared memory. Towards the other end, it exchanges internal Indramat telegrams with the configured device via the dual port RAM or via a serial interface. The communication process opens a communication channel (thread) for each of these configured devices. It therefore permits communication via diverse protocol variants and through several parallel interfaces simultaneously.

Note: A maximum of eight threads can be opened. Configuration of the communication addresses and the processing options of the individual Indramat devices are realised by way of the Indramat system configurator and are saved in the „IND_DEV.INI“ file (see chapter entitle „Directory and file structure of the function interface“).

Windows NT kernel-mode driver (MTCNC00I.SYS)

For access to hardware-oriented components (serial interface and dual port RAM etc.), under the windows NT operating system the kernel mode driver „MTCNC00I.SYS is needed (see the chapter entitled „Installation of the Windows NT kernel-mode driver“).

This kernel-mode driver establishes the link between the function interface and a serial interface or a dual port RAM etc. The kernel-mode driver for the function interface supports the physical interfaces RS232-C, dual port RAM for the MTC200-P, IPS200-P and RS485.

CAUTION

Via the kernel-mode driver, the function interface only supports RS485 interfaces that have been approved and tested by Indramat.

In the event of failure to observe this, all warranty claims lodged against Indramat will be null and void.

BOF process

As afar as its structure is concerned, the BOF process is an internal user program and uses the first LOG channel for communication with the logic process. It provides static and dynamic configuration data, it supplies funcitonally higher-grade functions and creates the corresponding data structure for each configured device. The BOF process thus puts together MTC200-P controller data, for example, with data of the PC's hard disk, which a user program can then access. The BOF process thus performs administrative tasks.

Note: The chapter entitled „Function interface commands“ describes how the data of Indramat devices and the hard disk of a PC can be accessed.

2.2 Configuration examples and linking possibilities

MPI link to Profibus FMS

The following figure shows linking of the MPI (Multi-Protocol Interface) with a Profibus FMS characteristic and additional user programs to the function interface.

The first LOG channel (Logic Channel 1) of the function interface is used by a user program, e.g. a customer's specific user interface. The Indramat GUI (**GUI** = **G**raphical **U**ser **I**nterface) runs in the DOS environment of Windows NT. The link to the second LOG channel (Logic Channel 2) is realised via a virtual device driver. The function interface is informed by the Indramat GUI about certain events (PLC program download or parameter download etc.) via this link. The function interface makes these events available to the user program in the form of system messages (SYS messages) (see chapter entitled „Access functions for working with SYS messages"). The MPI link to the function interface is realised via the third LOG channel (Logic Channel 3).

The illustration also shows that three parallel Windows NT processes access the data of one Indramat device (MT-CNC controller). Here, the same access privileges are basically available to each process.

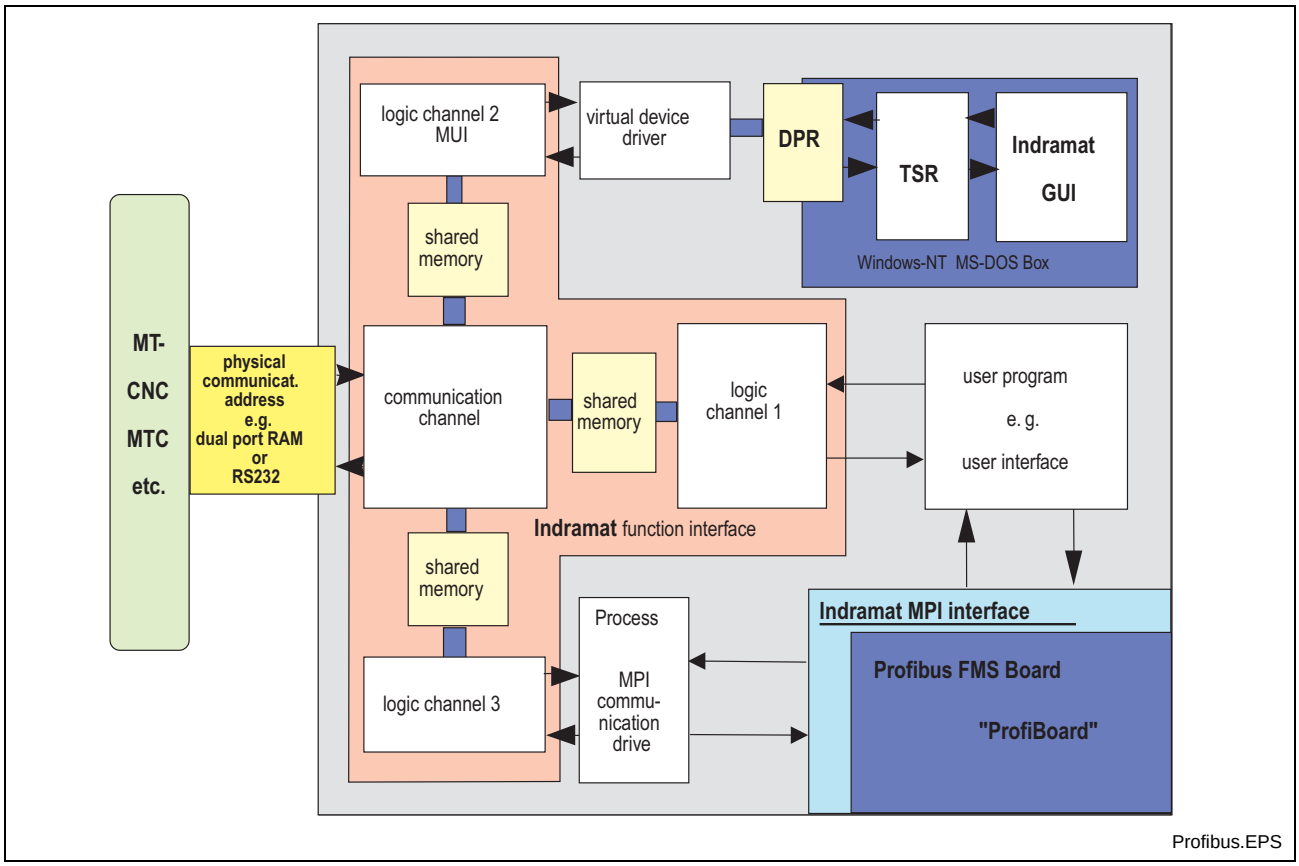


Fig. 2-2: MPI with Profibus FMS link

Indramat user interface (BOF) and DDE server

The following graphic shows the software structure with the 18VRS Indramat user interface and with use of the Indramat DDE server.

The components „WIN-MTC" and „WIN-HMI" are components of the Indramat user interface WIN200. The user interface is converted to components of WIN200 on the fly. WIN200 is the basis. The DOS interface is „menu-oriented", i.e. WIN200 calls up certain menus that are part of the user interface. It then returns to the WIN200 user interface. This results in the possibility of exchanging menus of the user interface for menus of the respective WIN variant.

Via standard communication mechanisms, the DDE server enables linking to external program packages such as WONDERWARE „InTouch". With the NetDDE option, the DDE server also permits linking via a network.

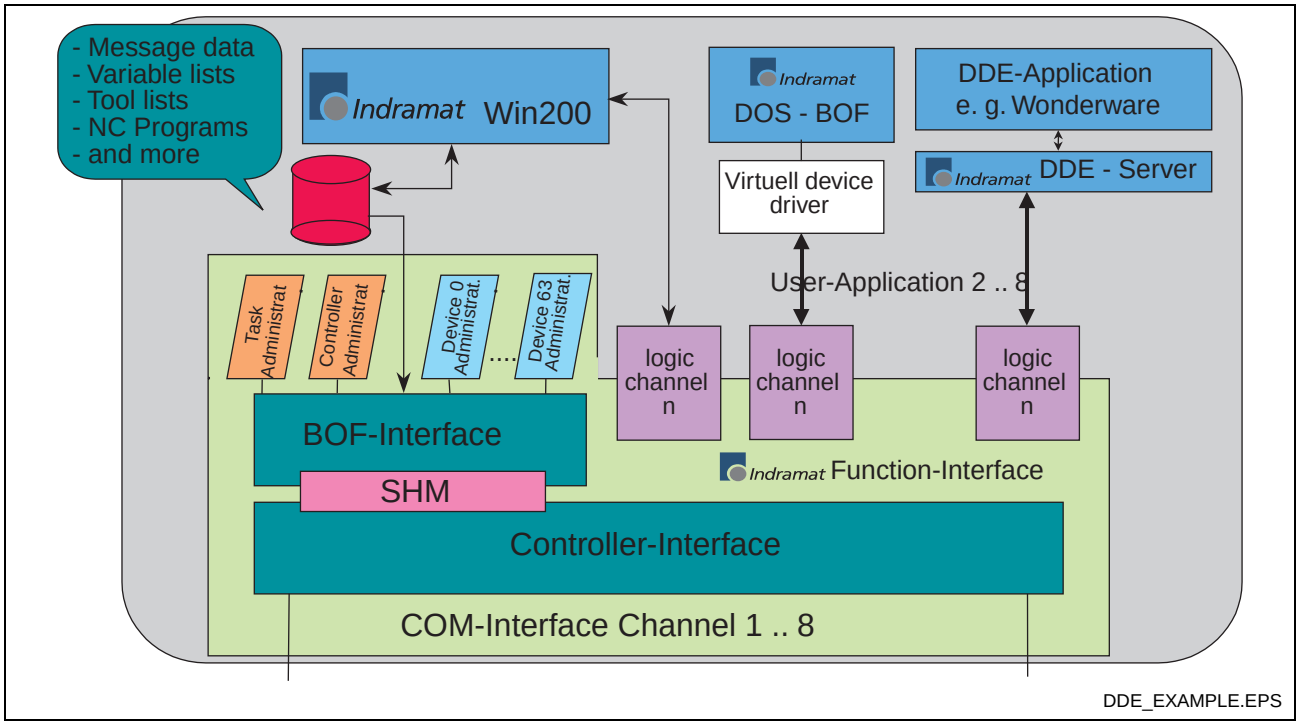


Fig. 2-3: Software structure: Indramat user interface and DDE server

Communication between a user program and Indramat devices

The following graphic shows communication of a user program on an Indramat PC (BTV30) with two Indramat devices (MTC200-R and MTC200-P). In this case, on the one end the device 00 (MTC200-R) communicates via the serial interface (COM1), while the device 01 (MTC200-P) communicates via a dual port RAM with the communication process. The communication process opens one thread for each parameterised communication channel. The user program shown can access the data of both devices. To this end, the corresponding device address is specified in the function interface command (FI command) (see chapter entitled „Structure of the function interface commands“).

Note: Combination of several cyclic requests (FI commands) on both devices is easily possible (see chapter entitled „Data transfer and result evaluation routines“).

The configuration data of the Indramat devices is compared against the actual status during the initialisation phase of the function interface. Requested FI commands are thus checked in relation to their validity for the configured device group. Any errors in the command mnemonic can therefore already be intercepted at the highest level.

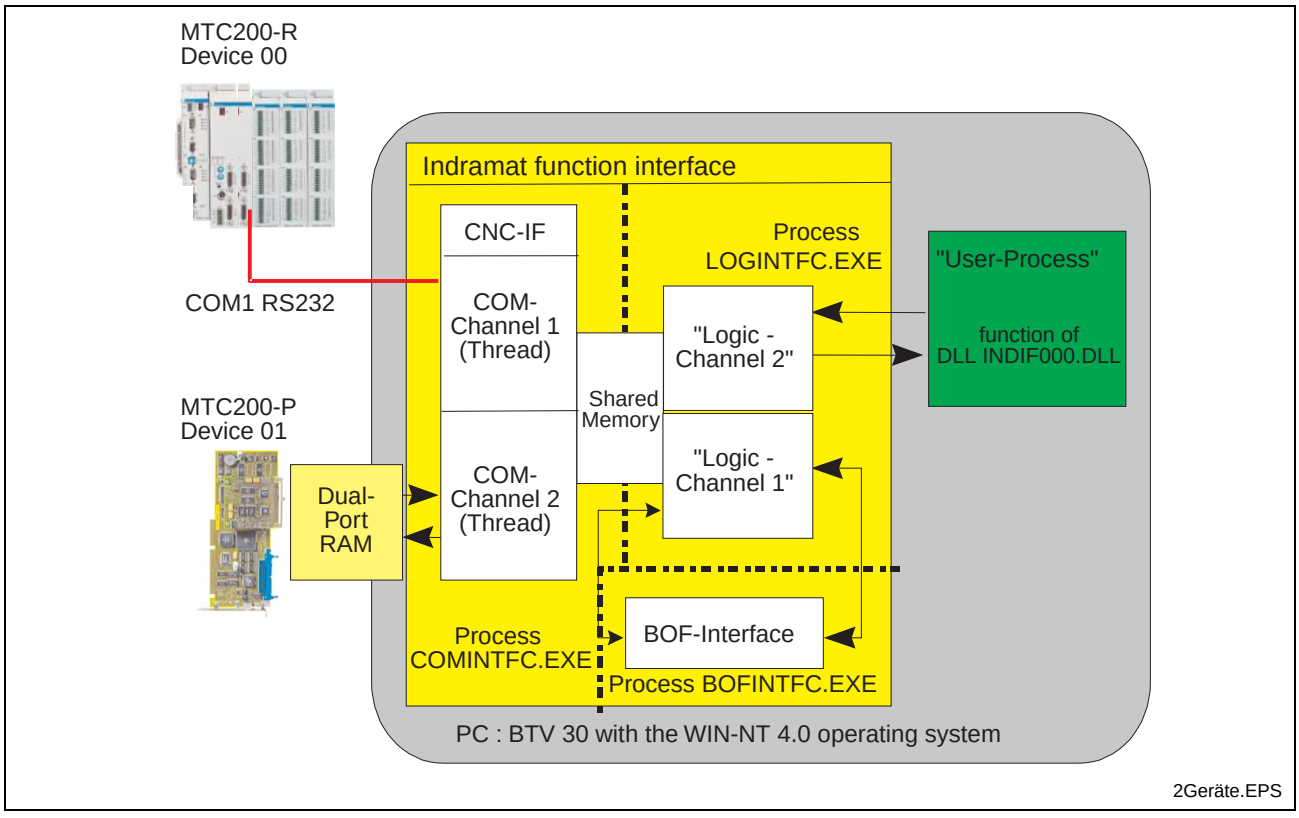


Fig. 2-4: Communication by a user program with Indramat devices

Communication between several user programs and Indramat devices

The following figure shows the software structure of the function interface in relation to communication with several devices during operation of several user programs.

Note: Combination of the decentralised MTC200-R with the integrated MTC200-P is a practicable configuration for a revolving machine, for example.

Here, the function interface enables parallel communication via different interfaces. Towards the user program end, four programs are logged on with the function interface in the following example. Each user program can communicated with each device independently of the other user program. During the operation of several devices and several user programs, the function interface operates like a two -stage, decoupled multiplexer. The communication process contains a multiplexer towards the hardware end and the logic process a multiplexer to the user programs.

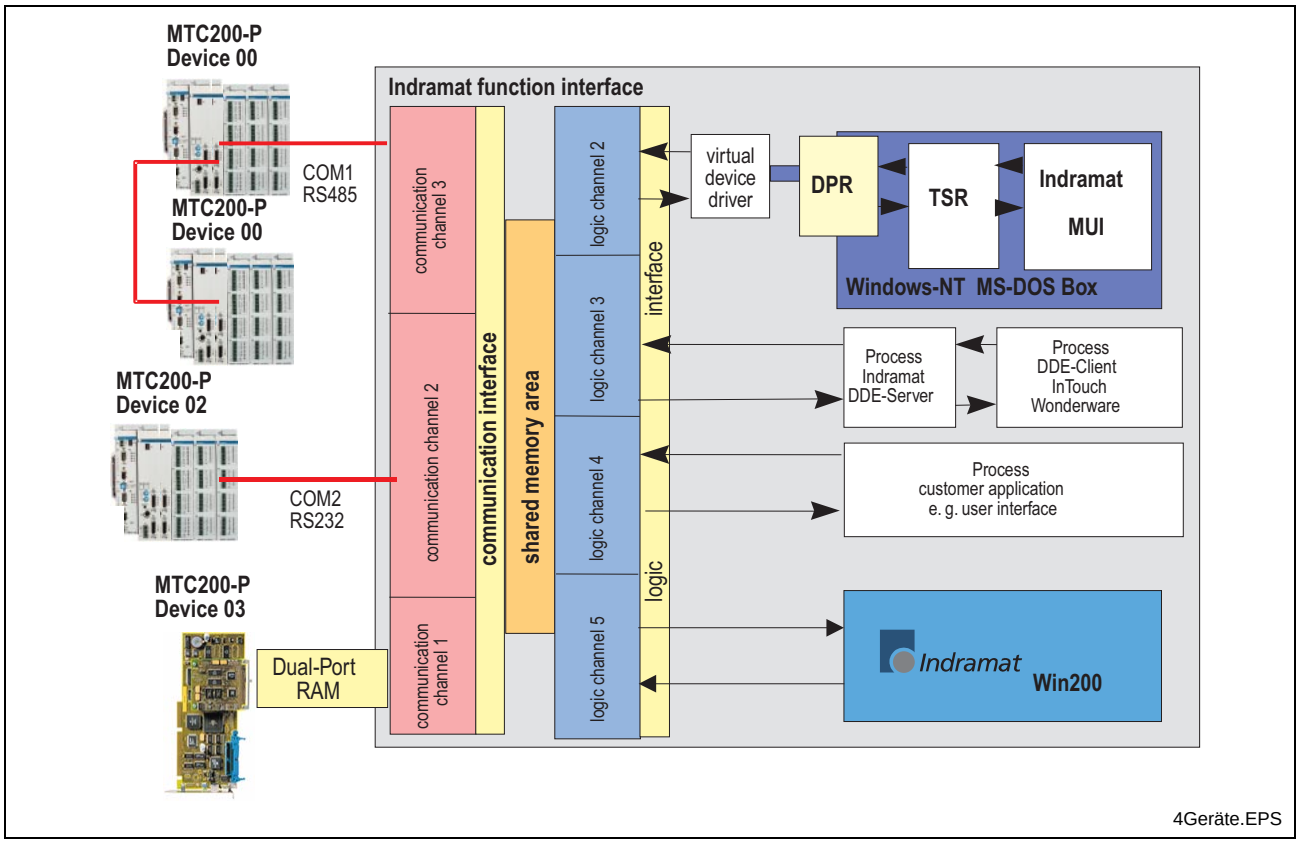


Fig. 2-5: Communication of several user programs with Indramat devices

Function interface structure with configuration data

The following graphic shows the functional relationship between the configuration data and the created data structures in the function interface.

The necessary communication interfaces and the existing device types are configured via the Indramat system configurator. The created configuration is stored in the „IND_DEV.INI“ file (see chapter entitled „Directory and file structure of the function interface“).

The device parameters are set via the user interface, thus defining the existing CNC processes and the number and features of the NC axes, for example.

The following device data is checked during the initialisation phase of the function interface:

- Function interface version and GBO version
- Configuration settings in the „IND_DEV.INI“ file
- Validity of the PLC program
- Validity of the NC parameter set and
- Initialisation of the machine keys

For the most frequently required data, the BOF process creates several data memories. The created data memories have the following meaning and contain the following information:

Common-MAP Log-on name of the application program and lifesign of the application.

System-MAP Complete device configuration. Current language setting of the Indramat user interface. Number of configured devices and the interface parameters of the existing devices.

Data-MAP x Device designation. Index identifier of the PLC program and of the parameter set (MTC). Process and axis configuration (MTC). Further frequently used configuration information. The function interface creates one Data-MAP for each configured device. The structure of each respective Data-MAP depends on the detected device type.

The Data-MAPs are updated automatically by the function interface after a parameter or a PLC program download, for example. To enable an user program to react to such a change, the function interface provides system messages (SYS messages) (see chapter entitled „Access functions for working with SYS messages“). An application logged on with the function interface is informed about a parameter download, for example, via these SYS messages.

Note: The modified parameter configuration may result in synchronisation of the user program to the change.

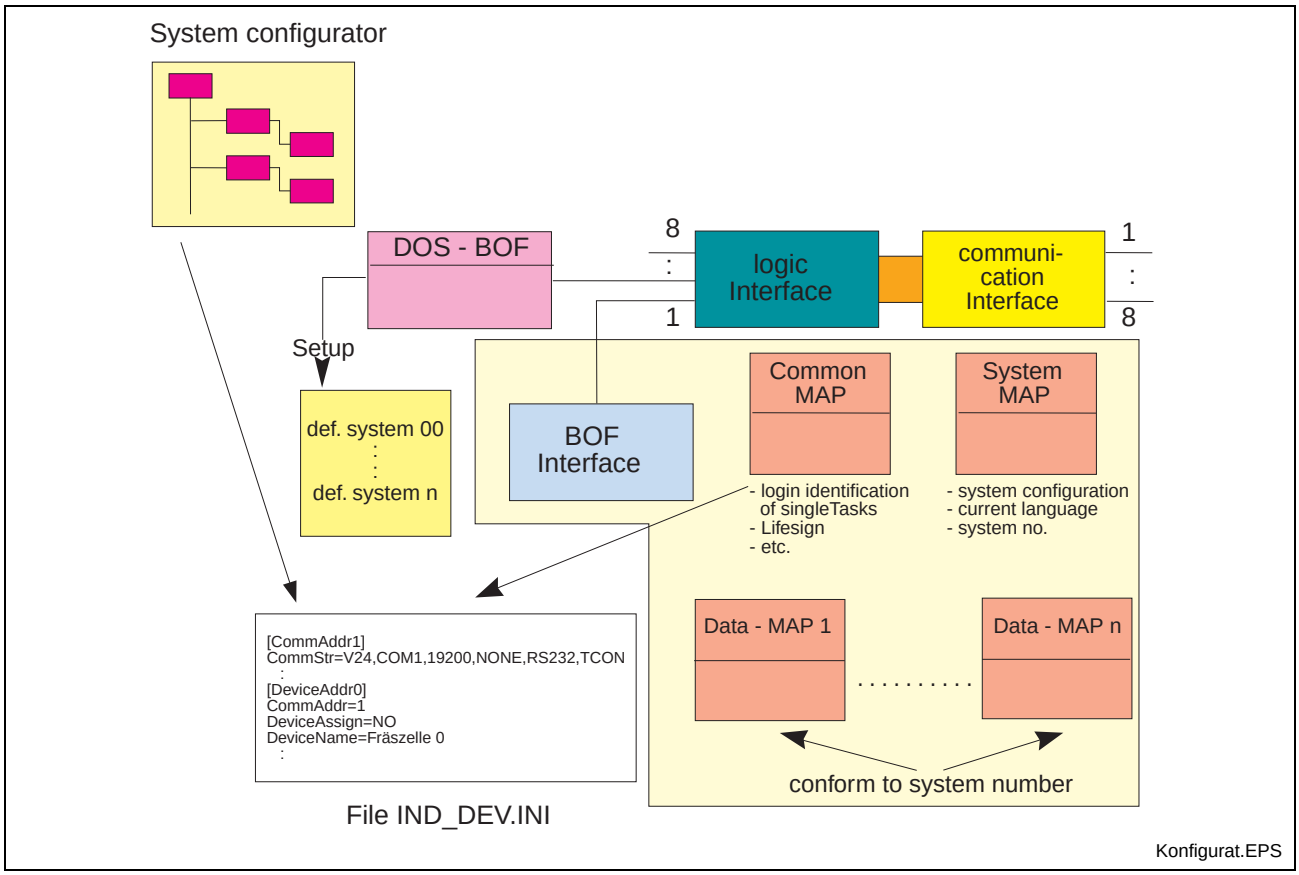


Fig. 2-6: Structure of the function interface with the configuration data

3 Programming

3.1 Guidelines

Every application software that wishes to access the function interface must be created in one of the following programming languages:

- Visual C/C++ (32-bit version), or
- Visual-Basic Version 5.0 and higher.

The following must be observed during the course of programming:

- The available computer capacity must not be exceeded. Depending on the requirement, you must check whether a more powerful PC may have to be used.

Note: As Windows NT processes, parts of the Indramat function interface require the highest priority.

- Absolute path entries in the application must be avoided because later changes to the drive (e.g.: from C:\ to D:\) or the directory structure will not be supported.

Note: The system directory and the drive of Windows NT can also be chosen freely.

The following conditions and states of the controllers or devices must be taken into account when programming:

- During a PLC program or parameter download from the Indramat user interface, other applications must not read or write to controller data. The system messages (SYS-MSGs) of the call interface are available for evaluation of this state. The system messages for the PLC program and parameter download must be taken into account in the logic of the user program.
- Reading or writing of PLC data is limited. PLC variables with a maximum length of 240 bytes can be read or written with the FI command PVS (see chapter entitled „Function interface commands“). PLC structures and arrays can have a dynamic length. Here, extremely precise planning is necessary for communication with the PLC.
- Basically any chosen PLC variable can be written via the function interface. In the application, however, only PLC variables should be written that are taken into account accordingly in the PLC program.

Note: Write access to PLC variables that are not declared must be avoided.

- Signals of the process/axis interface should never be modified directly by the application. Use a write/read buffer in the PLC.

CAUTION

When using the function interface, safe and error-free operation of the controller system is only guaranteed when the aforementioned guidelines are observed.

If the guidelines are not observed, all warranty claims against Indramat will be null and void.

Software for development of user programs (PRO version)

The installation diskettes or CD-ROM that bear the designations

- SWD-CONTRL-FUN-04VRS-MS-C1,44 (diskette version)
- SWD-CONTRL-FUN-04VRS-MS-CD600 (CD-ROM version)

contain not only the run time version of the function interface, but also the necessary configuration software for the development of user programs in the programming languages Visual C/C++ and Visual Basic.

When the software option:

- SWS-FUN*PC-PRO-04VRS-MS (PRO version)

is released, the relevant source files are installed. The functions that can be used in the user program are published in these files (see chapter entitled „Installation of the function interface „).

Note: The Visual Basic examples for the individual function interface routines are also supplied with the PRO version. At the time of writing, the creation of an example program had unfortunately not been completed in good time. Please accept our apologies. It goes without saying that we will include the example program „VCDemo“ with the next function interface version.

Visual C/C++

The following C sources (*.C) and header files (*.H) must be included in the user program as detailed in the chapter entitled „Settings of the C++ development environment“:

File	Purpose
INDIF000.C	Load procedure for the functions of the DLL.
INDRAMAT.C	Load procedure for the global functions of the DLL.
INDIF000.H	Function definitions of the individual routines.
INDIFX00.H	Function definitions of the individual routines.
INDRAMAT.H	Global function interface routines

Note: After the default installation procedure, the source files are located in the „C:\MT-CNC\Sample\FI\VCDemo\“ directory (see chapter entitled „Installation of the function interface”).

Visual Basic

No settings need to be made in the development environment of Visual Basic 5.0. Only the following source file has to be included in the user program:

File	Purpose
INDIF000.BAS	Load procedure for the functions of the DLL; function definitions of the individual routines;

Note: After the default installation procedure, this source file is located in the „C:\MT-CNC\Sample\FI\VBDemo\“ directory (see chapter entitled „Installation of the function interface“).

By way of the Project menu item:

- Add Module

the source file „INDIF000.BAS“ is incorporated into the user program as a new module.

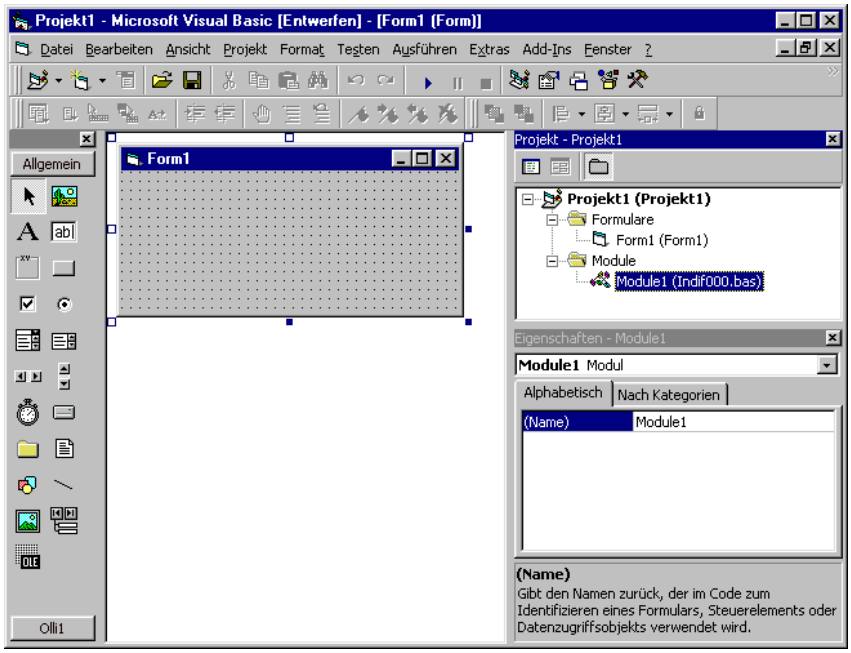


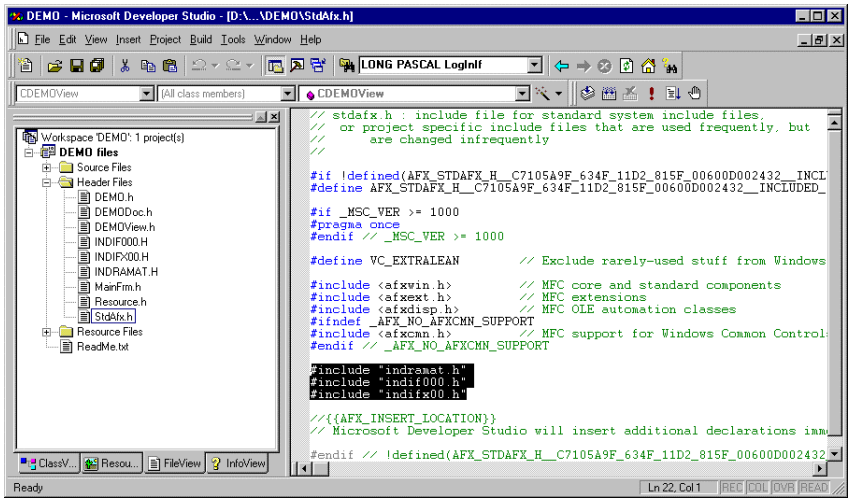
Fig. 3-1: Including the file „INDIF000.BAS” in the Visual Basic user project

Settings of the C++ development environment

To make sure that the functions of the library „INDIF000.DLL” (see chapter entitled „Contents of the directory [drive]:\...\IND_DLL\“) of the function interface are globally available, the following header files:

- INDIF000.H,
- INDIFX00.H and
- INDRAMAT.H

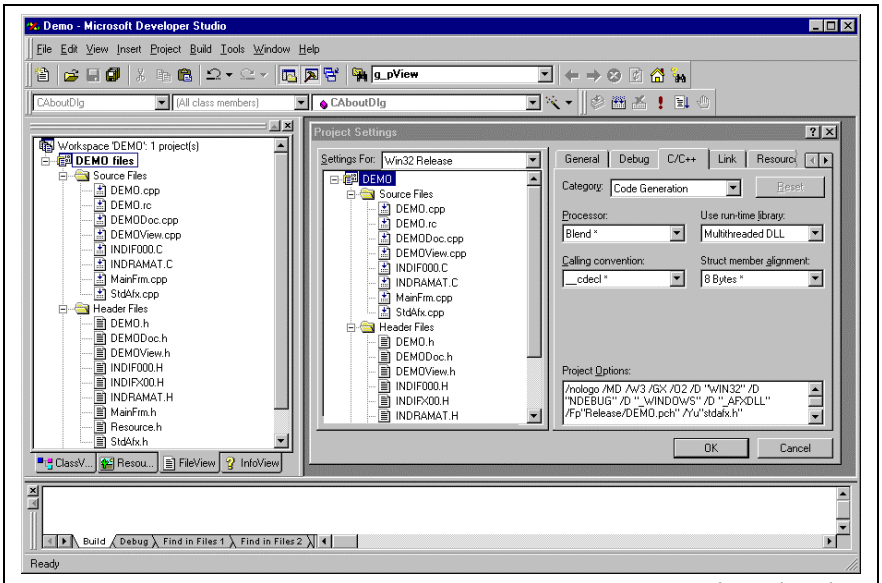
must be incorporated into the user program with the „#include“ syntax.



StdAfx.bmp

Fig. 3-2: Including the Indramat header files in the user program

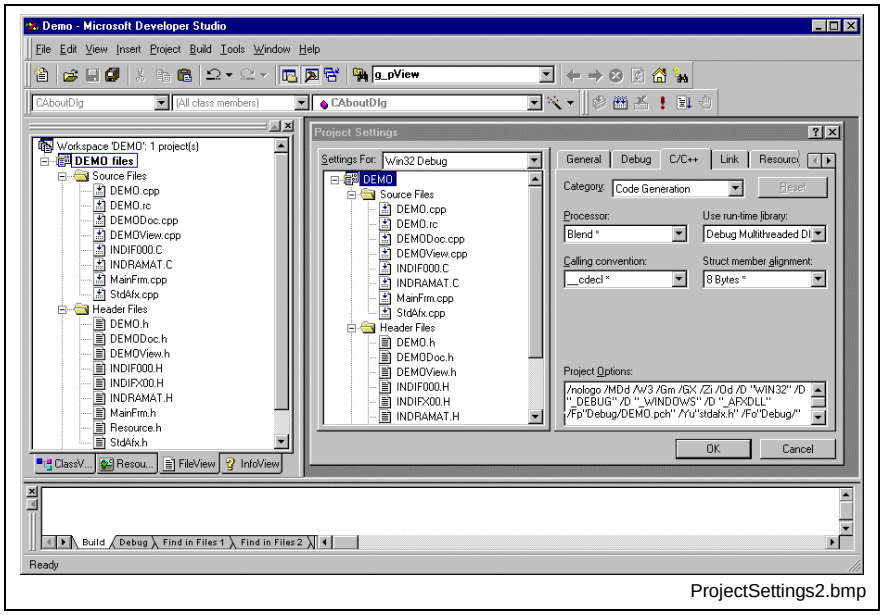
For Visual C++ 5.0, the entry „Multithreaded DLL“ must be chosen in the project settings: „For Win32 Release“ on the tab entitled „C/C++“ under the category „Code-Generation“ in the combo box entitled „Use run-time library“.



ProjectSettings1.bmp

Fig. 3-3: Project settings „For Win32 Release“: Multithreaded DLL

In the project settings: „For Win32 Debug“, select the entry „Debug Multithreaded DLL“ on the tab entitled „C/C++“ under the category „Code-Generation“ in the combo box entitled „Use run-time library“.

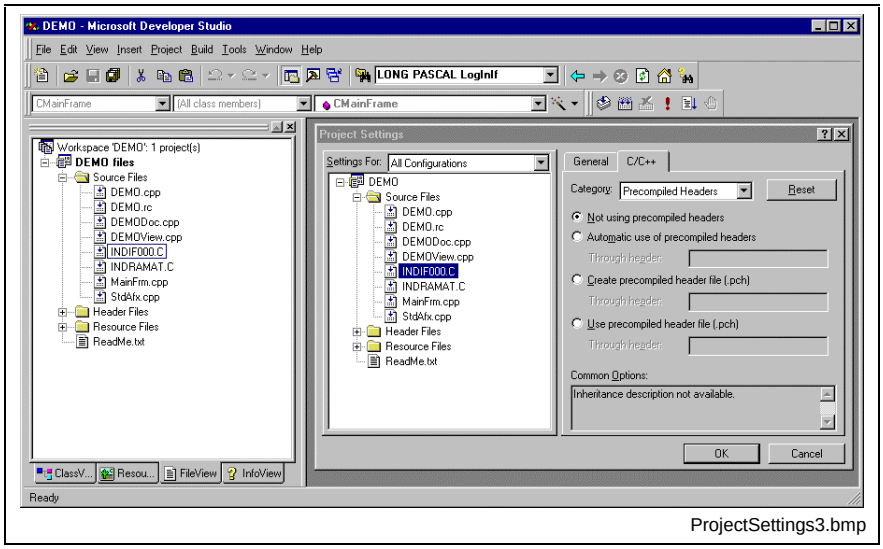


ProjectSettings2.bmp

Fig. 3-4: Project-settings „For Win32 Debug“: Debug Multithreaded DLL

In the Project-Settings: „For All Configurations“, on the tab entitled „C/C++“ under the category „Precompiled Headers“, the „Not using precompiled Headers“ setting must be selected for the C sources:

- INDIF000.C and
- INDRAMAT.C.



ProjectSettings3.bmp

Fig. 3-5: Project settings: „For All Configurations“

3.2 Routines for log-on and log-off

The log-on routine „LogInIf" must always first be called up before the access functions in the following chapters can be used. The log-off routine „LogOutIf" must be called "up when it is intended to finish working with the function interface.

Log-on routine „LogInIf"

Explanation With the routine „LogInIf", a user program logs on with the management structure of the function interface.

Syntax

```
LONG PASCAL LogInIf (   CHAR *lpcTaskName,
                        CHAR *lpcCommandLine,
                        CHAR *lpcParentWinName,
                        HANDLE *lhTerminateEvent,
                        UCHAR lucIfChannel,
                        UCHAR lucIfChannelGrp,
                        HANDLE *lhSysMsgEvent,
                        UCHAR *lucTaskId,
                        DWORD *ldwIFChannelId   );
```

Transfer parameters

Parameter	Explanation
[IN] lpcTaskName	Pointer to the user program name.
[IN] lpcCommandLine	Pointer to the command line for the BOF and logic process. Generally, the transfer parameters of the user program are entered here. This enables the function interface to switch to the diagnostic mode via the command line.
[IN] lpcParentWinName	Pointer to the name of the parent window of the process. max. length = MAX_PARENT_WIN_NAME_LEN. (see file „INDIF000.H" or „INDIF000.BAS") NULL = no parent window available (normal case)
[OUT] lhTerminateEvent	HANDLE to the termination event of a process. NULL = no TEvent available
[IN] lucIfChannel	Defines whether or not the process requests a LOG channel. 0 = no LOG channel request >0 = LOG channel is requested (normal case).
[IN] lucIfChannelGrp	Maximum number of function calls within a group request [1...MAXGRP]. Default: 10 (see entries n the „INDIFX00.H" file)
[OUT] lhSysMsgEvent	HANDLE to the Sys-Msg event. NULL = no Sys-Msg event available
[OUT] lucTaskId	TaskID that a user program is assigned when logging on for management purposes [1..MAX_TASK_ANZAHL] (see entries in the „INDIFX00.H" file).
[OUT] ldwIFChannelId	Assigned ID of the communication channel [2...8]

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried with the in the form of a general error result line with the „„ReadGroupItem“. For further information, refer to the chapter entitled „General error result line”.

LogInIf example (Visual Basic: VBDEMO.FRM)

```
Private Sub Form_Load()

'INPUT-Values of the LogInIf-routine
'***********************************
Dim TaskName As String            'Application’s name
Dim CommandLine As String         'Command for starting conditions, e.g. „/C=t /B=w“
Dim ParentWinName As String       'Titlebar’s (Window’s )name
Dim IfChannel As Byte             'Funktion-Interface Channel
Dim IfChannelGrp As Byte          'Value for group request
Dim ResBuf As String * 32768      'Resultbuffer

'Return-Values of the LogInIf-routine
'************************************
Dim TaskId As Byte
Dim IfChannelId As Long

'General declarations
'********************
Dim lRet As Long                    'Routine's returnvalue
Dim ErrMsg As String                'Error message string
Dim nHookList(0 To 4) As Integer    'Number of FI-System Messages (FI-SYS-MSGs)
Dim lpThreadId As Long

'Timer interval initialisation
'*****************************
TimerInterval.Caption = CyclicOutputTimer.Interval
CycleTime.Value = CyclicOutputTimer.Interval

TaskName = „VBDemo.exe“     'Application’s name
CommandLine = Command   'Command for starting conditions, e.g. /C=t /B=w
ParentWinName = „VBDemo“    'Titlebar’s (Window’s )name
IfChannel = 1               'Funktion-Interface Channel 1 requested
IfChannelGrp = 10           'Max. value for group request
lRet = 1                    'Default Returnvalue = 1 for error handling

'Call LogInIf-Routine (Start Interface)
'**************************************
lRet = LogInIf(TaskName, CommandLine, ParentWinName, SysThread.hTerminateEvent, IfChannel, IfChan-
nelGrp, SysThread.hSysMsgEvent, TaskId, IfChannelId)

'Error handling & Function-interface channel identification output
'*****************************************************************
If lRet Then   'error handling
    VBDemoStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
    ErrMsg = „LogIn Error code: „ + CStr(lRet)
    VBDemoStatus.Caption = ErrMsg
Else    'Function-interface channel identification output
    VBDemoStatus.BackColor = QBColor(10) 'set BackgroundColor to bright green
    VBDemoStatus.Caption = „Login succeeded on FI-Channel „ & IfChannelId
End If

'Creating Funktion-Interface-System-Message-List (FI-SysMsg)
'***********************************************************
nHookList(0) = 4                'Number of FI-SYS-MSGs
nHookList(1) = MSG_PCLUPDBEG    'PLC Download Begin
nHookList(2) = MSG_PCLUPDEND    'PLC Download End
nHookList(3) = MSG_PARUPDBEG    'Parameter Download Begin
nHookList(4) = MSG_PARUPDEND    'Parameter Download End

lRet = HookIfMsgList(nHookList(0))      'Ptr-Handed over in Basic is equalto C

If lRet Then   'error handling
    SYS_Messages.BackColor = QBColor(12) 'set BackgroundColor to bright red
    ErrMsg = „HookIfMsgList terminated with error code: „ + CStr(lRet)
    SYS_Messages.Caption = ErrMsg
End If
```

```
'Starting FI-SYS-Msg Thread
'*************************
hThread = CreateThread(0, _
                    0, _
                    AddressOf SysThread.SysMsgThreadProc, _
                    0, _
                    0, _
                    lpThreadId)

If hThread = 0 Then 'error handling
    SYS_Messages.BackColor = QBColor(12) 'set BackgroundColor to bright red
    ErrMsg = „Thread couldn't be created“ & Err.LastDllError
    SYS_Messages.Caption = ErrMsg
End If

'Process verification for the Function-Interface
'**********************************************
lRet = DataTransfer(„XX_BW_RPR1“, 0, 0, 1, ResBuf, 32768, 1)

End Sub
```

LogInIf example (Visual C++)

```
// General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80];
// Input parameters of the LogInIf routine
//*********************************************
HANDLE ghTerminateEv = NULL;
HANDLE ghSysMsgEv = NULL;
UCHAR gucTaskId = 0;
DWORD gdwlFChannelId  = 0;
// LogInIf routine (Start Interface)
//***********************************
IRet = LogInIf(„VCDemo.exe“,  // Name of user program,
m_lpCmdLine,                  // Command, e.g. „/C=t“,
„Demo“,               // Window´s Name,
&ghTerminateEv,               // HANDLE to  TerminateEvent,
1,                            // Interface channel requested,
10,                           // Max. number of function calls in group,
&ghSysMsgEv,                  // HANDLE to Sys-Msg event,
&gucTaskId,                   // Task-ID,
&gdwlFChannelId);             // Communication channel ID

// Error Handling
//*******************
if (lRet)
{
     sprintf(acErrMsg,“Function-Interface LogInIf ErrorCode: %ld „,lRet);
     MessageBox (GetFocus(),acErrMsg,“Function Interface Error“, MB_OK);
}
```

Log-off routine „LogOutIf“

Explanation With „LogOutIf”, a user program logs off from the management structure of the function interface.

Syntax **LONG PASCAL LogOutIf ( );**

Transfer parameters None

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „„ReadGroupItem“. For further information, refer to the chapter entitled „General error result line“.

LogOutIf example (Visual Basic: VBDEMO.FRM)

```
Public Sub Form_Terminate()
'IN-/Output Values
'*****************
Dim lRet As Long            'Routine's returnvalue
Dim ErrMsg As String        'Error message string

'Closing Function-Interface Channel
'**********************************
lRet = LogOutIf()           'Stop Function-Interface

If lRet Then 'error handling
    VBDemoStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
    ErrMsg = „LogOut termination with error code: „ + CStr(lRet)
    VBDemoStatus.Caption = ErrMsg
End If

CloseHandle (hThread)       'Thread clearance

End Sub
```

LogOutIf example (Visual C++)

```
// General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80];
//LogOutIf routine (Stop Interface)
//************************************
IRet = LogOutIf();
// Errorr Handling
//*****************
if (lRet)
{
     sprintf(acErrMsg,"Function-Interface LogOutIf ErrorCode: %ld „,lRet);
     MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
}
```

3.3 Data transfer and result evaluation routines

The „DataTransfer" routine serves to read and write data from and to Indramat devices. The data returned in the result buffer has the following structure:

Single request

Group element 1	Row 1	Column 1	...	Column j
	:	:	:	:
	Row m	Column 1	...	Column j

Note: As only a command line and **not** a group of command lines can be edited by way of the „DataTransfer" routine, it has only one result group. The data in the result buffer can be evaluated by means of the „ReadGroupItem" item.

The „GetNumberOfGroups" routine determines the number of groups in the result. The number of rows that possess a certain group is determined by way of the "GetNumberOfRows" routine and the number of columns in the line by means of the „GetNumberOfItems" routine.

Group request (cyclic requests ONLY)

In the cases of a group request, several command lines (individual requests) are requested at the same time. The command lines of a group request are separated by blanks. For each of these command lines in a group request, precisely one group element is returned in the result. Therefore the data returned in the result buffer has the following structure:

Group element 1	Row 1	Column 1	...	Column j
	:	:	:	:
	Row i	Column 1	...	Column j
:	:	:	:	:
Group element n	Row 1	Column 1	...	Column j
	:	:	:	:
	Row i	Column 1	...	Column j

Example of a group request

In the event of a group request (BR_NPS... BR_ABN... BR_AGF...), the individual group elements can be accessed with *[bGroup]*. Meanings:

- *1st group element (bGroup = 1)*: BR_NPS...
- *2nd group element (bGroup = 2)*: BR_ABN...
- *3rd group element (bGroup = 3)*: BR_AGF...

„DataTransfer“ routine

Explanation

With the „DataTransfer“ routine, data is read or written in accordance with the parameterised functions (see chapter entitled „Function interface commands”).

Syntax

```
LONG PASCAL DataTransfer (  CHAR *pszFunction,
                            CHAR acValue[ ],
                            LONG *ValLen,
                            LONG *ValType,
                            CHAR acResBuf[ ],
                            LONG lMaxResLen,
                            LONG lResBufType   );
```

Transfer parameters

Parameters	Explanation
[IN] pszFunction	Command line
[IN] acValue	Value to write
[IN] ValLen	Length of the value to write.
[IN] ValType	Data code of the value to write.
[OUT] acResBuf	Result buffer
[IN] lMaxResLen	Length of the result buffer depending on the requested data. 1K = 1024 bytes can be assumed as a reference value.
[IN] lResBufType	Data code of result data

Note: The data returned in the result buffer is coded. For access to the individual elements, the contents of the result buffer must be processed further with the „„ReadGroupItem“.

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „„ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

DataTransfer example (Visual Basic: VBDEMO.FRM)

```
Private Sub DataTransferFunc()

'Read/Write Data from/to the various devices via the function-interface
'*********************************************************************
Dim ResBuf As String * 32768      'Resultbuffer
Dim lRet As Long                  'Routine's returnvalue
Dim lLen As Long                  'Value's lenght
Dim pszFunction As String         'FI-command
Dim ErrMsg As String              'Error message string
Dim szBuf As String * 32768       'Buffer for controller data
Dim DataValidation As Boolean     'Flag for data validation
Dim szVal As String               'Writevalue

pszFunction = SingleRequest.Text  'Hand over FI-Command from Editbox
szVal = WriteValue.Text           'Hand over WriteValue from Editbox

'DataTransfer to function-interface
'**********************************
lRet = DataTransfer(pszFunction, szVal, Len(szVal), 1, ResBuf, 32768, 1)

If lRet Then   'error handling
    ErrMsg = „DataTransfer terminated with error code: „ + CStr(lRet)
    SingleRequestStatus.Caption = ErrMsg
    SingleRequestStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
    lRet = ReadGroupItem(ResBuf, 1, -1, -1, szBuf, 32768, lLen, DataValidation)
    Output.Text = szBuf
Else 'Valid reply
    SingleRequestStatus.BackColor = QBColor(10) 'set BackgroundColor to bright green
    SingleRequestStatus.Caption = „DataTransfer command was succesfully completed"
    lRet = ReadGroupItem(ResBuf, 1, -1, -1, szBuf, 32768, lLen, DataValidation)
    Output.Text = szBuf
End If
End Sub
```

DataTransfer example (Visual C++)

```
// General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80];
int i,j;

// Input parameters of the DataTransfer routine
//*****************************************************
CHAR * szValue = „„;
CHAR * szFunction = „02_CR_CCP4";
CHAR acResultbuf[1000];

// Input parameters of the GetNumberOfRows routine
//************************************************************
LONG lNumOfRows;

// Input parameters of the GetNumberOfItems routine
//************************************************************
LONG lNumOfItems;

// Input parameters of the ReadGroupItem routine
//*****************************************************
LONG lItemLen;
CHAR acItembuf[50];
BOOL boItemValid;

// Access to function interface
//**********************************

lRet = DataTransfer (szFunction,     // Command line,
szValue,                             // Value,
strlen(szValue),                     // Length of value,
1,                                   // Data code of value,
acResultbuf,                         // Result buffer,
1000,                                // Length of resultbuffer,
1);                                  // Data code of result data
// Error Handling

  if (lRet)
{
      sprintf(acErrMsg,"Function-Interface DataTransfer ErrorCode: %ld „,lRet);
      MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);

}
```

```
// Determining the number of rows
//*******************************
lRet = GetNumberOfRows(acResultbuf,  // Result data,
1,                                   // Group element,
& lNumOfRows);                       // Number of rows

 // Error Handling

 if (lRet)
{
  sprintf(acErrMsg,"Function-Interface GetNumberOfRows ErrorCode: %ld „,lRet);
  MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
 }

// Result evaluation
//************************
for (i=1; i<=lNumOfRows; i++)

{

     // Determining the number of elements
     //*******************************
     lRet = GetNumberOfItems(acResultbuf,    // Result data,

     1,                              // Group element,
     i,                              // Row,
     &lNumOfItems);                          // Number of elements ?

     // Error Handling

     if (lRet)
     {
   sprintf(acErrMsg,"Function-Interface GetNumberOfItems ErrorCode:%ld „,lRet);
      MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
     }

     // Evaluating all rows
     //*************************

     for (j=1; j<=lNumOfItems; j++)

     {

        // Evaluating all results of one row

       lRet=ReadGroupItem(acResultbuf,       // Result data,
       1,                                    // Group element,
       i,                                    // Row,
       j,                                    // Column,
       acItembuf,                            // single result,
       50,                                   // Length of the single result buffer,
       &lItemLen,                            // Length of the result,
       &boItemValid);                        // Value valid ?

       // Error handling

         if (lRet)
        {
         sprintf(acErrMsg,"Function-Interface ReadGroupItem ErrorCode:%ld „,lRet);
        MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
        }

     }

}
```

„ReadGroupItem“ routine

Explanation

This routine permits reading out of a single result, a whole row or a table of a single or group request. All result data must always be evaluated by way of this routine.

Syntax

```
LONG PASCAL ReadGroupItem (     CHAR acResBuf[ ]
                                BYTE bGroup,
                                LONG lRow,
                                LONG lItem,
                                CHAR acItemBuf[ ],
                                LONG lItemBufLen,
                                LONG *plItemLen,
                                BOOL *pbItemValid );
```

Transfer parameters

Parameters	Explanation and value range
[IN] CHAR acResBuf[ ]	Buffer for the overall result
[IN] BYTE bGroup	Entry of the group element [1...n]
[IN] LONG lRow	-1: Output of a whole table and thus all rows of a request. [1...n]: The relevant result row
[IN] LONG lItem	-1: Output of one row 0: Output of the telegram header [1...n]: Single result (element of a row)
[OUT] CHAR acItemBuf[ ]	Buffer for requested partial result
[IN] LONG lItemBufLen	Length of buffer for partial result
[OUT] LONG plItemLen	Length of partial result
[OUT] BOOL pbItemValid	TRUE: if the value of the partial result is valid.

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Example of the „ReadGroupItem“ routine

The following example is based on the assumption that a single request (*bGroup* = 1) has been requested:

Row	Column 1	Column 2	Column 3	Column 4
1	E1	E2	E3	
2	E4	E5		
3	E6	E7	E8	E9

Syntax example	Result
ReadGroupItem(acResBuf, **1**, **1**, **1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E1
ReadGroupItem(acResBuf, **1**, **1**, **2**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E2
ReadGroupItem(acResBuf, **1**, **2**, **2**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E5
ReadGroupItem(acResBuf, **1**, **3**, **4**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E9
ReadGroupItem(acResBuf, **1**, **2**, **3**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	Error code
ReadGroupItem(acResBuf, **1**, **1**, **-1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E1 E2 E3
ReadGroupItem(acResBuf, **1**, **2**, **-1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E4 E5
ReadGroupItem(acResBuf, **1**, **3**, **-1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E6 E7 E8 E9
ReadGroupItem(acResBuf, **1**, **-1**, **-1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E1 E2 E3 E4 E5 E6 E7 E8 E9
ReadGroupItem(acResBuf, **1**, **1**, **0**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	returns the telegram code: e.g. 001234567800_CC_ ....

Example in Visual Basic/ C++ (see „DataTransfer“ routine)

GetNumberOfGroups“ routine

Explanation The „GetNumberOfGroups” routine determines the number of group elements.

Syntax

```
LONG PASCAL GetNumberOfGroups (    CHAR *pszValBuf,
                                   LONG *plGroupSize);
```

Transfer parameters

Parameters	Explanation
[IN] CHAR *pszValBuf	Buffer for the total result
[OUT] LONG plGroupSize	Number of group elements

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „„ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

Example in Visual C++ (see „DataTransfer“ routine)

„GetNumberOfRows“ routine

Explanation The „GetNumberOfRows” routine determines the number of rows in the specified group element.

Syntax

```
LONG PASCAL GetNumberOfRows (   CHAR *pszValBuf,
                                BYTE bGroupIndex,
                                LONG *plNumberOfRow );
```

Transfer parameters

Parameters	Explanation
[IN] CHAR *pszValBuf	Buffer for the total result
[IN] BYTE bGroupIndex	Number of the group element
[OUT] LONG *plNumberOfRow	Number of rows in a group element

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „„ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

Example in Visual C++ (see „DataTransfer“ routine)

„GetNumberOfItems“ routine

Explanation The „GetNumberOfItems“ routine determines the number of partial results depending on the row number and the number of the group element.

Syntax

```
LONG PASCAL GetNumberOfItems (   CHAR *pszValBuf,
                                 BYTE bGroupIndex,
                                 BYTE bRowIndex,
                                 LONG *plNumberOfItems);
```

Transfer parameters

Parameters	Explanation
[IN] CHAR *pszValBuf	Buffer for the total result
[IN] BYTE bGroupIndex	Number of the group element
[IN] BYTE bRowIndex	Row index 0: number of all partial results
[OUT] LONG plNumberOfItems	Number of partial results for one specific row

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

Example in Visual C++ (see „DataTransfer“ routine)

3.4 Routines for cyclic reading via pipes

The pipe access function serves the purpose of cyclic reading of device data via the function interface. Several command lines can be passed on simultaneously by means of one group request. The command lines of one group request are separated by blanks (see also „„ReadGroupItem“). A pipe is started with the „StartCyclicPipe” routine and then constantly supplies itself with updated data. This data is now accessed asynchronously via the „ReadCyclicPipe” routine. The cyclic request is ended via the „StopCyclicPipe“ routine.

„StartCyclicPipe“ routine

Explanation The „StartCyclicPipe” routine starts a pipe for cyclic reading of data.

Syntax

```
LONG PASCAL StartCyclicPipe (  WORD wPipe,
                               CHAR *pszFunktion,
                               LONG lBufSize,
                               LONG lGroupSize,
                               DWORD dwSleep );
```

Transfer parameters

Parameters	Explanation
[IN] wPipe	Pipe number [1...1000]
[IN] *pszFunktion	String group in accordance with the defined function calls.
[IN] lBufSize	Result buffer size [bytes]
[IN] lGroupSize	Number of group elements [1...n]
[IN] dwSleep	Read delay time [ms]

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „„ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

StartCyclicPipe example (Visual Basic: VBDEMO.FRM)

```
Public Sub StartCyclicFunc()
'Start of a cyclic request
'*************************
Dim lRet As Long                'Routine's returnvalue
Dim ErrMsg As String            'Error message string
Dim pszFunction As String       'FI-command
pszFunction = CyclicRequest.Text    'Hand over FI-Command from Editbox
If Not CyclicRun Then 'in case of a cyclic request has NOT been started
    lRet = StartCyclicPipe(1, pszFunction, 32768, 2, 250)
    If lRet Then   'error handling
        CyclicRequestStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
        ErrMsg = „StartCyclicPipe terminated with error code: „ + CStr(lRet)
        CyclicRequestStatus.Caption = ErrMsg
        Exit Sub 'in case of an error has occured

    End If
    CyclicRun = True                    'Flag for a cyclic request is started
    CyclicOutputTimer.Enabled = True    'Timer output is started
End If
End Sub
```

StartCyclicPipe example (Visual C++)

```
// General declarations

//******************************
LONG lRet;
CHAR acErrMsg[80];

// Input parameters of the StartCyclicPipe routine
//*********************************************************
CHAR * szGroupFunction = „00_CC_AGF_0 00_CC_PVS_ErrorFlg“;
// Open pipe
//**************
lRet = StartCyclicPipe(wPipeNo,             // Pipe number,
szGroupFunction,                    // Function call group,
1000,                               // Result buffer size,
2,                                  // Number of group elements,
500);                               // Read delay time [ms]

// Error handling
//*******************
if (lRet)
{
    sprintf(acErrMsg,“Function-Interface StartCyclicPipe ErrorCode: %ld „,lRet);
    MessageBox (GetFocus(),acErrMsg,“Function Interface Error“, MB_OK);
}
```

„ReadCyclicPipe“ routine

Explanation The „ReadCyclicPipe” routine reads the data of a pipe started with „StartCyclicPipe”.

Syntax

```
LONG PASCAL ReadCyclicPipe (    WORD wPipe,
                                CHAR acResult[ ],
                                LONG lBufSize,
                                BYTE *pbGroupFault,
                                LONG *plAttr  );
```

Transfer parameters

Parameters	Explanation
[IN] WORD wPipe	Number of the pipe
[OUT] CHAR acResult[]	Buffer for the total result
[IN] LONG lBufSize	Buffer size of the total result
[OUT] BYTE *pbGroupFault	Number of the group element in the event of an error.
[OUT] LONG *plAttr	Result attribute

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

ReadCyclicPipe example (Visual Basic: VBDEMO.FRM)

```
Private Sub CyclicOutputTimer_Timer()

'IN-/Output Values
'*****************
Dim lRet As Long                    'Routine's returnvalue
Dim ErrMsg As String                'Error message string
Dim ResultBuffer As String * 32768
Dim lNumberOfRows As Long           'Number of Rows ->
Dim i As Long                       'Index for the number of rows
Dim szBuf As String * 256           'Buffer for controller data
Dim lLen As Long                    'Value's lenght
Dim DataValidation As Boolean       'Flag for data validation
Dim bGroup As Byte
Dim lAttr As Long

lRet = ReadCyclicPipe(1, ResultBuffer, 32768, bGroup, lAttr)

If lRet Then   'error handling
    CyclicRequestStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
    ErrMsg = „ReadCyclicPipe terminated with error code: „ + CStr(lRet)
    CyclicRequestStatus.Caption = ErrMsg
    Exit Sub
End If
OutputList.Clear
If lRet = 0 Then
            lRet = GetNumberOfRows(ResultBuffer, 1, lNumberOfRows)
            Rows.Text = lNumberOfRows
            If lRet Then   'error handling
                CyclicRequestStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
                ErrMsg = „GetNumberOfRows terminated with error code: „ + CStr(lRet)
                CyclicRequestStatus.Caption = ErrMsg
            End If
            For i = 1 To lNumberOfRows
                lRet = ReadGroupItem(ResultBuffer, 1, i, -1, szBuf, 32768, lLen, DataValidation)
                If lRet Then   'error handling
                   CyclicRequestStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
                   ErrMsg = „ReadGroupItem terminated with error code: „ + CStr(lRet)
                   CyclicRequestStatus.Caption = ErrMsg
                End If
                OutputList.AddItem (szBuf)
            Next
            CyclicRequestStatus.BackColor = QBColor(10) 'set BackgroundColor to bright green
            CyclicRequestStatus.Caption = „ReadCyclicPipe command was succesfully completed“
End If
End Sub
```

ReadCyclicPipe example (Visual C++)

```
// General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80];
int i;
// Input parameters of the ReadCyclicPipe routine
//*************************************************************
CHAR acResultbuf[1000];
UCHAR bIndexItemFault;
LONG lAttr;
// Input parameters of the GetNumberOfGroups routine
//*************************************************************
LONG lNumOfGroups;
// Read pipe
//*************
lRet = ReadCyclicPipe(wPipeNo,              // Pipe number,
acResultbuf,                                // Result buffer,
1000,                                       // Length of result buffer,
&bIndexItemFault,                           // Index of the group
                                            // element with an error,
&lAttr);                                    // Result attribute
// Error handling
if (lRet)
{
     sprintf(acErrMsg,"Function-Interface ReadCyclicPipe ErrorCode: %ld „,lRet);
    MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
}
// Determine number of groups
//****************************
lRet = GetNumberOfGroups(acResultbuf,              // Result buffer,
       &lNumOfGroups);// Number of groups,
// Error handling
if (lRet)
{
sprintf(acErrMsg,"Function-Interface GetNumberOfGroups ErrorCode: %ld",lRet);
   MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
}
// Result evaluation
//************************
for (i=1; i<=lNumOfGroups; i++)
{
     // Result evaluation for each group result
     // e.g.   LONG lItemLen;
     //CHAR acItembuf[50];
     //int iItemValid;
     //
    lRet=ReadGroupItem(acResultbuf,  // Result buffer,
    i,                               // Group element,
    1,                               // Row,
    1,                               // Element,
    acItembuf,                       // Single result buffer,
    50,                     // Length of single result buffer,
    &lItemLen,                       // Length of single result,
    &iItemValid);                    // Single result valid?
// Error handling
     if (lRet)
     {
        sprintf(acErrMsg,"Function-Interface ReadGroupItem ErrorCode: %ld „,lRet)
        MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
     }
}
```

„StopCyclicPipe“ routine

Explanation The „StopCyclicPipe” routine stops the data request of a pipe started with „StartCyclicPipe”.

Syntax **LONG PASCAL StopCyclicPipe (WORD wPipe);**

Transfer parameters

Parameters	Explanation
[IN] WORD wPipe	Pipe number

Return values
0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „„ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

StopCyclicPipe example (Visual Basic: VBDEMO.FRM)

```
Public Sub StopCyclicFunc()
'Stop of a cyclic request
'************************

'IN-/Output Values
'*****************
Dim lRet As Long            'Routine's returnvalue
Dim ErrMsg As String        'Error message string

'Cyclic request termination
'**************************
If CyclicRun Then 'in case of a cyclic request has been started
    CyclicOutputTimer.Enabled = False       'Timer output is stopped
    lRet = StopCyclicPipe(1)

    If lRet Then   'error handling
        CyclicRequestStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
        ErrMsg = „StopCyclicPipe terminated with error code: „ + CStr(lRet)
        CyclicRequestStatus.Caption = ErrMsg
    End If

    CyclicRun = False 'Flag for a cyclic request is stoped

End If
End Sub
```

StopCyclicPipe example (Visual C++)

```
// General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80];

// Close pipe
//*******************
lRet = StopCyclicPipe(wPipeNo);             // Pipe number
// Error handling
//*******************

if (lRet)
{
     sprintf(acErrMsg,"Function-Interface StopCyclicPipe ErrorCode: %ld „,lRet);
     MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
}
```

„SuspendCyclicPipe“ routine

Explanation The „SuspendCyclicPipe” routine sets the data request of a pipe started with „StartCyclicPipe” to a standby mode. It serves to stop communication while at the same time keeping to the management structure of the function interface established by the „StartCyclicPipe“ routine (see „ResumeCyclicPipe“).

Syntax **LONG PASCAL SuspendCyclicPipe (WORD wPipe);**

Transfer parameters

Parameter	Explanation
[IN] WORD wPipe	Number of the pipe

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „„ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

SuspendCyclicPipe example (Visual Basic: VBDEMO.FRM)

```
Public Sub SuspendCyclicFunc()
'Stand-by-Modus for a cyclic request
'***********************************

'IN-/Output Values
'*****************
Dim lRet As Long            'Routine's returnvalue
Dim ErrMsg As String        'Error message string

 If CyclicRun Then 'in case of a cyclic request has been started
    CyclicOutputTimer.Enabled = False       'Timer output is stopped
    lRet = SuspendCyclicPipe(1)

     If lRet Then   'error handling
         CyclicRequestStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
         ErrMsg = „SuspendCyclicPipe terminated with error code: „ + CStr(lRet)
         CyclicRequestStatus.Caption = ErrMsg
     End If

  End If
End Sub
```

SuspendCyclicPipe example (Visual C++)

```
// General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80];

// Suspend pipe
//************************
lRet = SuspendCyclicPipe(wPipeNo);   // Pipe number

// Error handling
//*******************

if (lRet)
{
  sprintf(acErrMsg,"Function-Interface SuspendCyclicPipe ErrorCode: %ld „,lRet);
   MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
}
```

„ResumeCyclicPipe“ routine

Explanation The „ResumeCyclicPipe” routine reactivates the data request of a pipe set to the standby mode with the „SuspendCyclicPipe“ routine.

Syntax **LONG PASCAL SuspendCyclicPipe (WORD wPipe);**

Transfer parameters

Parameter	Explanation
[IN] WORD wPipe	Number of the pipe

Return values
0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

ResumeCyclicPipe example (Visual Basic: VBDEMO.FRM)

```
Public Sub ResumeCyclicFunc()
'Activates a suspended cyclic Pipe
'*********************************

'IN-/Output Values
'*****************
Dim lRet As Long            'Routine's returnvalue
Dim ErrMsg As String        'Error message string

'Cyclic request termination
'**************************
 If CyclicRun Then 'in case of a cyclic request has been started
    CyclicOutputTimer.Enabled = True       'Timer output is started
    lRet = ResumeCyclicPipe(1)


    If lRet Then   'error handling
        CyclicRequestStatus.BackColor = QBColor(12) 'set BackgroundColor to bright red
        ErrMsg = „ResumeCyclicPipe terminated with error code: „ + CStr(lRet)
        CyclicRequestStatus.Caption = ErrMsg
    End If

End If
End Sub
```

ResumeCyclicPipe example (Visual C++)

```
// General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80];

// Resume pipe
//*******************

lRet = ResumeCyclicPipe(wPipeNo);    // Pipe number
// Error handling
//******************

if (lRet)
{
   sprintf(acErrMsg,"Function-Interface ResumeCyclicPipe ErrorCode: %ld „,lRet);
  MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
}
```

3.5 Access functions for working with SYS messages

Although each user program operates in its own process space independently of other user programs, it can only access the function interface in cooperation with others.

For example, the user program cannot gain read or write access to controller data during an NC program download from the Indramat user interface.

To avoid this kind of resource conflict, the function interface announces a resource request from elsewhere by means of a SYS message event.

Now, depending on the incoming SYS message, the user program must disable access to the affiliated resource or can open it once again.

To ensure that the user program is only informed about the status changes by means of a SYS message that it actually needs, it logs on for specific SYS messages with the „HookIfMsgList".

It generally waits for the SYS message in a thread in parallel with the main program. After arrival of the event, the message number must first be fetched with the „GetIfMsg". After internal processing of the status change (disable or open etc.), the message must be acknowledged by means of the „SetIfMsgConf" because only then does the status change become effective.

„HookIfMsgList" routine

Explanation

The „HookIfMsgList" routine logs on a user program for specific SYS messages. The application can then be informed at any time, by the SYS message event, of the occurrence of one of these SYS messages.

Syntax

LONG PASCAL HookIfMsgList (WORD *lpwSysMsgList);

Transfer parameters

Parameter	Explanation
[IN] WORD *lpwSysMsgList	List of messages

Note: These messages must be acknowledged by the process.

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes" chapter).

Note: The error can also be queried in the form of a general error result line with the „„ReadGroupItem" routine. For further information, refer to the chapter entitled „General error result line".

Example in Visual Basic

(see Log-on routine „LogInIf")

HookIfMsgList example (Visual C++)

```
//General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80]

// Input parameters of the HookIfMsgList routine
//*****************************************************

WORD wSysMsgList[3] = {2,     // Number of messages,

MSG_PCLUPDBEG,                // Message constants for

MSG_PCLUPDEND};               // PLC download start and // end

// Log on for SYS-MSG
//***************************
lRet = HookIfMsgList((WORD*) &wSysMsgList);         // SYS-MSG list
// Error handling
//*******************

if (lRet)
{
     sprintf(acErrMsg,"Function-Interface HookIfMsgList  ErrorCode: %ld „,lRet);
    MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
}
```

„GetIfMsg“ routine

Explanation The „GetIfMsg” routine for fetching a pending SYS-MSG is called up once the user program has been informed by means of a SYS message event.

Syntax

```
LONG PASCAL GetIfMsg (    WORD *pwMsgNr,
                          CHAR *pcSysMsgBuffer,
                          WORD *pwSysMsgBufferLen,
                          CHAR *pcTaskName   );
```

Transfer parameters

Parameters	Explanation
[OUT] WORD *pwMsgNr	Contains the currently pending SYS-MSG message
[OUT] CHAR *pcSysMsgBuffer	Pointer to the data in the SYS-MSG BUFFER; contains the device address as ASCII characters
[OUT] WORD *pwSysMsgBufferLen	Length of the data in the SYS-MSG-BUFFER
[OUT] CHAR *pcTaskName	Pointer to the name of the task that has triggered the SYS-MSG message

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

Example in Visual Basic See SYS-MSG example (Visual Basic: SYSTHREAD.BAS)

GetIfMsg example (Visual C++)

```
//General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80];
// Input parameters of the GetIfMsg routine
//*********************************************
CHAR acSysMsgBuffer[MSG_BUFFER_LEN] =„„;
CHAR acTaskName[MAX_TASK_NAME_LENGTH] =„„;
WORD wMsgNr;
WORD wSysMsgBufferLen;

 // Fetch SYS message
//***************************
lRet=GetIfMsg(&wMsgNr,        // SYS-MSG number,
acSysMsgBuffer,                    // Buffer for SYS-MSG info,
&wSysMsgBufferLen,                 // Data length of SYS-MSG buffer,
acTaskName);                       // Buffer for name of triggering // task
// Error handling
//******************
if (lRet)
{
     sprintf(acErrMsg,"Function-Interface GetIfMsg ErrorCode: %ld „,lRet);
     MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
}
```

„SetIfMsgConf“ routine

Explanation The access function „SetIfMsgConf” serves to acknowledge the SYS-MSG message that has been received with „GetIfMsg”. The SYS-MSG acknowledgement message has the value SYS-MSG message „_Q“.

Syntax **LONG PASCAL SetIfMsgConf (WORD wMsgNr_Quit);**

Transfer parameters

Parameter	Explanation
[IN] WORD wMsgNr_Quit	SYS-MSG acknowledgement message

Return values

0: Request successfully completed.
1...n: Request completed with errors (see „Error codes“ chapter).

Note: The error can also be queried in the form of a general error result line with the „„ReadGroupItem“ routine. For further information, refer to the chapter entitled „General error result line“.

Example in Visual Basic See SYS-MSG example (Visual Basic: SYSTHREAD.BAS)

SetIfMsgConf example (Visual C++)

Assumption: wQMsgNr contains the SYS-MSG acknowledgement message

```
// General declarations
//*****************************
LONG lRet;
CHAR acErrMsg[80];
// Acknowledge SYS-MSG
//************************
lRet = SetIfMsgConf(wQMsgNr);              // SYS-MSG acknowledgement message
// Error handling
//******************
if (lRet)
{
     sprintf(acErrMsg,"Function-Interface SetIfMsgConf ErrorCode: %ld „,lRet);
     MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
}
```

What SYS messages are there what must the reaction be?

The SYS messages used most frequently for a user program are listed in the following table. All system messages can be found in the configuration files „INDIF000.H“ and „INDIF000.BAS“. The system messages always consist of a SYS-MSG and of a SYS-MSG acknowledgement (see „SetIfMsgConf“ routine).

Note: The device address where the system message has been sent is returned as ASCII characters in the buffer „CHAR *pcSysMsgBuffer“ of „GetIfMsg“. The buffer may contain further information specific to the SYS-MSG, e.g. the parameter identification string.

SYS message	What happens?	User program reaction
MSG_FWAUPDBEG	Start of the firmware download, i.e. firmware is downloaded into the System200 component of BOF/GBO.	Interruption of communication with all configured Indramat devices
MSG_FWAUPDEND	End of firmware download.	Communication resumed. **NOTE**: Process and axis configuration data etc. may have changed.
MSG_PARUPDBEG	Start of parameter download, i.e. a parameter set is downloaded into the System200 component of BOF/GBO.	Interruption of communication with all configured Indramat devices.
MSG_PARUPDEND	End of parameter download.	Communication resumed. **NOTE**: Process and axis configuration data etc. may have changed.
MSG_PCLUPDBEG	Start of PLC program download, i.e. a PLC program is downloaded into the System200 component of BOF/GBO.	Interruption of communication with all configured Indramat devices.
MSG_PCLUPDEND	End of PLC program download.	Communication resumed.
MSG_MEMUPDBEG	BOF/GBO begins deleting the data memory in the System200 component.	Interruption of communication with all configured Indramat devices.
MSG_MEMUPDEND	BOF/GBO has deleted the data memory in the System200 component.	Communication resumed. **NOTE**: Configuration data has been deleted.
MSG_SYSERRGEN	This SYS-MSG is sent if a system error occurs, i.e. access to the PLC is currently not possible.	**No** interruption of communication with Indramat devices necessary. (Serves to visualise a system error of a specific Indramat device.)
MSG_SYSERRDEL	A system error is deleted.	**No** interruption of communication with Indramat devices necessary. (Information to the effect that the system error on a specific Indramat device has disappeared.)
MSG_MECERRGEN	This SYS-MSG is sent if a mechanism error occurs.	**No** interruption of communication with Indramat devices necessary. (Serves to visualise a mechanism error of a specific Indramat device.)
MSG_MECERRDEL	A mechanism error is cleared.	**No** interruption of communication with Indramat devices necessary. (Information to the effect that the mechanism error on a specific Indramat device has disappeared.)

SYS message	What happens?	User program reaction
MSG_LAGCHABEG	Language change on the Indramat user interface has been initialised.	**No** interruption of communication with Indramat devices necessary. (Information to the effect that the language of the user interface has been changed over.)
MSG_LAGCHAEND	Language change on the Indramat user interface has been completed.	**No** interruption of communication with Indramat devices necessary. (Information to the effect that the language of the user interface has been changed over.)
MSG_PC__ALIVE	A PC/device logs on/off in the PC network. The device address/FarDevice address from where the system message has been sent is returned as ASCII characters in the buffer „CHAR *pcSysMsgBuffer“ of „GetIfMsg“. The buffer also contains the information „0“ for PC logged off or „1“ for PC logged on.	Depending on the applications, this system message serves on the one hand as pure information about logging on/off by PCs and on the other hand an interruption in communication with the PC/device or resumption of communication with subsequent synchronisation is necessary.

SYS-MSG example (Visual Basic: SYSTHREAD.BAS)

```
Public Sub SysMsgThreadProc()
'This subroutine is processing FI-SYS-Messages
'*********************************************
Dim lWait As Long
Dim hEvList(0 To 1) As Long
'Transmission of events whose the thread is reacting
'**************************************************
hEvList(0) = hTerminateEvent
hEvList(1) = hSysMsgEvent
Demo.SYS_Messages.BackColor = QBColor(10) 'set BackgroundColor to bright green
Demo.SYS_Messages.Caption = „Thread is runnning“
Do 'Threadloop
    lWait = WaitForMultipleObjects(2, hEvList(0), 0, INFINITE
    If lWait = 0 Then     'TerminateEvent from another FI-application has occured
        Demo.SYS_Messages.BackColor = QBColor(12) 'set BackgroundColor to bright red
        Demo.SYS_Messages.Caption = „Terminate Event has occured“
        Demo.TerminateEvent = True
        Exit Do     'End of the threadloop
    ElseIf lWait = 1 Then
    'SysMsgs which are received from the FI
    '**************************************
        Dim lRet As Long                  'Routine's returnvalue
        Dim nMsgNr As Integer
        Dim strMsgBuf As String * 256
        Dim strTaskName As String * 256
        Dim szMsg As String
        'Getting SYS-MSG-number
        '**********************
        lRet = GetIfMsg(nMsgNr, strMsgBuf, 256, strTaskName)
        If lRet Then    'error handling
            szMsg = „GetIfMsg terminated with error code: „ + CStr(lRet)
            Demo.SYS_Messages.Caption = szMsg
        Else
            Select Case nMsgNr
                Case MSG_PCLUPDBEG
                    Demo.SuspendCyclicFunc  'Termination of a cyclic request
                    lRet = SetIfMsgConf(MSG_PCLUPDBEG_Q)  'verification of the SYS-Message
                    If lRet Then    'error handling
                        szMsg = „SetIfMsgConf terminated with error code: „ + CStr(lRet)
                        Demo.SYS_Messages.Caption = szMsg
                    End If
                    Demo.SYS_Messages.BackColor = QBColor(11) 'set BackgroundColor to bright blue
                    Demo.SYS_Messages.Caption = „PLC Download BEGIN“
                Case MSG_PCLUPDEND
                    lRet = SetIfMsgConf(MSG_PCLUPDEND_Q) 'verification of the SYS-Message
                    Demo.ResumeCyclicFunc   'Start of a cyclic request
                    Demo.SYS_Messages.BackColor = QBColor(6) 'set BackgroundColor to brown
                    Demo.SYS_Messages.Caption = „PLC Download END“
                    If lRet Then 'error handling
                        szMsg = „SetIfMsgConf terminated with error code: „ + CStr(lRet)
                        Demo.SYS_Messages.Caption = szMsg
                    End If
                    Sleep (2000)    'Wait 2 sec.
                    Demo.SYS_Messages.BackColor = QBColor(10)'set BackgroundColor to bright green
                    Demo.SYS_Messages.Caption = „Thread is still runnning“
```

```
                Case MSG_PARUPDBEG
                    Demo.SuspendCyclicFunc  'Termination of a cyclic request
                    lRet = SetIfMsgConf(MSG_PARUPDBEG_Q) 'verification of the SYS-Message
                    If lRet Then 'error handling
                        szMsg = „SetIfMsgConf terminated with error code: „ + CStr(lRet)
                        Demo.SYS_Messages.Caption = szMsg
                    End If
                    Demo.SYS_Messages.BackColor = QBColor(11) 'set BackgroundColor to bright blue
                    Demo.SYS_Messages.Caption = „Parameter Download BEGIN“
                Case MSG_PARUPDEND
                    lRet = SetIfMsgConf(MSG_PARUPDEND_Q) 'verification of the SYS-Message
                    Demo.ResumeCyclicFunc   'Start of a cyclic request
                    Demo.SYS_Messages.BackColor = QBColor(6) 'set BackgroundColor to brown
                    Demo.SYS_Messages.Caption = „Parameter Download END“
                    If lRet Then    'error handling
                        szMsg = „SetIfMsgConf terminated with error code: „ + CStr(lRet)
                        Demo.SYS_Messages.Caption = szMsg
                    End If
                    Sleep (2000)    'Wait 2 sec.
                    Demo.SYS_Messages.BackColor = QBColor(10)
                    Demo.SYS_Messages.Caption = „Thread is still runnning“
            End Select
        End If
    ElseIf lWait = 2 Then
            Exit Do     'End of Threadloop
    End If
Loop
 End Sub
```

3.6 Tips and tricks for working with the interface

This chapter lists tips and tricks which ought to help you to get on faster when developing your user program (see also the chapter entitled „Frequently answered questions: FAQ“).

We have also found the working with Windows NT without a mouse repeatedly causes problems and so we have listed, in a table, the most important Windows NT keystroke combinations.

Problem	Remedy
You send an FI command in your application and receive: • no reply **- or -** • an unexpected reply **- or -** • an error code (see chapter entitled „Error codes“)	Frequent cause: Device address has not been specified **- or -** has been specified incorrectly. ⇒ Check the correct formulation of the FI command (see chapter entitled „Structure of function interface commands „ and the chapter „Function interface commands“). ⇒ Use the VBDemo to send the FI command that is causing problems. (See Sending FI commands with the „VBDemo“ application)
Your user program no longer reacts.	⇒ See Purging main memory with the „KILLTASK.EXE“.
Your user program terminates with a Windows general protection fault, „DR. WATSON“	⇒ See Purging main memory with the „KILLTASK.EXE“ ⇒ Rectify the programming error and restart your application.
The overall system (Windows NT, user program and Indramat user interface reacts sluggishly	⇒ Check the Windows NT settings for performance boosting, idle activity, swapping of the kernel mode drivers and idle activity as described in the chapter entitled „Settings of Windows NT system properties“
Purging the main memory with the „KILLTASK.EXE“ tool does not function, i.e. neither the three basic processes of the function interface nor your user program are removed from main memory.	⇒ Start the Task Manager, e.g. with the keystroke combination Ctrl + Shift + Esc (see chapter entitled „Windows NT Task Manager“) ⇒ Click the „Processes“ tab ⇒ End the three basic processes of the function interface and your user program: ⇒ LOGINTFC.EXE (logic process) ⇒ COMINTFC.EXE (communication process) ⇒ BOFINTFC.EXE (user interface process) ⇒ by clicking the End process button.
Your application terminates because: • required files are missing **- or -** • path entries do not exist/are not correct.	⇒ Check the presence of the required files in the intended directories. ⇒ Check the path entries **Note!** • Avoid absolute path entries in your application because any later drive or directory structure changes (e.g. from C:\ to D:\) will not be supported. • The system directory or the drive of Windows NT can also be chosen freely.

Purging main memory with the „KILLTASK.EXE“ tool

During the software creation process, this tool can be used to purge the main memory. Following default installation (see chapter entitled „Windows NT and function interface installation“), it is located in the default directory „C:\MT-CNC\IND_DRV\“.

The tool offers the following two possibilities of purging the main memory:

- Complete reinitialisation and
- Selective reinitialisation of the function interface.

Complete reinitialisation

When the „KILLTASK.EXE“ application is started, the three basic processes of the function interface:

- Logic process (LOGINTFC.EXE)
- Communication process (COMINTFC.EXE)
- User interface process (BOFINTFC.EXE)

and all applications that are logged on with the function interface are removed from main memory.

CAUTION

⇒ Correctly end all other (stable) function interface applications beforehand.

! Only then run the kill task !

Failure to observe this stipulation will render all warranty claims against Indramat null and void.

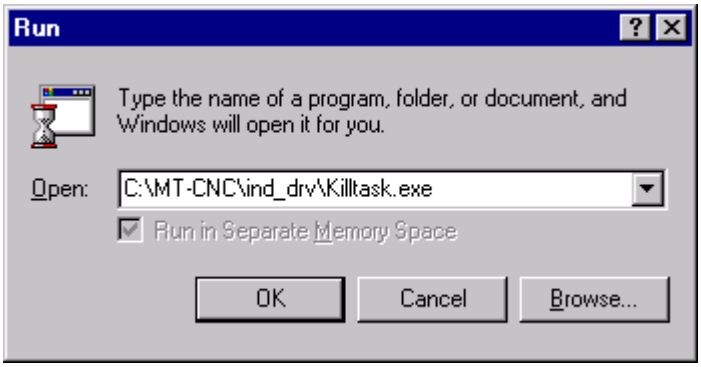
For complete reinitialisation, proceed as follows:

⇨ Click Start and then the „Run“ option.

⇨ Click the „Search“ button to search for the „KILLTASK.EXE“ tool.

Note: Following default installation, the „KILLTASK.EXE“ application (see chapter entitled „Windows NT and function interface installation“) is located in the default directory „C:\MT-CNC\IND_DRV\“.

⇨ Click the OK button.
All applications that are logged on with the function interface - and the basic processes of the function interface itself - are removed from main memory.



Killtask00.bmp

Fig. 3-6: Windows NT „Run“ dialog box: complete reinitialisation

Selective reinitialisation

Only the applications that are logged on with the function interface are removed the main memory and from the function interface management structure.

For selective reinitialisation, proceed as follows:

⇨ Click Start and then „Run“ option.

Note: You can search for the „KILLTASK.EXE“ application by clicking the „Search...“ button. After default installation (see chapter entitled „Windows NT and function interface installation“), this file is located in the default directory „C:\MT-CNC\IND_DRV\“.

⇨ In the text box area, enter the name of the application that you wish to remove from main memory and from the function interface management structure (in this case VBDemo.exe).

⇨ Then click the OK button.

The user program (in this case VBDemo.exe) is removed from main memory and from the function interface management structure

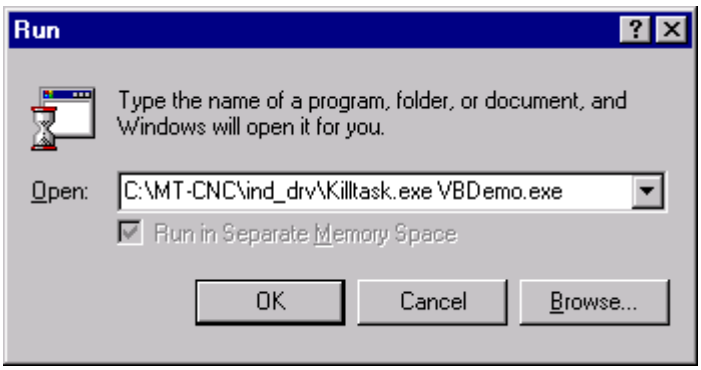


Fig. 3-7: Windows NT „Run“ dialog box: selective reinitialisation

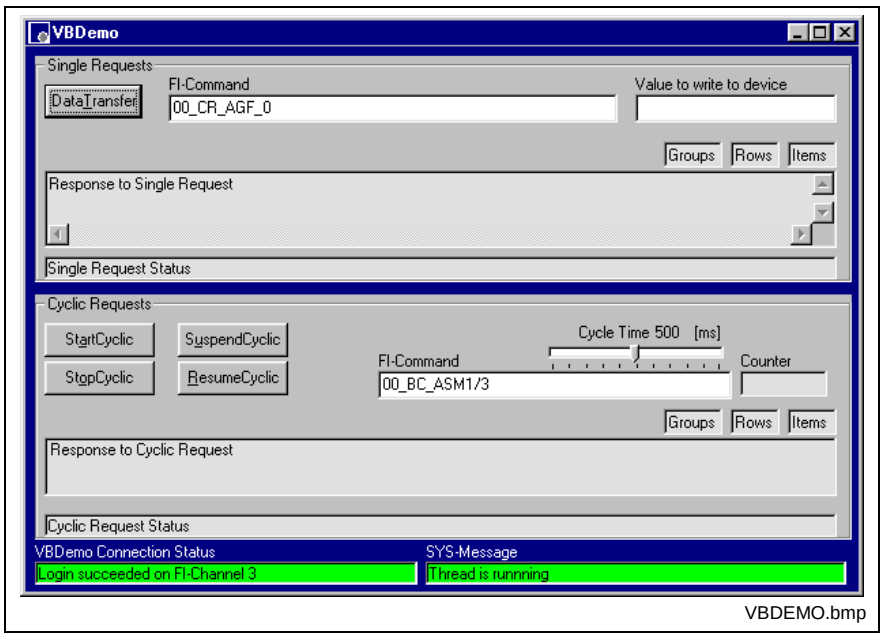
Sending FI commands with the „VBDemo“ application

Individual FI commands and cyclic requests can be sent with the „VBDemo“ application.

Proceed as follows to start the application:

⇨ Click Start, point to Programs, then to Indramat and finally to FI.

⇨ Click VBDemo.



VBDEMO.bmp

Fig. 3-8: The user program „VBDemo“

„Single Requests“ dialog box

This dialog box permits you to send individual reading and writing requests by means of „DataTransfer“.

Enter the FI command in the „FI-Command“ text box area. In the case of writing requests, in the text box area „Value to write to device“ additionally enter the value that you wish to write (see chapter entitled „Function interface commands“).

By clicking the DataTransfer button, then send the entered FI command to the function interface.

The response from the function interface is displayed in the „Response to Single Request“ text box.

„Cyclic Requests“ dialog box

This dialog box permits you to send writing cyclic requests by means of „StartCyclicPipe“.

Enter the FI command in the „FI-Command“ text box area (see chapter entitled „Function interface commands“).

Then send the entered FI command cyclically to the function interface by clicking the StartCyclic button.

The response from the function interface is displayed in the „Response to Cyclic Request“ text box.

Note: With the „Cyclic Time“ slider, you can modify the request interval from 10 to 1000 ms.

To end the cyclic request, click the StopCyclic button. This runs the „StopCyclicPipe“.

Note: If you click the SuspendCyclic button, the „SuspendCyclicPipe“ is run and the cyclic request is set to the standby mode.
To reactivate the cyclic request, click the ResumeCyclic button, thus running the „ResumeCyclicPipe“.

„VBDemo Connection Status“ dialog box

Indicates the log-on status of the application in the function interface. There are two states:

- Dialog box is highlighted in green and displays the function interface channel (LOG channel) that has been assigned to the application.
- Dialog box is highlighted in red and displays the error code that was caused on log-on by the Log-on routine „LogInIf“.

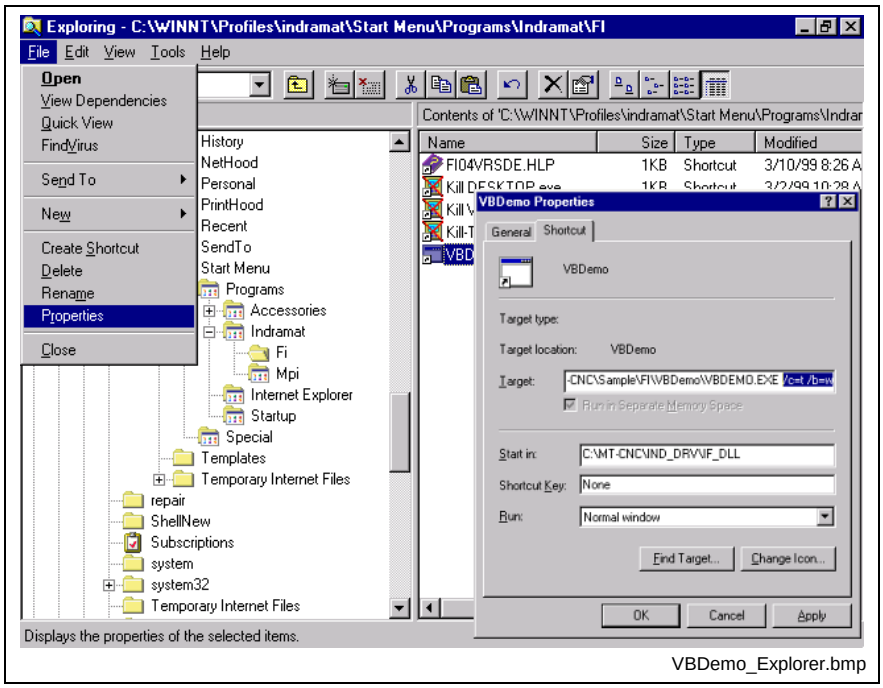
„SYS-Messages“ dialog box

This dialog box displays the function interface system messages that the user program „VBDemo“ is to react to (see Access functions for working with SYS messages).

Starting „VBDemo“ in the diagnostic mode

Proceed as follows to start the user program „VBDemo“ in the diagnostic mode:

⇨ Open the Windows NT Explorer: to do this, click Start, point to Programs and then click Windows NT Explorer.
⇨ Via Winnt, go to Profiles and into the user profile where the function interface has been installed.
⇨ Click Start menu, point to Programs, then to Indramat and then finally click FI.
⇨ Click VBDemo and open the Properties dialog via the Explorer's File menu.
⇨ Click the Shortcut tab and enter the start parameters „/c=t /b=w“in the „Target“ text box area.
⇨ Click the Close button and VBDemo will be started in the diagnostic mode the next time it is called up.



VBDemo_Explorer.bmp

Fig. 3-9: Starting VBDemo in the diagnostic mode of the function interface

Output of diagnostic information

By input of start parameters when starting the user interface process „BOFINTFC.EXE“, it is possible to display diverse function interface diagnostic information .

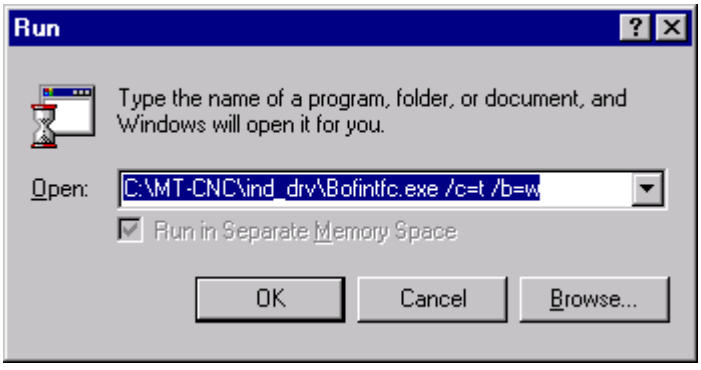
Proceed as follows to start the function interface in the diagnostic mode:

⇨ Click Start and then the „Run“ option.

Note: You can search for the user interface process „BOFINTFC.EXE“ by clicking the „Search...“ button. After default installation (see chapter entitled „Windows NT and function interface installation“), this file is located in the default directory „C:\MT-CNC\IND_DRV\“.

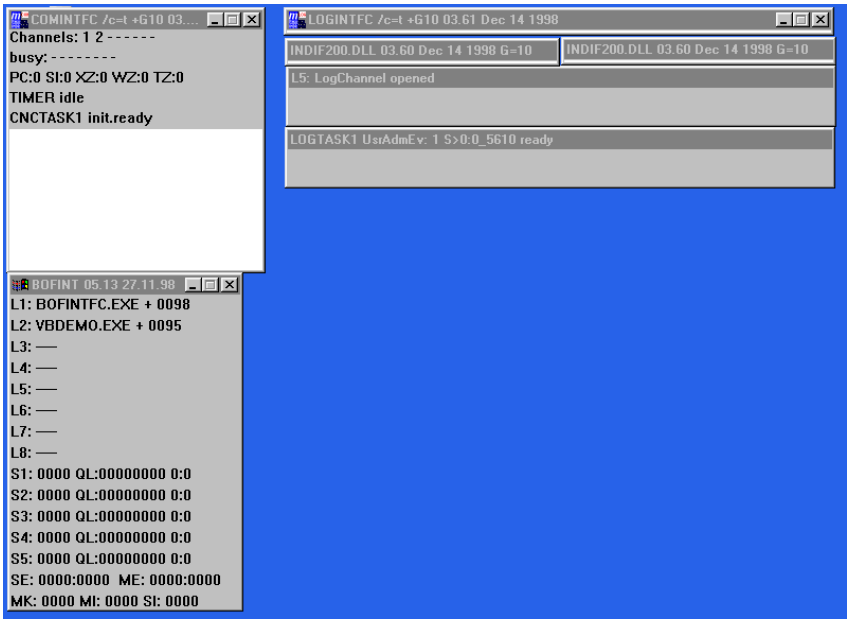
⇨ In the text box area, enter the start parameters „/c=t /b=w“ in lower case and separated by a blank.

⇨ Then click the OK button.
The function interface is now started in the diagnostic mode.



AusführenDiagnose.bmp

Fig. 3-10: Windows NT „Run“ dialog box: BOFINTFC.EXE



Diagnose.bmp

Fig. 3-11: Diagnostic mode of the function interface

Meanings of counters

The 3rd line of the diagnostic window of the communication process (COMINTFC.EXE) lists 5 counters. The individual counters have the following meanings:

PC Number of communication errors occurring in the device → PC transfer direction.

SI Number of communication errors occurring in the PC → device transfer direction.

XZ Number of repetitions of communication that were necessary to transfer a valid telegram to the device.

WZ This counter is incremented if, despite five repeated attempts, it has not been possible to transfer a valid telegram to the device. The counter is incremented by one if the „XZ“ counter has previously been incremented by five. In this case, the timeout counter is also incremented by one.

TZ Timeout counter. The number of occurring timeouts is counted in this counter. A timeout is generated if, despite five repeated attempts, it has not been possible to transfer a valid telegram to the device.

The active control channels (CNC/DMA task) are displayed in the bottom lines.

Data accesses of the individual logged-on application are displayed in the diagnostic windows of the LOG channel on the left of the screen.

The logged-on applications that are known to the management mechanism of the user interface process are displayed in the check window.

Windows NT keystroke combinations

The following table lists the most important keystroke combinations to facilitate operation of Windows NT without a mouse.

Action	Keystroke combination
Opening the Start bar	Ctrl + Esc
Navigating in the open Start bar or in the open submenus of the Start bar	Arrow keys left, right>, or Arrow keys up, down
Selecting (starting) applications in the open submenus of the Start bar	Enter
Starting the Windows NT Task Manager	Ctrl + Shift + Esc
Focusing within the Windows NT menus	Tab
Applying the „right mouse click„ to focused buttons	Shift + F10
Switching to different tabs within menus	Ctrl + Tab
Switching between started applications **Note:** special conditions apply to the Indramat user interface „WIN-HMI“. for further information, refer to the user description entitled „Standard user interface WIN-HMI for automated production“, DOK-MTC200-WIN-HMI*V02-FKB2-DE-E“.	Alt + Tab

4 Windows NT and function interface installation

4.1 The Windows NT operating system

Use of the Windows NT operating system and the possibility of running different applications in parallel call for powerful computer hardware.

The hardware requirements depend directly on the number and size of applications running in parallel on the PC. This must already be taken into account in the project planning phase. Network cards used and their drivers require a great amount of computer performance, which might be lacking elsewhere. Highly precise planning is imperative when it comes to the choice of hardware.

Note: For the Windows NT operating system, we recommend a PC with a Pentium processor and at least 32 MB of RAM and at least 500 MB hard disk capacity.

Multitasking and Windows NT

While the so-called „cooperative“ or non-preemptive“ multitasking still had control over several started applications in 3.1x, genuine „preemptive“ multitasking has been realised in Windows NT.

Non-preemptive multitasking

It is not the operating system, but the application itself which decides how much computing time is to be exploited and when this is to be briefly granted to the operating system. The drawback here is that, when several applications are running in parallel, only limited working in parallel is possible.

Preemptive multitasking

The operating system itself decides how much computing time is assigned to the individual applications. Switching between applications is more fluid because the operating system is able to distribute computing time faster and at shorter intervals, thus creating the impression that several instructions are able to run „simultaneously and without restriction“.

Windows NT Task Manager

Executed applications can be monitored and controlled with the Task Manager, i.e. started applications can be brought to the front of the desktop or can be ended. It is also possible to start applications or to switch to others.

Calling up the Task Manager

- ⇨ By means of the Ctrl + Shift + Esc keystroke combination
- ⇨ Right mouse click on the Task bar

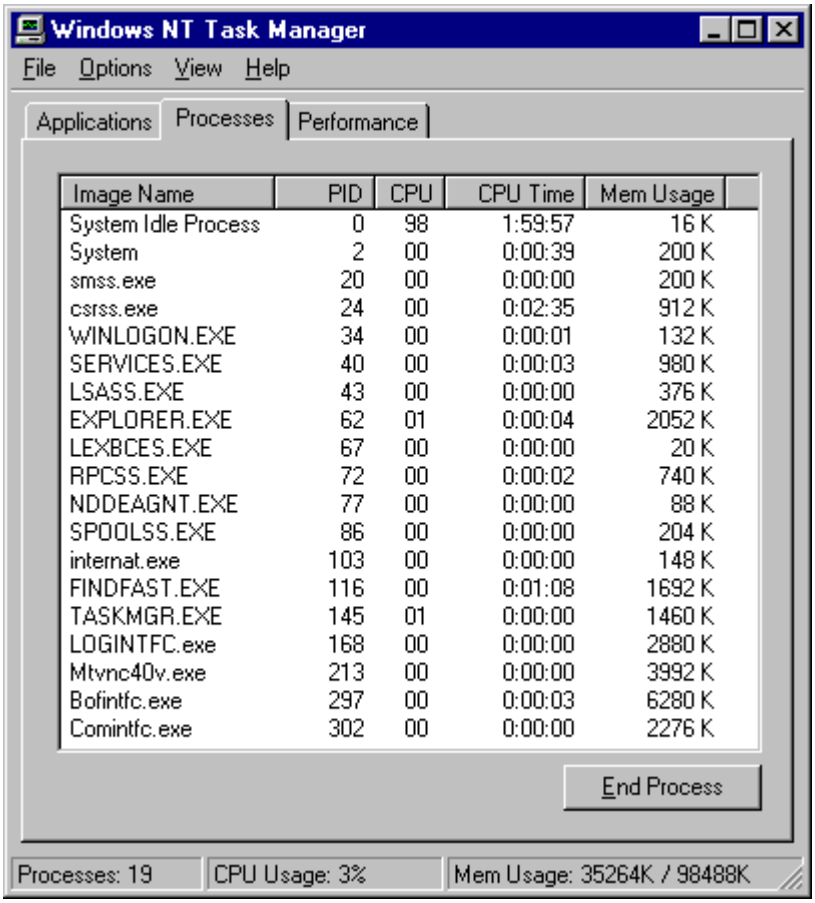


Fig. 4-1: Windows NT Task Manager

Note: With the Alt + Tab keystroke combination, applications can be brought up front on the desktop without a selection having to be made via the Task Manager.

4.2 NT system properties settings

Features

To arrive at an optimum reaction time for the function interface, performance boosting for the application in the foreground must be set to „No increase“.

Note: Reliable and error-free operation of the function interface is only guaranteed if performance boosting for the application in the foreground is set back to „No increase“.

Proceed as follows to set this:

⇒ Click Start, point to Settings, then to Control Panel and finally System.

⇒ Click the Features tab and set the slider to „No improvement“.

⇒ Then click the OK button.

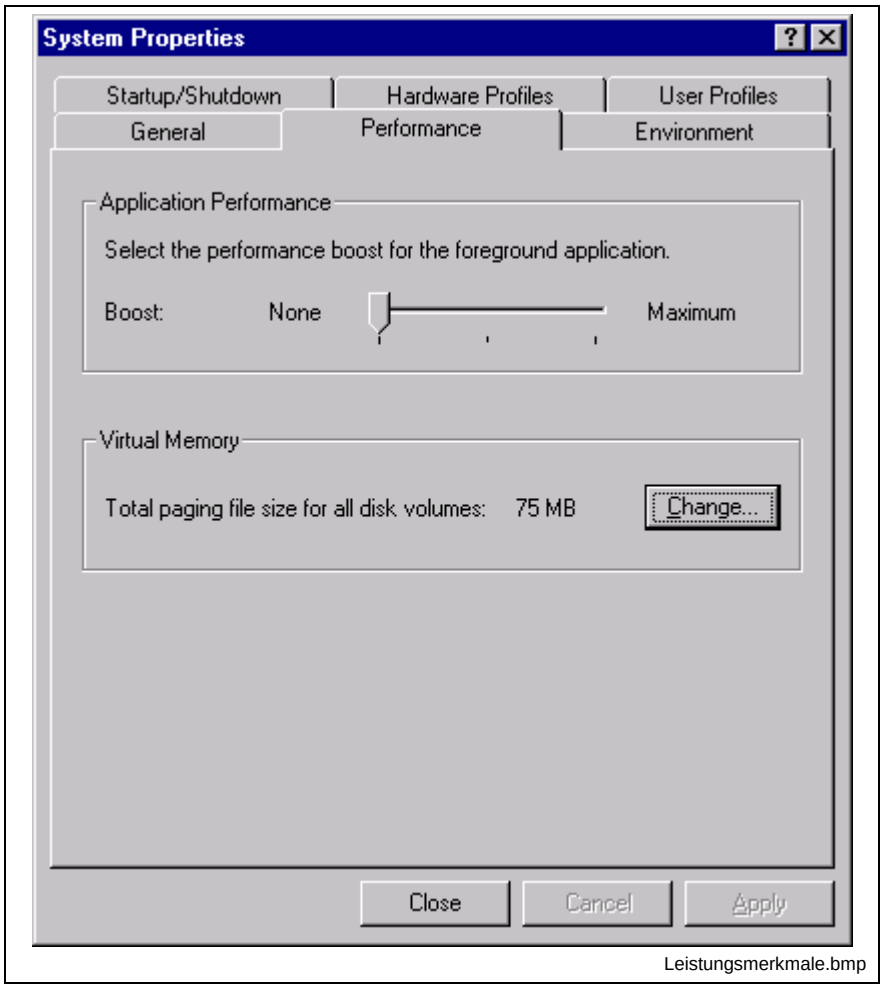


Fig. 4-2: Setting the Windows NT system property „Features“

Note: The settings for the „Virtual Memory“ may differ from your system settings.

Properties of date/time

For the exchange of BOF/GBO files between two PCs, it is necessary to ensure that the time zone settings on both PCs are identical. The automatic changeover between summer and winter time must also be deactivated.

Proceed as follows to set this:

- ⇨ Click Start, point to Settings, then to Control Panel and finally Date/Time.
- ⇨ Click the Time Zone tab and deactivate the „Automatically adjust clock for daylight saving changes“ toggle button.
- ⇨ Then click the OK button.

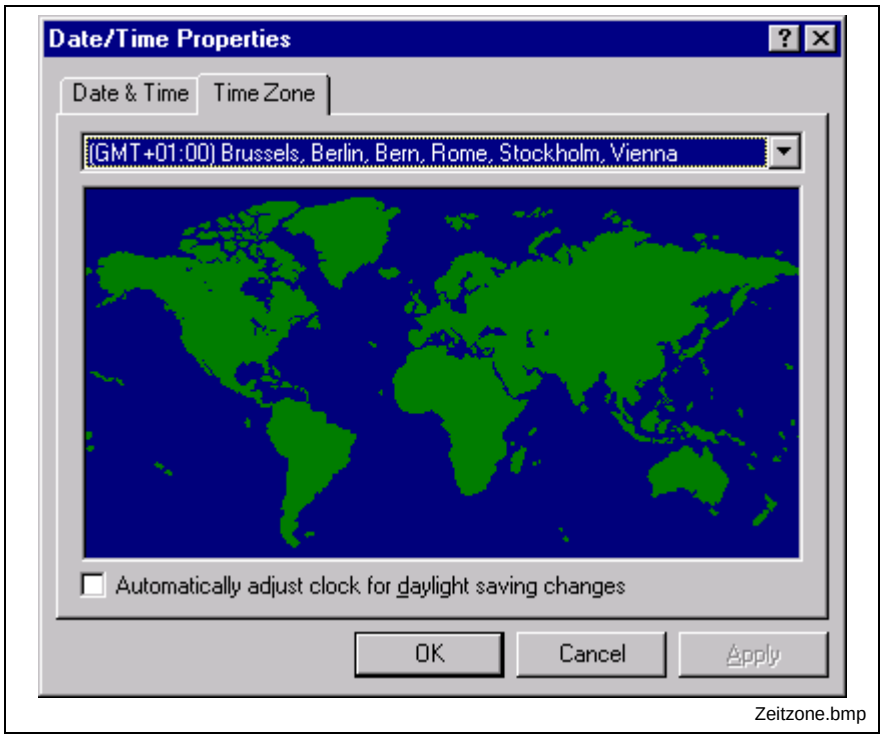


Fig. 4-3: Properties of Date/time

Environment

For use of the specific functions of the Indramat function interface, the individual applications must be informed about the directory structure of the DLLs. On the „environment“ tab, the installation program adds the path entries [Drive]:\...\IND_DLL and C:\IND_BASE to the text box „User variables for ...“; [Drive] stands for the installation drive.

Note: The Path user variables are added by the default installation (see chapter „Installing the function interface„), i.e. by the „C:\IND_BASE“ and „C:\MT-CNC\IND_DLL“ entries.

Proceed as follows to check these entries:

- ⇨ Click Start, point to Settings, then to Control Panel and finally System.
- ⇨ Click the Environment tab and then Path.
- ⇨ Then click the Cancel button.

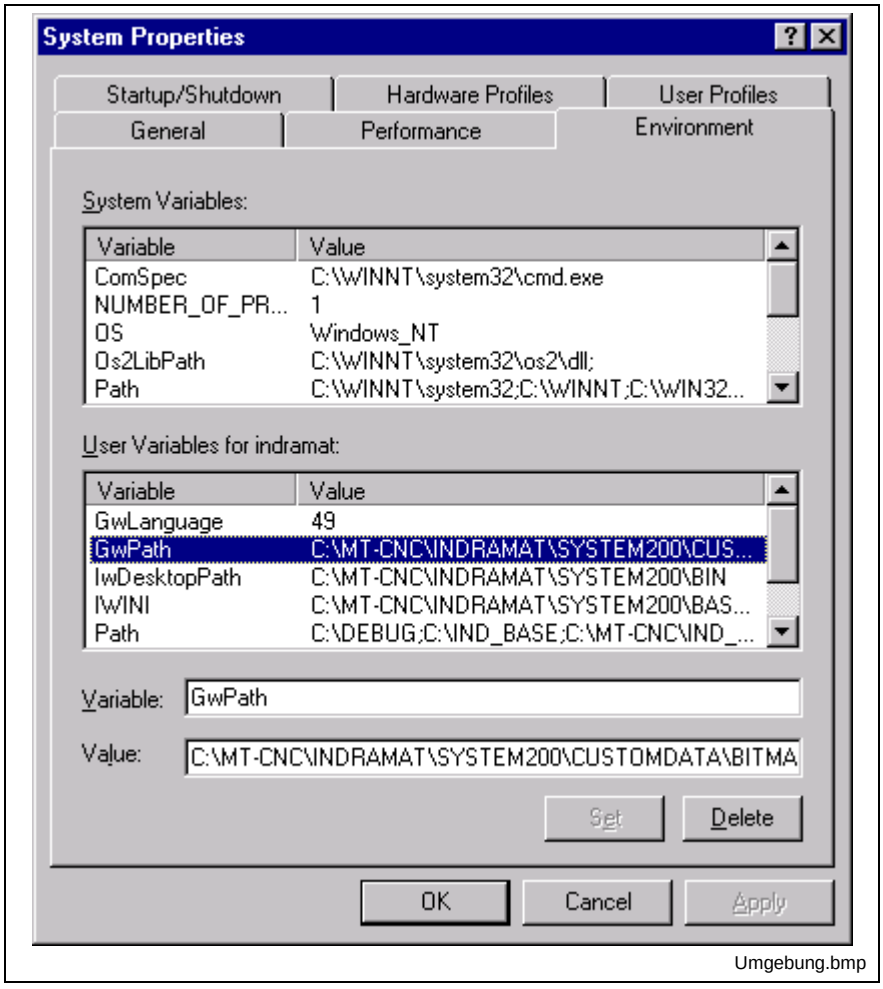


Fig. 4-4: Setting the windows NT „Environment“ system feature

Note: The other entries may differ from your system entries or from the user profile entries.

Paging the kernel mode drivers

The Windows NT operating system swaps parts of the kernel mode drivers to the page file in the event of intensive use of memory. This may have a highly negative impact on the performance of the overall system. Windows NT allows you to suppress paging of kernel mode drivers.

Note: This is set by the installation program of the function interface.

You can make manual changes by calling up the registry editor „REGEDT32“.

⇨ Click Start and then the Run option.

⇨ Enter „REGEDT32.EXE“ in the text box.

⇨ Then click the OK button and the registry editor is started.

The setting is modified by way of the key:

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session-Manager\MemoryManagement

Here, the installation program sets the value of the „DisablePagingExecutive“ variable to „1“. As the result of this modification, the system code is held in the RAM and is not paged. On installation of the Windows NT operating system, this variable's value was set to „0“ to permit temporary paging.

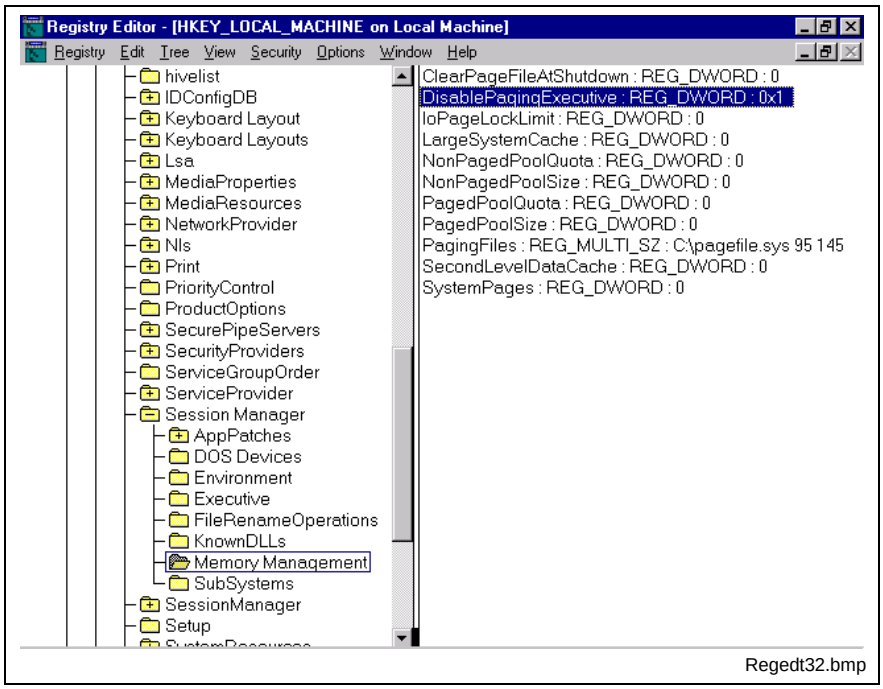


Fig. 4-5: Registry editor: „DisablePagingExecutive“ variable

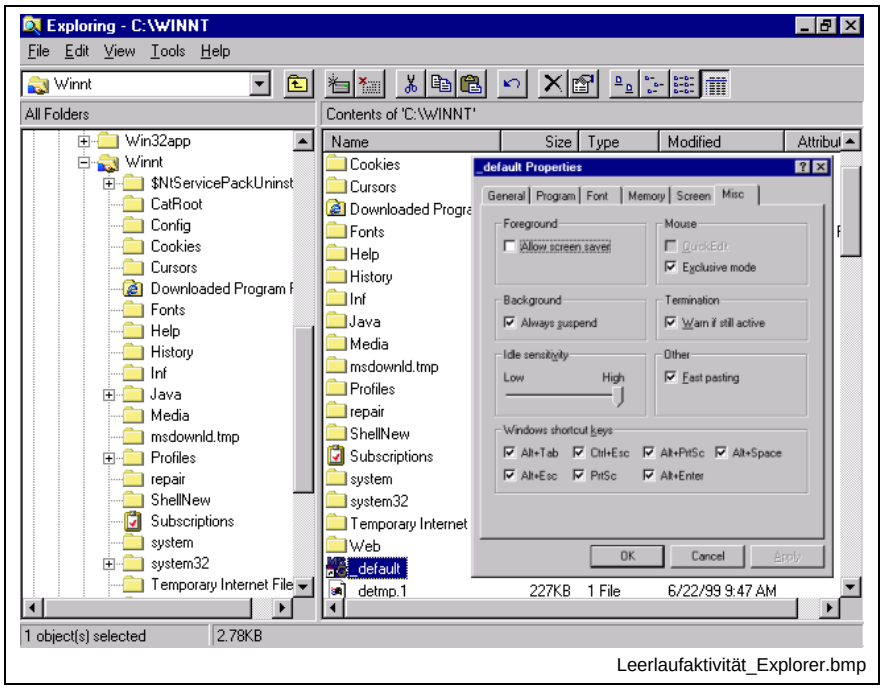
Note: Error-free operation of the function interface is only guaranteed if the „DisablePagingExecutive“ parameter is set to the value „1“.

Idle activity

To arrive at more efficient use of the CPU resources when a foreground program (DOS box) is idle, the idle activity of the „_default.pif“ file must be set to „High“. With this setting, a background program is assigned less CPU resources, i.e. when the DOS box is idle, as much CPU performance as possible is passed on to other active applications.

Proceed as follows to set this:

- ⇨ Open the windows NT Explorer: click Start, point to Programs and then click the Windows NT Explorer.
- ⇨ Go via the Winnt folder
- ⇨ Click the „_default.pif“ file and open the Properties dialog via the Explorer's File menu.
- ⇨ Click the Special tab and set the idle activity slider to „High“.
- ⇨ Then click the OK button.



Leerlaufaktivität_Explorer.bmp

Fig. 4-6: Properties of the „_default.pif“ file: idle activity

Note: The other settings may differ from the settings of your system or your user profile.

4.3 Installing the Windows NT kernel-mode driver

The Windows NT kernel-mode driver „MTCNC00I.SYS“ establishes the link between the function interface and a serial interface or dual port RAM etc. (see chapter entitled „The structure of the function interface „).

Note: This driver is installed by the installation program of the function interface.

You can install it manually by calling up the registry editor „REGINI“ by way of the windows NT command prompt .

- ⇨ Click Start and point to Programs.
- ⇨ Click Command prompt.
- ⇨ Enter „cd winnt\system32\drivers“ to switch to the driver directory (see chapter entitled „Contents of the driver directory „[Drive]:\Winnt\System32\Drivers\“)
- ⇨ Then confirm your input by pressing the Return key.
- ⇨ Enter „regini mtcnc00i.ini“ to start the registry editor and to install the Windows NT kernel-mode driver.
- ⇨ Then confirm your input by pressing the Return key.
- ⇨ Enter Exit and confirm your input by pressing the Return key to close the command prompt.

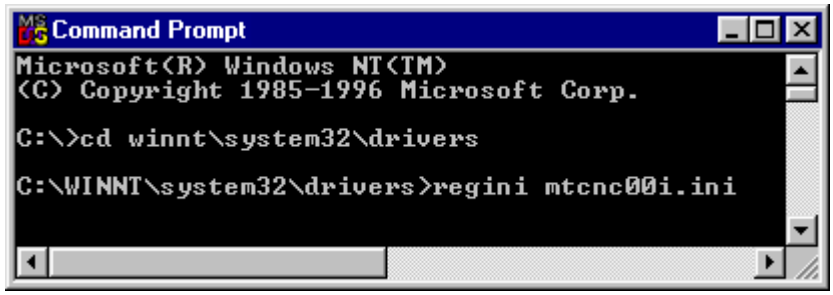


Fig. 4-7: Command prompt: installing the kernel-mode driver

Note: After installation of the Windows NT kernel-mode driver, you must restart Windows NT or your computer.

Proceed as follows to check whether installation has been completed successfully:

- ⇨ Click Start, point to Settings and then to Control Panel.
- ⇨ Then click Hardware.

After installation, the Windows NT kernel-mode driver „MTCNC00I.SYS“ is listed as the device „mtcnc00i“ with the status „Started“ and the start mode „Automatic“.

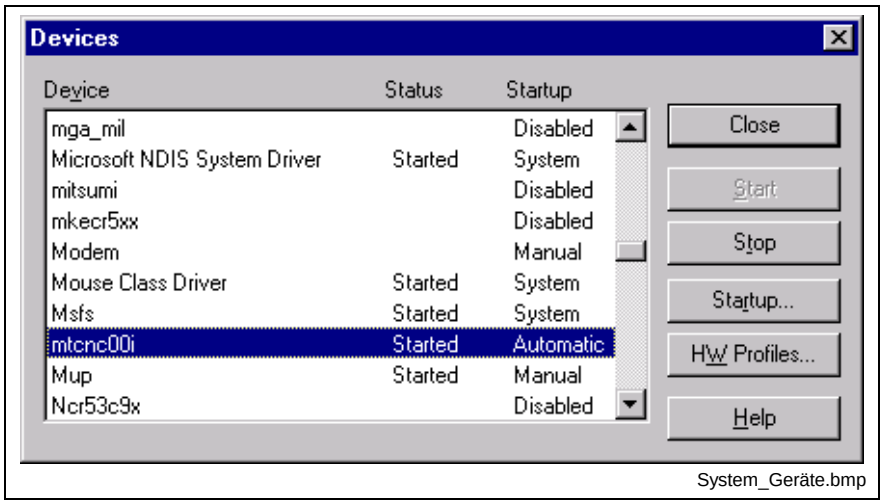


Fig. 4-8: Control Panel: device „MTCNC00I“

4.4 Installing the function interface

CAUTION

It is the responsibility of the user to archive all user data before running installation. Attention must also be paid to making sure that only ever one version of the function interface is installed on the target system.

Installation from CD-ROM

⇨ To install the function interface, insert the CD-ROM marked:

- SWD-CONTRL-FUN-04VRS-MS-CD600

in the appropriate drive (e.g. drive D) of your PC and installation will be run automatically.

Installation from diskettes

⇨ When installing from diskettes, insert the 1st diskette marked:

- SWD-CONTRL-FUN-04VRS-MS-C1,44

in the appropriate drive (e.g. drive A) of your PC.

Note: In the rest of the description, we assume that drive A is being used in the event of installation from diskette.

Proceed as follows to start installation of the function interface:

⇨ Click Start and then the Run option.

⇨ Enter „A:\Setup.exe“ in the text box area.

⇨ Then click the OK button and installation of the function interface will be started.

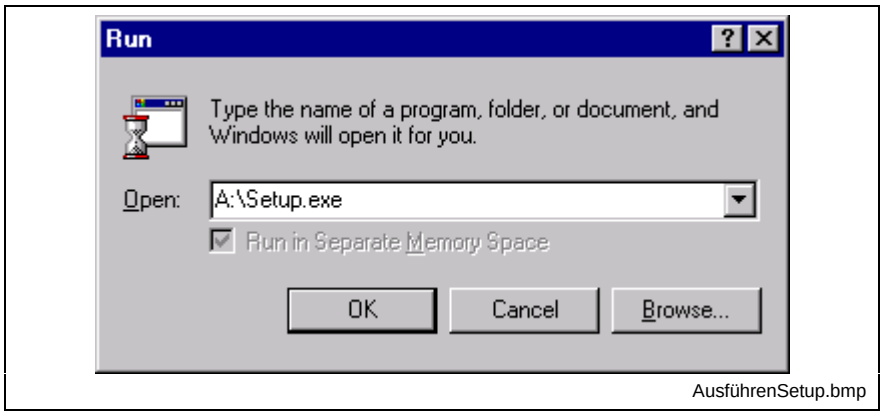


Fig. 4-9: Windows NT „Run“ dialog box: diskette installation

In the first dialog, you can select the language of the InstallShield wizard.

⇒ To do this, click the arrow on the right of the default setting „English“ and select the required language of the installation program by clicking it.

⇒ Then click the OK button.

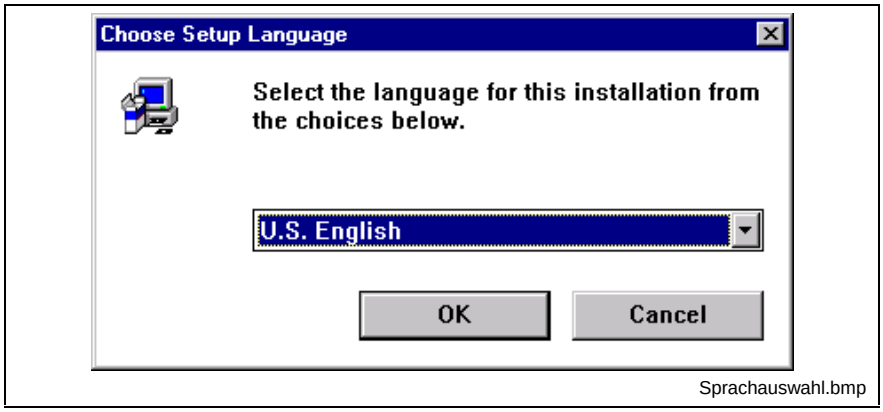


Fig. 4-10: Selecting the language of the InstallShield wizard

After the language has been selected, the main screen of the installation program appears, in which a window containing copyright information and the version of the installation program are displayed for about 3 seconds.

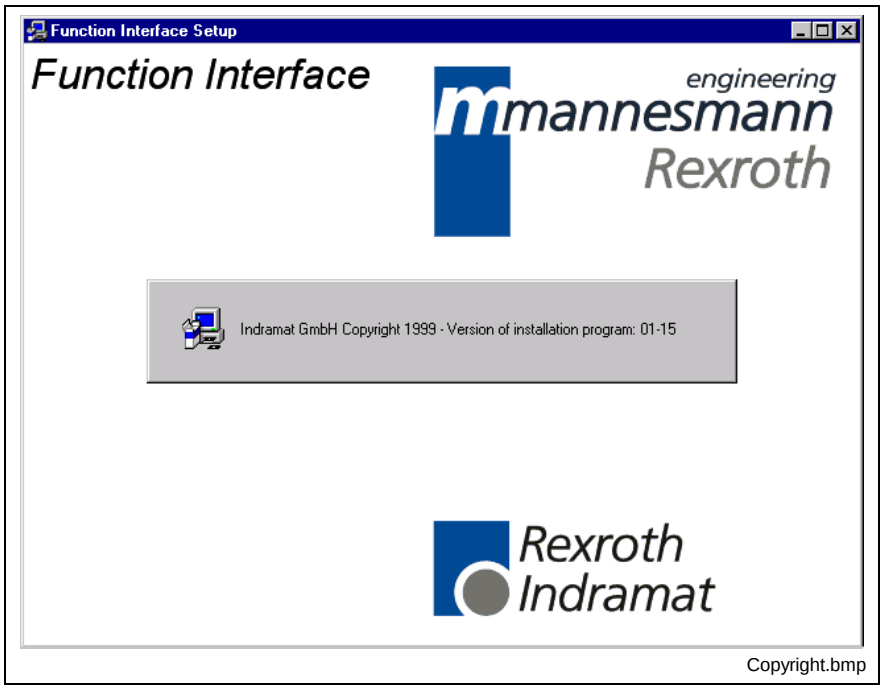


Fig. 4-11: Copyright information and version of the installation program

A window then appears, welcoming you to installation of the function interface.

Note: Please observe the instructions given here. The installation program can be cancelled by clicking the Cancel button.

⇨ Click the Continue button to continue the installation program.

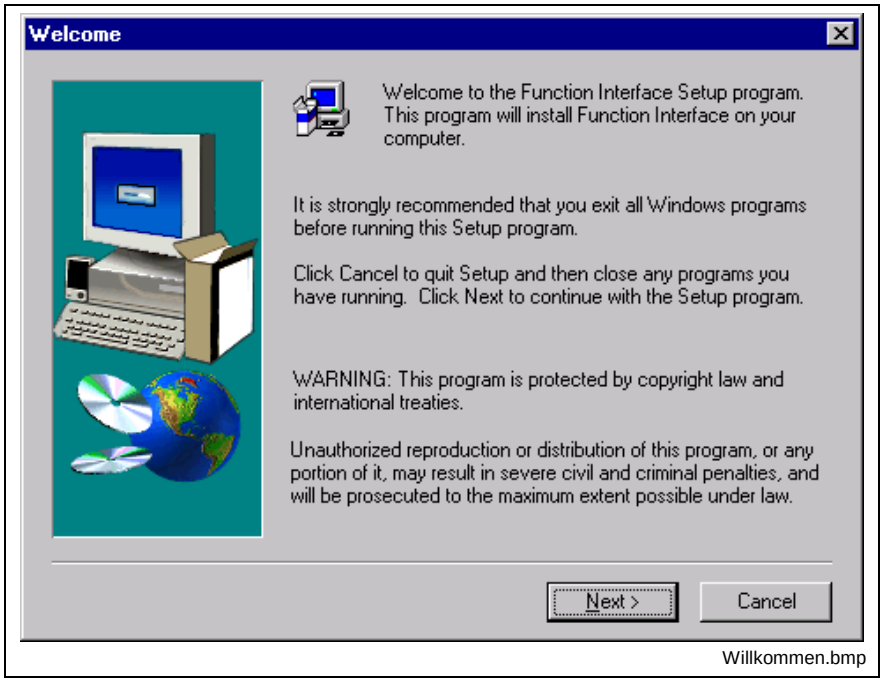


Fig. 4-12: Welcome screen of the installation pro

The software licence agreement is displayed in the next window. You can display the complete licence agreement by pressing the Down key (cursor key).

Note: To install the function interface, you must consent to this licence agreement. Installation is cancelled if you click the No button. You can return to the previous window by clicking the Back button.

⇨ Click the Yes button if you consent to all conditions of the software licence agreement.

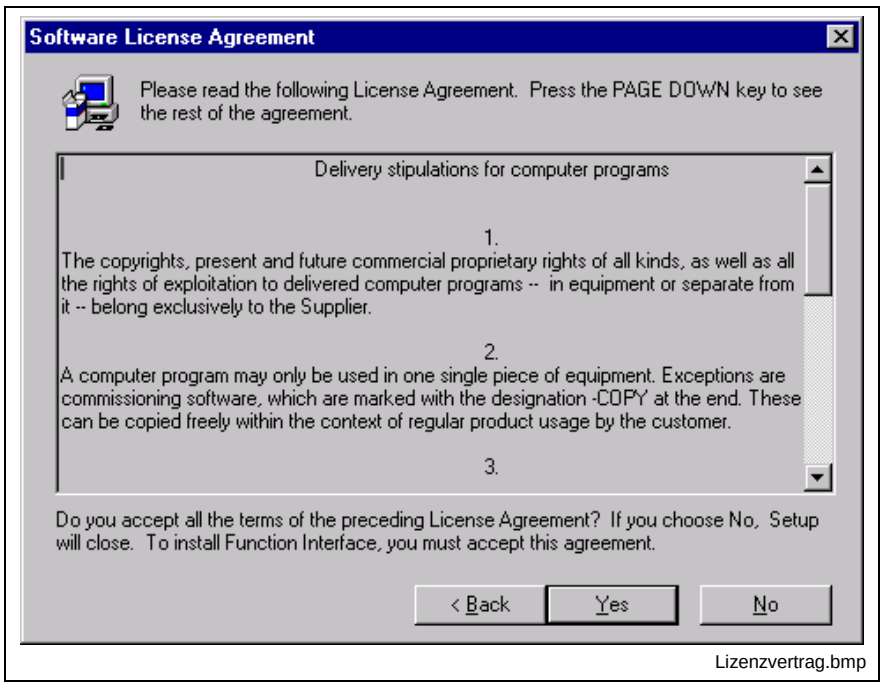


Fig. 4-13: Conditions of the software licence agreement

In the window that follows, you are prompted to enter user information.

⇨ Enter the user's name, company and serial number in the appropriate text box areas.

Note: For details of the serial number, refer to the label on the diskette set or the CD-ROM. You can cancel installation by clicking the Cancel button. You can return to the previous window by clicking the Back button.

⇨ Then click the Continue button to continue the installation program.

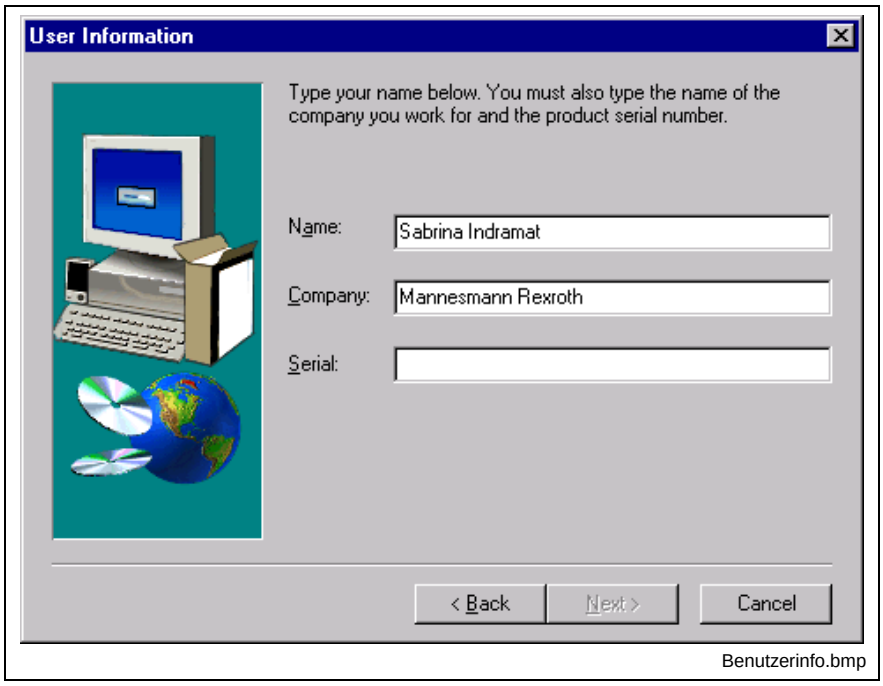


Fig. 4-14: User information input

In the window that follows, you can select the target path. In the default installation process, the target path „C:\MT-CNC“ is proposed.

Note: you are advised to choose this setting, possibly on a different drive. **Pay attention to the fact that only one version of the function interface is ever installed on the target system.** To modify the default path, click the Search button. The select the corresponding drive and directory.

⇨ You can continue installation by clicking the Continue button.

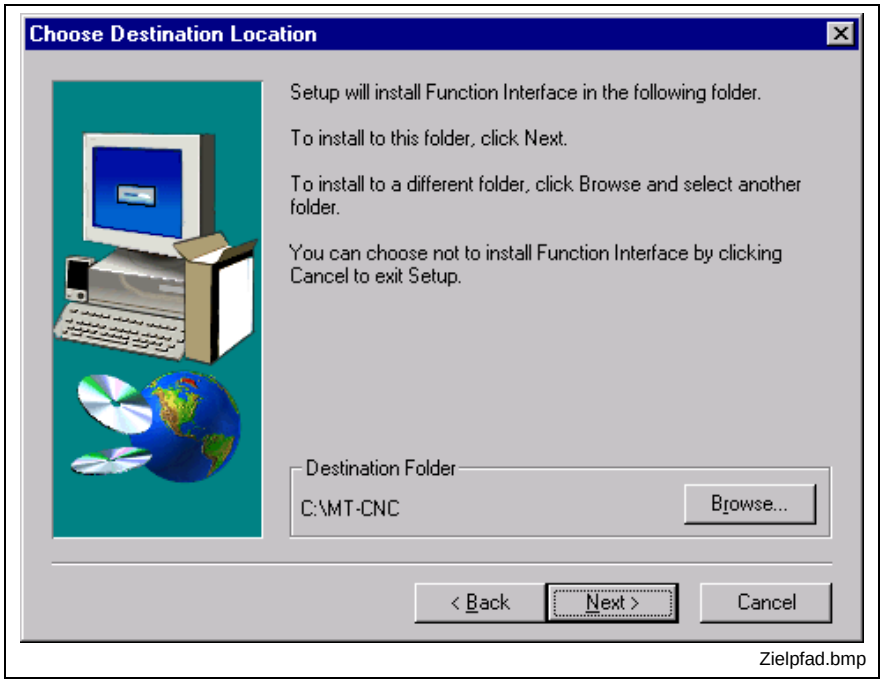
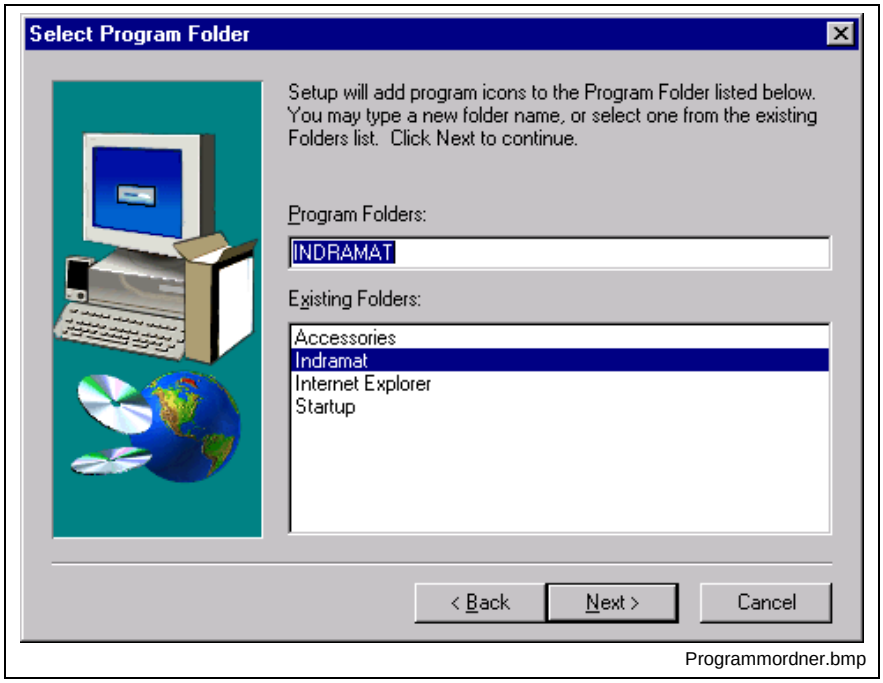


Fig. 4-15: Selecting the installation target path

In the next installation step, new icons are assigned to the existing program folders. However, you also have a possibility of creating new program folders here.

⇨ To add icons to an existing program folder, click the folder. Otherwise, enter the name of the new program folder in the „Program folder:“ text box area.

⇨ Installation is continued when you click the Continue button.



Programmordner.bmp

Fig. 4-16: Selecting the program folder

In the following window, you are prompted to enter the release code for the PRO version.

Note: The PRO version contains the necessary configuration software for the development of user programs in the Visual C/C++ and Visual Basic programming language as well as example programs for the relevant programming languages. The release code for the PRO version can be ordered from your local sales office by quoting the **type designation: SWS-FUN*PC-PRO-04VRS-MS**.

⇨ Enter the release code for the PRO version in the text box area with the name „Source code for interface programming“.

⇨ Then click the Continue button to continue installation.

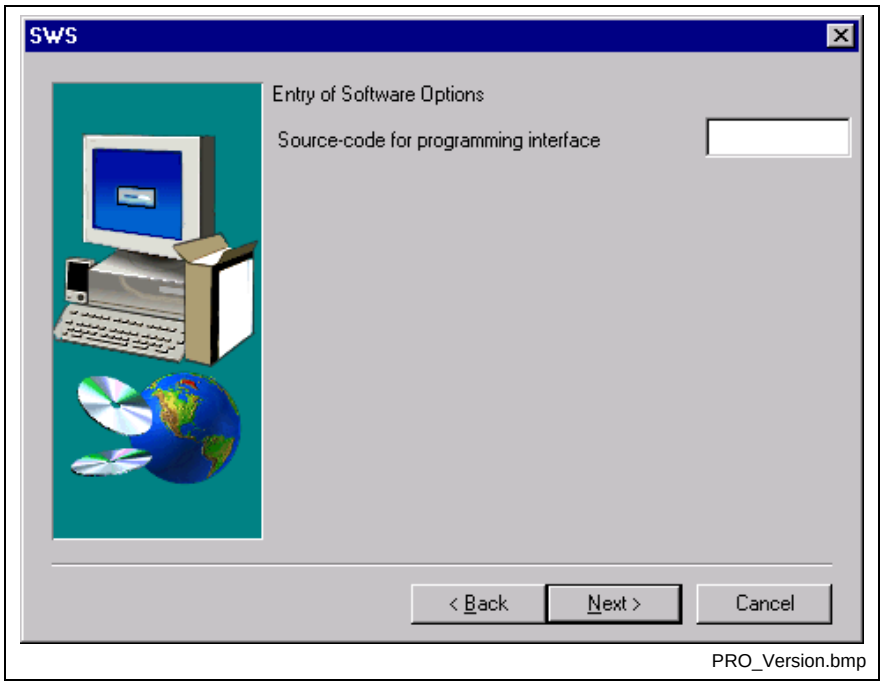


Fig. 4-17: Releasing the configuration software: PRO version

In the next installation step you see the settings that you have made for installation. If you would like to modify the current settings, click the Back button to return to the previous installation steps.

⇒ Installation is cancelled when you click the Cancel button.

⇒ Click the Continue button if you wish to retain the current settings and start copying.

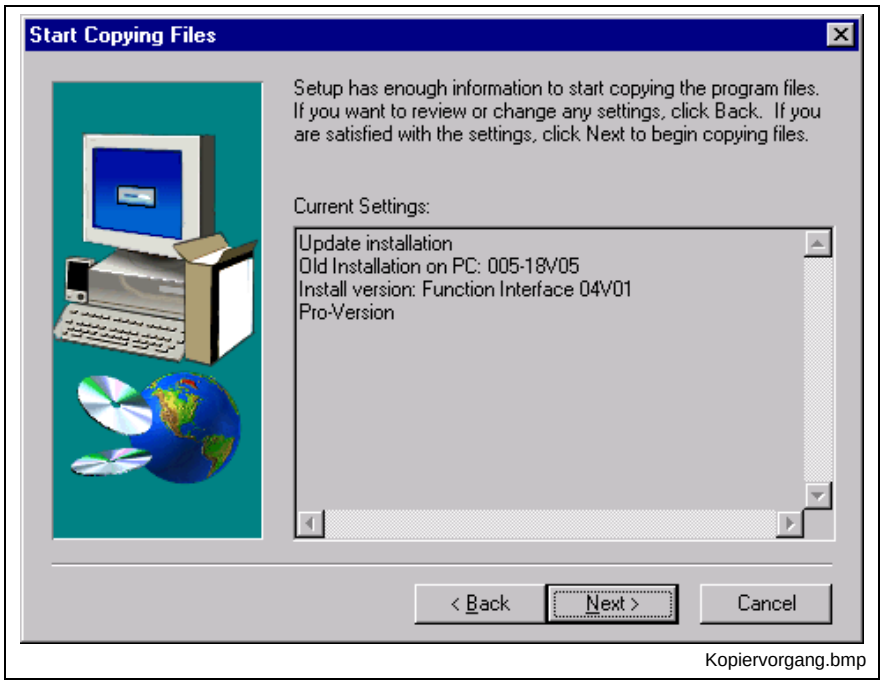
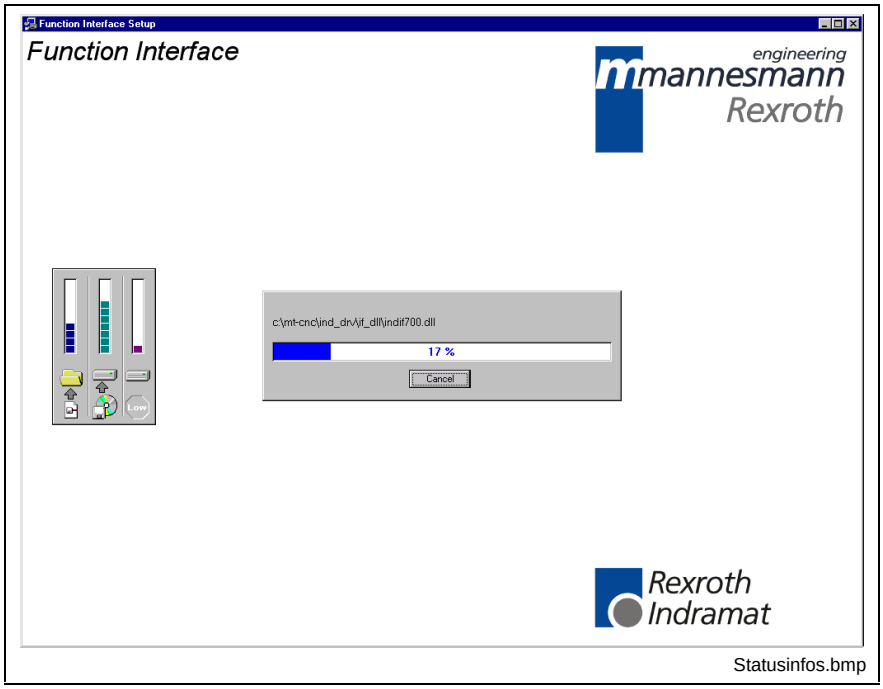


Fig. 4-18: Confirmation window for starting copying

The following figure shows the copy process, which may take a few minutes. In the left area you will see three bars to indicate the installation

status of the currently copied function interface component as well as the copy status of the inserted medium (diskette or CD-ROM) and the free hard disk capacity. The installation program will be cancelled and a relevant message will be displayed if the capacity of your hard disk is not sufficient for installation.

The status window in the middle shows the progress of the overall installation as a percentage by means of a bar.



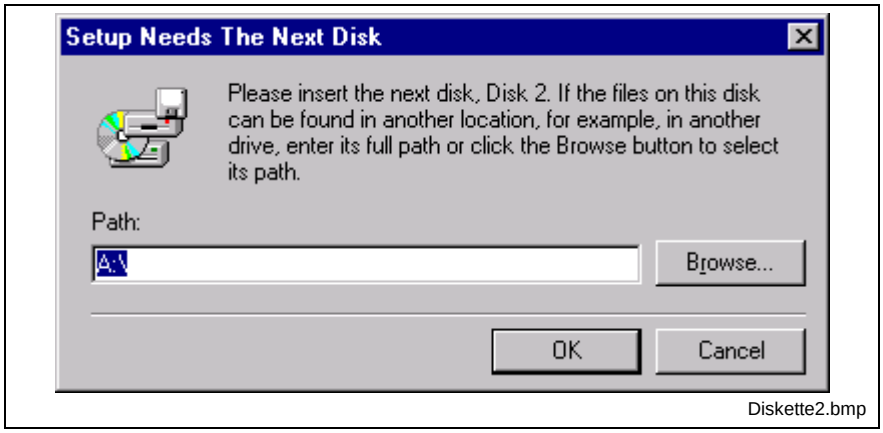
Statusinfos.bmp

Fig. 4-19: Copy status information

Note: Installation can be cancelled by clicking the <Cancel> button.

When installing from diskettes, you are prompted to insert further diskettes.

⇨ Insert the corresponding diskette in the drive and click the OK button to continue installation.



Diskette2.bmp

Fig. 4-20: Inserting next diskette

Once copying has been completed, you have a possibility of creating icons for installed applications on the desktop.

⇨ Click the Yes button if you wish to install icons on the desktop.

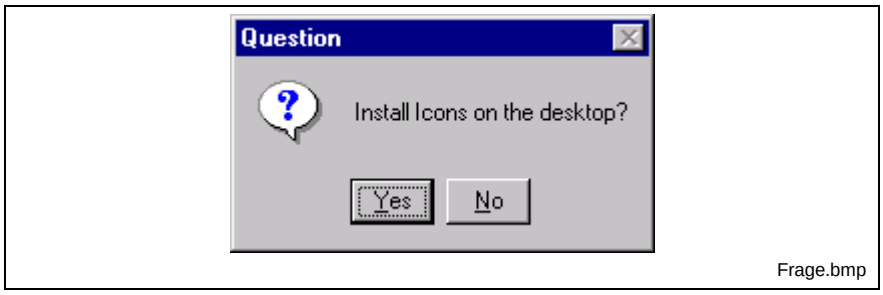


Fig. 4-21: Installing application icons on the desktop

A message appears to tell you that installation has been completed and that the computer must be restarted.

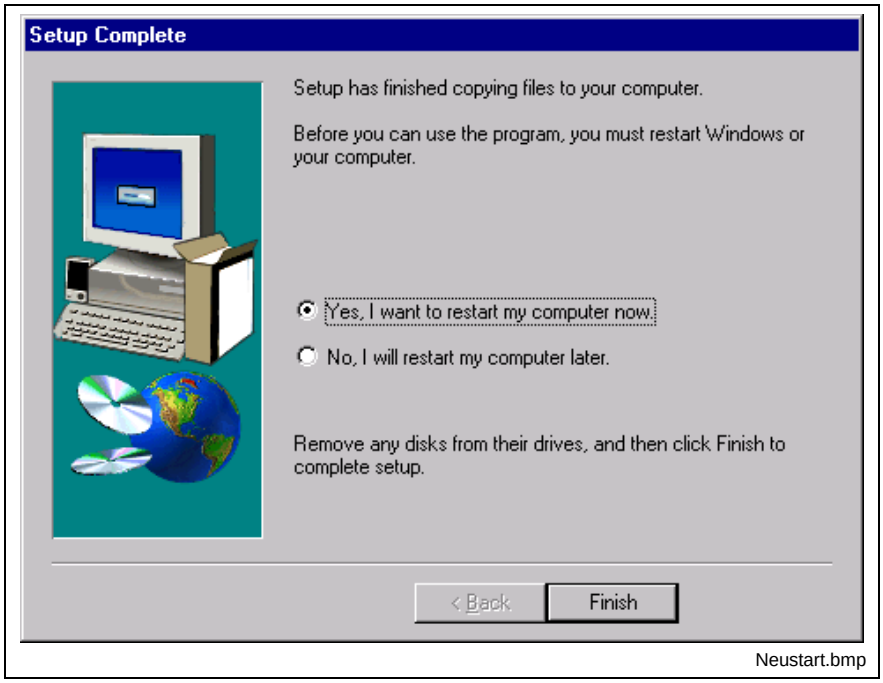


Fig. 4-22: Setup completed: restart the computer

Note: You must restart windows NT or the computer before you can work with the function interface.

4.5 Directory and file structure of the function interface

Contents of the „INDRAMAT.INI“ file

The global settings for the function interface are defined in this file. The function interface searches for the „INDRAMAT.INI“ file in the „C:\IND_BASE“ directory. The default setting refers to the [Drive]:\MT-CNC\IND_DRV path, and the name of the subdirectory \IND_DRV is fixed.

The preceding directories and the drive [Drive] can be chosen freely, though. The „INDRAMAT.INI“ file conforms to the Microsoft Windows INI standard and has the following structure:

Designator	Values	Explanation
[IfConfig]	The configuration settings for the function interface are located here	
IfInstDir=	e.g.: C:\MT-CNC\IND_DRV	Directory in which the three basic processes of the function interface are installed. The entry is set by the installation program.
AndInstDir=	e.g..: C:\Andron	!Optional! Directory for Andron controller software. The entry refers to the „ANDRON.EXE“ application.
IfDllMode=	e.g.: 04.10 03.xx [00..70], 04.xx [00,10]	The mode that is to be supported by he function interface is specified. The IfDllMode of a more recent function interface version can be used, for instance, for troubleshooting in the mode of the predecessor version.
IfVersion=	e.g.: 04V01	Current version of the function interface.
GBOVERSION=	e.g.: 005-18V05	Current version of the Indramat user interface (BOF/GBO).
INDRAMAT_x=	x=1....9 File designation	Reference to the C:\IND_BASE directory. The existence of the file designations specified here is checked when the function interface is started. Meanings: File designation without extension is a DLL. e.g. Indramat = INDRAMT.DLL Several file designators are separated by a „comma“.
IND_DLL_x=	x=1...9 File designation	Reference to the [Drive]:\....\IND_DLL directory. The existence of the file designations specified here is checked when the function interface is started. Meanings: File designation without extension is a DLL. e.g. INDFS100 = INDFS100.DLL. Several file designators are separated by a „comma“.
IF_DLL_x=	x=1...9 File designation	Reference to the [Drive]:\....\IND_DRV\IF_DLL directory. The existence of the file designations specified here is checked when the function interface is started. Meanings: File designation without extension is a DLL. e.g. INDIF000 = INDIF000.DLL Several file designators are separated by a „comma“.
[Install]	Entries concerning the installed System200 software components are located here.	
HMIVersion=	e.g.: 01V05	Version identifier of the System200 software component WIN-HMI
TYP=	e.g.: HMI	System200 software component WIN-HMI
ServicePack=	e.g.: 2 [1,2,...]	Identifier of the ServicePack of the installed System200 software component

Example entries in the „INDRAMT.INI“ file

```
[IfConfig]
IfInstDir=C:\MT-CNC\IND_DRV
AndInstDir=C:\Andron
IfDllMode=04.10
IFVERSION=04V01
GBOVERSION=005-18V05
INDRAMAT_1=indramat
IND_DLL_1=indfs100,indma110,indma900,indut140,indof160
IF_DLL_1=indif000,indif120,indif130,Indif150
IF_DLL_2=indif200,indif210,indif220,indif300,indif310,indif320
IF_DLL_3=indif330,indif340,indif350,indif360,indif400
IF_DLL_4=indif500,indif510,indif520,indif530,indif540,indif550
IF_DLL_5=indif600,indif610,indif700,indif810,indif820,indif840
IF_DLL_6=indifA00

[Install]
HMIVersion=01V05
TYP=HMI
```

The DLL entries (If_DLL_1,...) can be extended up to the ninth entry (If_DLL_9). The presence of the DLLs is only checked if they have been entered beforehand next to the relevant parameters. If a file name is specified without an extension, the „DLL“ extension is assumed automatically. If the presence of a different file is to be checked, its file extension must also be specified, e.g. „user_program.dat“.

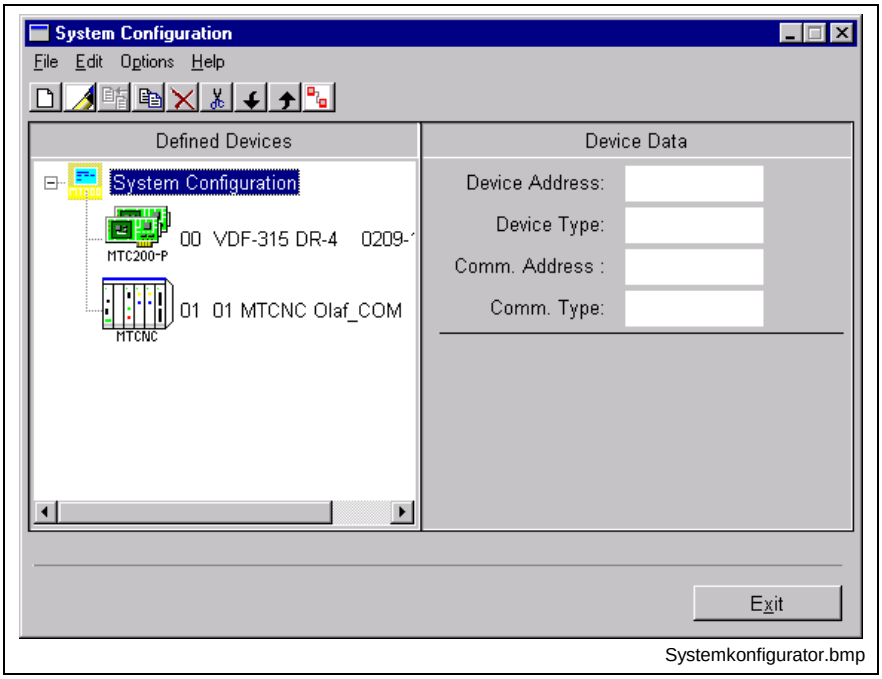
Contents of the „IND_DEV.INI“ file

The configuration of the individual communication addresses and the settings of the various Indramat devices are defined in this file. The „IND_DEV.INI“ file is edited by the system configurator and is located in the „[Drive]\...\CONFIG\“ directory.

Indramat system configurator

The Indramat system configurator is an editor which is used to define and list the devices connected to the controller PC. The device addresses, the device type and the description of the communication path to the device are used for this purpose. The aim is to create a 1:! image of the device structure connected to the controller PC, which is referred to as the system configuration.

The system configuration is stored on the controller PC. Properties on which the devices are based can also be assigned, for example the allocation of a device of the type MTVNC (virtual MTC for the „offline simulation“ function) to a device of the type MTC200-P or MTCNC etc. to form a simulation pair. In this case, parameter sets of the real devices, for example, can be used by the virtual device so as to be able to start a simulation of NC programs.



Systemkonfigurator.bmp

Fig. 4-23: Indramat system configurator

The „IND_DEV.INI“ file conforms to the Microsoft Windows INI standard and has the following structure:

Designator	**Values**	**Explanation**
[CommAddrX]	X = 1...8	Allocation of the communication channel (thread) of the function interface
CommStr=	V24, Port [COM1...4], Baudrate, Parity, Type of interface, Packet Counter - or - DMA, Address, Offset, Length - or - SHM, Channel No. [1...15]	Communication via serial RS232 interface, e.g. V24,COM1,19200,NONE,RS232,TCON Communication via serial RS485 interface, e.g. V24,COM2,19200,NONE,R485H,TCON In the case of the MTC200-P, a DMA channel is needed for communication via a dual port RAM, e.g. DMA,$D000,$0000,$2000 Communication channel to the MTVNC via a shared memory, e.g. SHM, 1
Timeout=	>= 1000 [msec] Default = 3500 [msec]	! OPTIONAL ! Time after which a response must be available from the device.
- or - only in the case of dual port RAM (with reference to the example of a MTC200-P)		
CommStr=	DPR, Address [$C000,$0000 ... $FE00,$0000], Length, RAM0, Packet-Counter	Communication via dual port RAM, e.g. MTC200-P DPR, $D000,$0000,$2000,RAM0,TCOFF.
PortAddr=	$200, $204, ... $31C e.g. $31C	Address of the MTC/MTS card in accordance with the setting on the respective card.
PortVal=	$20, $21, ... $3F e.g. $28 if address [$D000,$0000]	Configuration byte for setting the physical memory address of the MTC/MTS card.
Timeout=	>= 1000 [msec] Default = 3500 [msec]	! OPTIONAL ! Time after which a response must be available from the device.

Designator	**Values**	**Explanation**
[DeviceAddrX]	X = 0...15	Device address
DeviceName=	max. 32 ASCII characters	Device designation; e.g. machining centre 12T34
DeviceTyp=	e.g. MTC200-P MTVNC MTCNC MTC200-P MTC200-R ISP200-P ISP200-R TRA200-P TRA200-R ECODRIVE03 MTA200-P SERCANS-A SERCANS-P	Device type: MTC with PLC PC variant Virtual MTC MT-CNC 02 / 03 MTC with PLC PC variant MTC with PLC RECO variant Standalone-PLC PC variant Standalone-PLC RECO variant TRANS200 PC variant TRANS200 RECO variant Ecodrive03 ANDRON controller SERCANS-A card (via serial interface) SERCANS-P card (card (via serial interface)
DeviceAssign=	0...15, NO	Allocation of a simulation pair. For example, the MTVNC is assigned to a real MTCNC here.
DeviceStatus=	ON, OFF	Allocation indicating whether the device is incorporated in the management structure of the function interface.
MtvncMode=	OFF, RUN, STANDBY	! Virtual MTC (MTVNC) only ! Status of the MTVNC in the event of inactive use.
MtvncMemory=	256, 257 .. 16383 [KB] Default = 512 [KB]	! Virtual MTC (MTVNC) only ! Size of the PC memory that is used by the MTVNC.
CommAddr=	1...8	Allocation of the communication address. Corresponds to the [CommAddr1...8] parameter.
PLC=	YES, NO	PLC support of the device, e.g. an MTVNC, TRANS200-R does not have a PLC and so the PLC=NO parameter is set.
[DeviceOrder]	The configuration settings for the system configurator are located here.	
Order=	0,1,2, ...15	Display sequence of configured devices.
[NetManager]	The configuration settings for the network drivers „NETINTFC.EXE“ are located here.	
NetManagerMode=	OFF, RUN	Results in starting of the network device driver.

Example entries in the „IND_DEV.INI“ file

Entry	Explanation
[CommAddr1] CommStr=DPR,$D000,$0000,$2000,RAM0,TCON PortAddr=$31C PortVal=$28	Communication address 1 Settings for communication via dual port RAM Port address of the MTC/MTS card Physical memory address of the MTC/MTS card.
[CommAddr2] CommStr=DMA,$D000,$0000,$2000	Communication address 2 DMA channel allocation.
[CommAddr3] CommStr=V24,COM1,19200,NONE,RS232,TCON	Communication address 3 Settings for communication via RS232.
[CommAddr4] CommStr=DPR,$D200,$0000,$2000,RAM0,TCON PortAddr=$318 PortVal=$29	Communication address 4 Settings for communication via dual port RAM Port address of the MTC/MTS card Physical memory address of the MTC/MTS card.
[CommAddr5] CommStr=SHM,1	Communication address 5 Settings for communication via Shared Memory.
[DeviceAddr0] CommAddr=1 DeviceAssign=NO DeviceName=VDF-315 DR-4 0209-15 DeviceStatus=ON DeviceTyp=MTC200-P PLC=YES	Device address 00 Allocated communication channel No MTVNC allocated Device designation Device is available and ready Device type Support of the PLC
[DeviceAddr1] CommAddr=3 DeviceAssign=3 DeviceName= machining centre 12T34 DeviceStatus=OFF DeviceTyp=MTCNC PLC=YES	Device address 01 Allocated communication channel Device address 03 (MTVNC) allocated (simulation pair) Device designation Device is not ready Device type Support of the PLC
[DeviceAddr2] CommAddr=4 DeviceAssign=NO DeviceName= machining centre 12T35 DeviceStatus=ON DeviceTyp=MTC200-P PLC=YES	Device address 02 Allocated communication channel No MTVNC allocated Device designation Device is available and ready Device type Support of the PLC
[DeviceAddr3] CommAddr=5 DeviceAssign=1 DeviceName= V-machining centre 12T34 DeviceStatus=ON DeviceTyp=MTVNC MtvncMemory=512 MtvncMode=RUN PLC=NO	Device address 03 Allocated communication channel Allocated to device address 01 (simulation pair) Device designation Device is available and ready Device type Size of PC memory Status of inactive use No support of the PLC
[DeviceAddr4] CommAddr=3 DeviceAssign=NO DeviceName= TRANS200 DeviceStatus=ON DeviceTyp=TRANS200-R PLC=NO	Device address 04 Allocated communication channel No MTVNC allocated Device designation Device is ready Device type No support of the PLC
[DeviceOrder] Order=3,0,1,2,4	Display sequence of the configured devices in the system configurator
[NetManager] NetManagerMode=RUN	Network driver is started

Contents of the system directory „[LW]:\Winnt\System32\“

The following Microsoft class libraries are placed in the system directory of Windows NT „[Drive]:\Winnt\System32\“:

File	Explanation
MFC30.DLL	Microsoft class libraries
MSVCRT20.DLL	Microsoft class libraries
MFC40.DLL	Microsoft class libraries
MFC42.DLL	Microsoft class libraries
MSVCRT40.DLL	Microsoft class libraries
MSVCRT.DLL	Microsoft class libraries
MSVCP50.DLL	Microsoft class libraries
MSVBVM50.DLL	Microsoft class libraries
COMCTL32.OCX	Dialog elements for Visual Basic applications
COMDLG32.OCX	Dialog elements for Visual Basic applications
REGSVR32.EXE	Application for registering the OCX files

Contents of the driver directory „[Drive]:\Winnt\System32\Drivers\“

The following files of the kernel-mode driver are placed in the driver directory of Windows NT „[Drive]:\Winnt\System32\Drivers\“:

File	Explanation
MTCNC00I.SYS	Windows NT kernel-mode driver
MTCNC00I.INI	Kernel-mode driver configuration file
REGINI.EXE	Application for registering the kernel-mode driver

Contents of the „C:\IND_BASE\“ directory

Indramat-specific files and function libraries that are needed for the starting and initialisation phase are placed in the „C:\IND_BASE\“ directory.

File	Explanation
INDRAMAT.INI	File containing global function interface settings
INDRAMAT.DLL	Accesses to global settings (GetInstPath, etc.)

Contents of the „[Drive]:\...\DOCUMENTATION\“ directory

As the result of standard installation, the drive and the path „[Drive]:\...\“ are set by default to „C:\MT-CNC\“. The following Windows 95/NT Help files from the English and German paper documentation are stored in the „C:\MT-CNC\DOCUMENTATION\“ directory:

File	Explanation
FI04VRSDE.HLP	Windows 95/NT Help file in German
FI04VRSDE.CNT	Definition file of the Windows 95/NT Help file
FI04VRSEN.HLP	Windows 95/NT Help file in German (in preparation)
FI04VRSEN.CNT	Definition file of the Windows 95/NT Help file

Contents of the „[Drive]:\...\SAMPLE\FI\VBDEMO\“ directory

As the result of standard installation, the drive and the path „[Drive]:\...\“ are set by default to „C:\MT-CNC\“. The files belonging to the example program for linking a Visual Basic application to the function interface are located in the „C:\MT-CNC\SAMPLE\FI\VBDEMO\“ directory.

File	Explanation
VBDEMO.EXE	Example program
The following Visual Basic source files of the „VBDemo.exe“ application are supplied with the configuration software. **SWS-FUN*PC-PRO-04VRS-MS (PRO version)**	
VBDEMO.VBP	Visual Basic project file
VBDEMO.VBW	Visual Basic project work area file
VBDEMO.FRX	Binary form file
VBDEMO.FRM	Form file
INDIF000.BAS	Function definitions of the individual routines; (see chapter entitled „Programming“)
SYSTHREAD.BAS	Basic module for the reaction to SYS messages
WIN32_API.BAS	Basic module for Windows API functions /subroutines

Contents of the „[Drive]:\...\SAMPLE\FI\VCDEMO\“ directory

As the result of standard installation, the drive and the path „[Drive]:\...\“ are set by default to „C:\MT-CNC\“. The „C:\MT-CNC\SAMPLE\FI\VCDEMO\“ directory contains the files for linking a Visual C/C++ application to the function interface.

File	Explanation
The following source files of the „VBDemo.exe“ application are supplied with the configuration software.**SWS-FUN*PC-PRO-04VRS-MS (PRO version)**	
INDIF000.C	Load procedure for the functions of the DLL.
INDRAMAT.C	Load procedure for the global functions of the DLL
INDIF000.H	Function definitions of the individual routines.
INDIFX00.H	Function definitions of the individual routines.

INDRAMAT.H	Global function interface routines

Contents of the „[Drive]:\...\IND_DLL\“ directory

As the result of standard installation, the drive and the path „[Drive]:\...\“ are set by default to „C:\MT-CNC\“. The following Indramat-specific function libraries are located in the „C:\MT-CNC\IND_DLL\“ directory:

File	Explanation
INDFS100.DLL	Editing the file identifier
INDMA900.DLL	Editing the MAP file
INDMA110.DLL	Linking the MAP file
INDUT140.DLL	Use of various system utilities
INDOF160.DLL	Use of various system utilities

Contents of the „[Drive]:\...\IND_DRV\“ directory

As the result of standard installation, the drive and the path „[Drive]:\...\“ are set by default to „C:\MT-CNC\“. The following files are located in the „C:\MT-CNC\IND_DRV\“ directory:

File	Explanation
MISX.DAT	Definition file for the MISX device group
MTCX.DAT	Definition file for the MTCX device group
MSCX.DAT	Definition file for the MSCX device group
MPCX.DAT	Definition file for the MPCX device group
MTAX.DAT	Definition file for the MTAX device group
MECX.DAT	Definition file for the MECX device group
MTRX.DAT	Definition file for the MTRX device group
VERSION.DAT	Definition file for the version identifier
COMINTFC.EXE	Communication process
BOFINTFC.EXE	User interface process
BOFINTFC.DAT	Definition file of the user interface process
LOGINTFC.EXE	Logic process
LOGINTFC.DAT	Definition file of the logic process
NETINTFC.EXE	Application for the client/server link
KILLTASK.EXE	Application for termination of function interface applications (see chapter entitled „Tips and tricks for working with the interface“)

Example entries in the „VERSION.DAT“ file

The version identifiers of the individual program portions and the version of the function interface are entered in the „VERSION.DAT“ file. This refers to all program portions (EXE, DLL) of the function interface. The following example shows the entries in this file:

Name	Build	Version	Date	Start parameters
IFVERSION	113	04V00	Feb 22	
INDRAMAT.DLL	113	04.01	Feb 15	
INDFS100.DLL	113	03.14	Feb 22	
INDIF300.DLL	113	03.63	Feb 16	
INDUT140.DLL	113	03.09	Feb 22	
INDIF310.DLL	113	03.32	Feb 16	
INDOF160.DLL	113	03.15	Feb 22	
INDIF200.DLL	113	03.71	Feb 22	
BOFINTFC.EXE	113	05.16	Feb 16	/b=w/c=t
INDIF360.DLL	113	03.07	Feb 03	
LOGINTFC.EXE	113	04.00	Feb 22	/c=t +G10
COMINTFC.EXE	113	04.00	Feb 22	/c=t +G10
INDIF210.DLL	113	04.00	Feb 22	
INDIF330.DLL	113	03.30	Feb 16	
INDIF540	113	03.01	Feb 22	
INDIF130.DLL	113	03.16	Feb 22	
INDIF810.DLL	113	04.00	Feb 03	
INDIF350.DLL	113	03.35	Feb 03	
INDIF320.DLL	113	03.28	Feb 03	
INDIF340.DLL	113	03.31	Feb 03	

Contents of the „[Drive]:\...\IND_DRV\IF_DLL\“ directory

As the result of standard installation, the drive and the path „[Drive]:\...\“ are set by default to „C:\MT-CNC\“. The following function libraries of the functin interface are located in the „C:\MT-CNC\IND_DRV\IF_DLL\“ directory:

File	Explanation
INDIF000.DLL	General functions for the user.
INDIF120.DLL	Functions for trace file output.
INDIF130.DLL	Functions for the user interface process.
INDIF150.DLL	Functions for the logic and communication process.
INDIF200.DLL	Functions for the logic and communication process.
INDIF210.DLL	Functions for the logic and communication process.
INDIF220.DLL	Functions for the logic and communication process.
INDIF300.DLL	Functions for the user interface process.
INDIF310.DLL	Functions for the user interface process.
INDIF320.DLL	Functions for the DOS - Windows NT links.
INDIF330.DLL	Functions for the user interface process.
INDIF340.DLL	Functions for the user interface process.
INDIF350.DLL	Functions for the NT - DOS links.
INDIF360.DLL	Functions for file accesses.
INDIF400.DLL	User interface process accesses to parameters.
INDIF500.DLL	Functions for access optimisation.
INDIF510.DLL	Functions for access optimisation.
INDIF520.DLL	Functions for access optimisation.
INDIF530.DLL	Functions for access optimisation.
INDIF540.DLL	Functions for access optimisation.
INDIF550.DLL	Functions for access optimisation.
INDIF600.DLL	Functions for accesses to NC programs.
INDIF610.DLL	Functions for accesses to NC programs.
INDIF700.DLL	Functions for accesses to NC compliers
INDIF810.DLL	Functions for accesses to Sercos parameters of the MTCX device group.
INDIF820.DLL	Functions for accesses to the MSCX device group.
INDIF840.DLL	Functions for accesses to the MTAX device group.
INDIFA00.DLL	Functions for HMI support of the MTCX device group.

5 Structure of function interface commands

5.1 Elements of the FI command

The function interface commands are subdivided into the following elements:

- Identifier,
- Selector and
- Data code.

Identifier

The identifier is composed of 11 ASCII characters for the device address, separators, interface designator, command, function code and function descriptor.

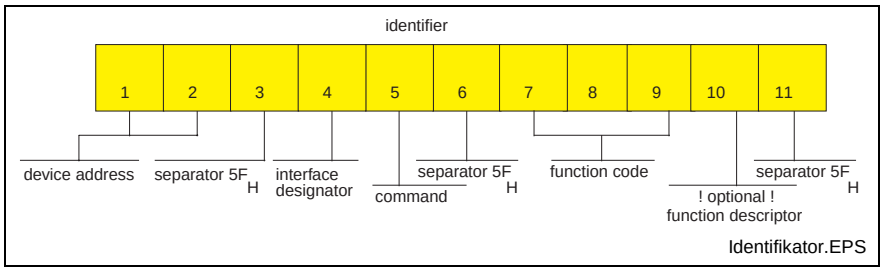


Fig. 5-1: Identifier

Device address

The device address corresponds to the system address within the Indramat BOF/GBO. For example, device 00 corresponds to system 0. Attention must be paid to the fact that the Indramat BOF/GBO always needs a device 00. The address is listed in the following table according to each specific device group and in the chapter entitled „Function interface commands“.

Address	Group	Affiliated device types
[xx]	MPCX	PCs
[00...15]	MTCX	MTCNC, MTC200-P, MTC200-R, MTVNC
[00]	MSCX	SERCANS-A, SERCANS-P
[00...63]	MISX	ISP200-P, ISP200-R
[00...15]	MTRX	TRA200-P, TRA200-R
[00...63]	MTAX	MTA200-P (ANDRON controller)

Separator

The separator „_“ separates the individual elements and is therefore a fixed component of the identifier.

Interface designator

BOF process
Controller (logic and communication processes)

All data accesses via the interface designator „B“ are managed by the BOF process so as to ensure that a user program can access the data from a controller via function calls, for example. When the function call BR_ASM5 (active system fault messages) is used, among other things the message number is fetched from the controller and the affiliated text is taken from the corresponding message file on the hard disk. When a file is opened, e.g. for editing with an editor, the BOF process ensures that a different user program cannot open the file once again

Data accesses by way of the interface designator „C“ are managed by the logic process and by the communication process, thus enabling access to data of the relevant device groups (MTCX and MTAX, etc.).

Read and write command

W = **W**rite

R = Single **R**ead

C = **C**yclic Read

B = **B**reak Cyclic Read

(Read) command „**R**“

A read request is passed on to the function interface with DataTransfer („00_CR_PPS_1_0_15_10“). On returning from this function, the user program is notified of a result buffer (*acBuffer) of a specific length (*lLen). In the result buffer, the requested data is made available in the requested data code. In the event of an error, the „DataTransfer“ routine is ended with an error (return value <>0) and it may be necessary to branch to an error routine. If the reply consists of several partial results (e.g. X1 125.4567 [mm]), the result must be interpreted with the „ReadGroupItem„ routine (see chapter entitled „Programming“)

Write command „**W**“

A new value for the specified PLC variable is passed on to the function interface in the specified buffer (*pcValue) with DataTransfer („00_CW_PVS_TEST“). On returning from the function, and in the event of the return value „0“, this function has been executed successfully. In the event of an error (return value <>0), it may be necessary to branch to an error routine. The write command may have to be repeated in the event of an error.

Function code

With its three letters, the function code provides information about the data to be accessed. The designator is encoded in the form of the data type designation. After the three letters, a function descriptor for the respective function code may be optionally necessary. This is described explicitly in the list of the access functions.

Example	Access without function descriptor
CR_PPS_1_0_1_2	Read the NC block N0002 out of the NC memory A, in the CNC process 0, from part program 1.

Example	Access with function descriptor
CR_NPA2_S00.00.022_S00.00.025	Read system parameters lines 22 to 25.

The function descriptor is necessary whenever several combinations can be read via the data code (e.g. 1 line, 1 element or several lines). In this case, the selectors would be assigned different meanings for the respective request. This is why these accesses are specified in greater detail with a function descriptor.

Selector

Example

Cyclic reading of the current axis speed of the 1st axis of device address 00.

The selector consists of a minimum of 1 character and a maximum of 17 characters. The selector is encoded in the form of numeric numbers that are separated by a separator ($5F_H$, 95_{Dec}). The selector directly depends on the addressed data type. Addressing is described in the sections dealing with the individual function calls.

Example 1 (cyclic reading)

Cyclic reading of the current axis speed of the 1st axis, of device address 00.

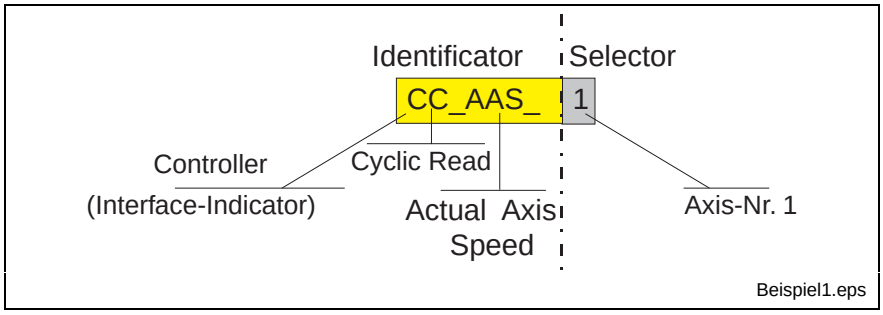


Fig. 5-2: Example 1: Cyclic reading in ASCII code

Example 2 (cyclic reading)

Cyclic reading of the current feed rate in the CNC process 3 of device address 02.

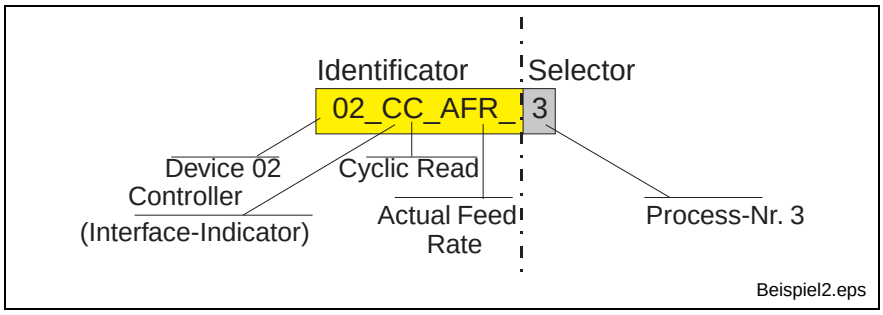


Fig. 5-3: Example 2: Cyclic reading in ASCII code

Data code

The data code is an optional designator in the function call. It can be specified after the selector, separated by a slash „/“. The specified data code defines the code of the data to be read; in the case of write functions, it defines the encoding of the request strings and the reply in the result buffer.

The following coding types are supported:

1 = ASCII default !

2 = Binary

3 = ANSI

4 = Unicode

Example 3 (cyclic reading) Cyclic reading of the current spindle speed of the 2nd spindle in the CNC process 1, of the device address 01 in the data code „Binary“.

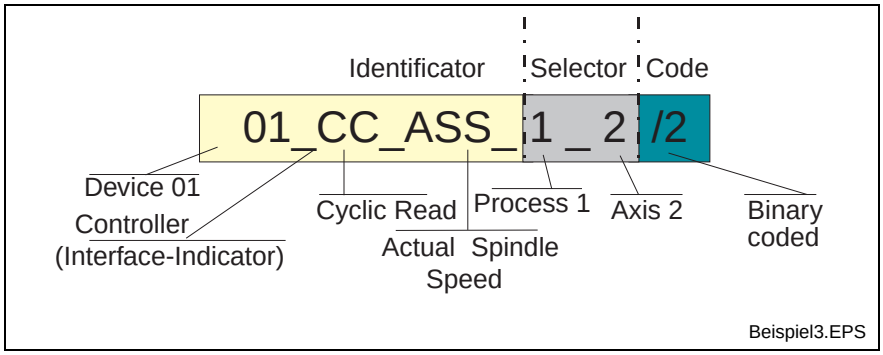


Fig. 5-4: Example 3: Cyclic reading in binary code

5.2 Data tables

The most frequently used parameters and their value ranges are listed in the following data tables.

General parameters

Parameter	Value range
Axis number	1...10
CNC memory	1=A, 2=B
NC block No.	1...10000
NC program number	0...99
Spindle number	S1, S2, S3
CNC process number	0...6
Mechanism number	0...31
Drive address	0...254
Tool number	0...9999999
Duplo No.	1...9999
Data block	0 = basic tool data 1...9 = cutter data
Data element	1...28 for basic tool data 1...40 for cutter data
Memory	M = Magazine/turret S = Spindle G = Gripper X = Index data
Position	1...999 in the case of M 1...4 in the case of S,G 0...16770215 in the case of X

Axis meanings

Code	Axis meaning	Axis type
0	X axis	Main axis
1	Y axis	Main axis
2	Z axis	Main axis
3	U axis	Secondary axis
4	V axis	Secondary axis
5	W axis	Secondary axis
6	A axis	Rotary axis
7	B axis	Rotary axis
8	C axis	Rotary axis
9	S1 axis	Spindle
10	S2 axis	Spindle
11	S3 axis	Spindle
20	Turret axis	Special type

Axis types

No.	Axis types	Remarks
0_H	AXIS_NOT_DEFINED	Axis not defined
1_H	ANALOG_LINEAR_AXIS	Analog linear axis
2_H	ANALOG_ROTARY_AXIS	Analog rotary axis
3_H	ANALOG_MAIN_SPINDLE	Analog spindle
4_H	ANALOG_COMB_TURRET_AXIS	Analog turret axis
5_H	C_AXIS	C axis
80_H	DYNAMIC_AXIS	Dynamically assignable axis
81_H	DIGITAL_LINEAR_AXIS	Linear axis
82_H	DIGITAL_ROTARY_AXIS	Rotary axis
83_H	DIGITAL_MAIN_SPINDLE	Spindle
84_H	DIGITAL_COM_TURRET_AXIS	Digital turret axis
85_H	DIGITAL_C_AXIS	Digitale C axis
87_H	DIGITAL_SERCOS_E_A	Digitale Sercos I/O

6 Function interface commands

6.1 FI commands for the MPCX device group

The following commands apply to the MPCX device group. Attention must be paid to the fact that the device address „XX“ must always be placed before the FI command, e.g. XX_BR_CCP1 (see chapter entitled „Elements of the FI command”)

Output of the device configuration: CCP

Designation **CCP** **C**ell **C**onfiguration **P**arameter

Explanation The configuration settings are read out of the „IND_DEV.INI“ file. The configuration of the individual communication addresses and the settings of the various Indramat devices are defined in this file (see chapter entitled „Contents of the „IND_DEV.INI“ file).

FI command Output of the configuration settings of all devices defined in the „IND_DEV.INI“ file.

XX_BR_CCP1 (Single Read)

Structure of the reply The following table shows the general structure of the reply of the FI command CCP1. The reply consists of one to n=16 rows (n=16 configurable devices) with 13 columns each.

Row 1...n	Column 1	...	Column 13

Value range/meanings of the columns

1 =	Device address	IND_DEV.INI entry: [DeviceAddrX]
2 =	Device name	IND_DEV.INI entry: DeviceName=
3 =	Device type	IND_DEV.INI entry: DeviceTyp=
4 =	PLC support	IND_DEV.INI entry: PLC=
5 =	Device status	IND_DEV.INI entry: DeviceStatus=
6 =	Allocation of a simulation pair	IND_DEV.INI entry: DeviceAssign=
7 =	Device mode	IND_DEV.INI entry: MtvncMode=
8 =	Communication channel	IND_DEV.INI entry: [CommAddrX]
9 =	Description of the communication channel	IND_DEV.INI entry: CommStr=
10 =	Timeout value	IND_DEV.INI entry: Timeout=
11 =	Device group	(see chapter entitled „Elements of the FI command“, identifier)
12 =	Type of component	IND_DEV.INI entry: Componenttype1=
13 =	Type of component	IND_DEV.INI entry: Componenttype2=

Example CCP1

Read the configuration settings of all devices defined in the „IND_DEV.INI“ file.

Assumption: the following device types are defined:

- Device address 00: SERCANS-A
- Device address 15: MTC200-P

FI command		**XX_BR_CCP1**
Row	**Column**	**Reply**
1	1	00
	2	Pressure roller drive
	3	SERCANS-A
	4	NO
	5	ON
	6	NO
	7	OFF
	8	4
	9	V24,COM2,19200,EVEN,RS232,TCOFF
	10	3500
	11	MSCX
	12	NONE
	13	NONE
2	1	15
	2	Conveyor
	3	MTC200-P
	4	YES
	5	ON
	6	NO
	7	OFF
	8	1
	9	DPR,$D000,$0000,$2000,RAM0,TCON
	10	3500
	11	MTCX
	12	MTS-P01.02
	13	MTC-P

FI command

Output of the configuration settings of the selected device types.

BR_CCP2_(1) **(Single Read)**

(1)= Device type [MTCNC, MTC200-P, MTC200-R, MTVNC, SERCANS-A, SERCANS-P, ISP200-P, ISP200-R, TRA200-P, TRA200-R, MTA200-P]

Structure of the reply

The following table shows the general structure of the reply of the FI command CCP2. The reply consists of one to n=16 rows (n=16 configurable devices) with 13 columns each.

Row 1...n	Column 1	...	Column 13

Value range/meanings of the columns

1 =	Device address	IND_DEV.INI entry: [DeviceAddrX]
2 =	Device name	IND_DEV.INI entry: DeviceName=
3 =	Device type	IND_DEV.INI entry: DeviceTyp=
4 =	PLC support	IND_DEV.INI entry: PLC=
5 =	Device status	IND_DEV.INI entry: DeviceStatus=
6 =	Allocation of a Simulation pair	IND_DEV.INI entry: DeviceAssign=
7 =	Device mode	IND_DEV.INI entry: MtvncMode=
8 =	Communication channel	IND_DEV.INI entry: [CommAddrX]
9 =	Description of the communication channel	IND_DEV.INI entry: CommStr=
10 =	Timeout value	IND_DEV.INI entry: Timeout=
11 =	Device group	(see chapter entitled „Elements of the FI command“, identifier)
12 =	Type of component	IND_DEV.INI entry: Componenttype1=
13 =	Type of component	IND_DEV.INI entry: Componenttype2=

Example CCP2

Read the configuration settings of the defined devices of the type SERCANS-A.

Assumption: the following device types are defined:

- Device address 00: SERCANS-A
- Device address 03: MTA200-P
- Device address 15: MTC200-P

FI command		**XX_BR_CCP2_SERCANS-A**
Row	**Column**	**Reply**
1	1	00
	2	Pressure roller drive
	3	SERCANS-A
	4	NO
	5	ON
	6	NO
	7	OFF
	8	4
	9	V24,COM2,19200,EVEN,RS232,TCOFF
	10	3500
	11	MSCX
	12	NONE
	13	NONE

FI command

Output of the configuration data of the devices that are addressed via the specified communication channel

BR_CCP3_(1) **(Single Read)**

(1)= Communication channel IND_DEV.INI entry: [CommAddrX]

Structure of the reply

The following table shows the general structure of the reply of the FI command CCP3. The reply consists of one to n=16 rows (n=16 configurable devices) with 13 columns each.

Row 1...n	Column 1	...	Column 13

Value range/meanings of the columns

1 =	Device address	IND_DEV.INI entry: [DeviceAddrX]
2 =	Device name	IND_DEV.INI entry: DeviceName=
3 =	Device type	IND_DEV.INI entry: DeviceTyp=
4 =	PLC support	IND_DEV.INI entry: PLC=
5 =	Device status	IND_DEV.INI entry: DeviceStatus=
6 =	Allocation of a Simulation pair	IND_DEV.INI entry: DeviceAssign=
7 =	Device mode	IND_DEV.INI entry: MtvncMode=
8 =	Communication channel	IND_DEV.INI entry: [CommAddrX]
9 =	Description of the communication channel	IND_DEV.INI entry: CommStr=
10 =	Timeout value	IND_DEV.INI entry: Timeout=
11 =	Device group	(see chapter entitled „Elements of the FI command“, identifier)
12 =	Type of component	IND_DEV.INI entry: Componenttype1=
13 =	Type of component	IND_DEV.INI entry: Componenttype2=

Example CCP3

Read the configuration data of the devices that are addressed via communication channel 1.

Assumption: the following device types are defined:

- Communication channel 4: SERCANS-A
- Communication channel 5: MTA200-P
- Communication channel 1: MTC200-P

FI command		**XX_BR_CCP3_4**
Row	**Column**	**Reply**
1	1	15
	2	Conveyor
	3	MTC200-P
	4	YES
	5	ON
	6	NO
	7	OFF
	8	1
	9	DPR,$D000,$0000,$2000,RAM0,TCON
	10	3500
	11	MTCX
	12	MTS-P01.2
	13	MTC-P

FI command

Output of the configuration data of the devices that are addressed via the specified communication channel.

BR_CCP4_(1) **(Single Read)**

(1) = Device group [MTCX, MSCX, MISX, MTRX, MTAX]
(see chapter entitled „Elements of the FI command“, identifier)

Structure of the reply

The following table shows the general structure of the reply of the FI command CCP4. The reply consists of one to n=16 rows (n=16 configurable devices) with 13 columns each.

Row 1...n	**Column 1**	**...**	**Column 13**

Value range/meanings of the columns

1 =	Device address	IND_DEV.INI entry: [DeviceAddrX]
2 =	Device name	IND_DEV.INI entry: DeviceName=
3 =	Device type	IND_DEV.INI entry: DeviceTyp=
4 =	PLC support	IND_DEV.INI entry: PLC=
5 =	Device status	IND_DEV.INI entry: DeviceStatus=
6 =	Allocation of a Simulation pair	IND_DEV.INI entry: DeviceAssign=
7 =	Device mode	IND_DEV.INI entry: MtvncMode=
8 =	Communication channel	IND_DEV.INI entry: [CommAddrX]
9 =	Description of the communication channel	IND_DEV.INI entry: CommStr=
10 =	Timeout value	IND_DEV.INI entry: Timeout=
11 =	Device group	(see chapter entitled „Elements of the FI command“, identifier)
12 =	Type of component	IND_DEV.INI entry: Componenttype1=
13 =	Type of component	IND_DEV.INI entry: Componenttype2=

Example CCP4

Read the configuration settings of the defined MSCX devices. Assumption: the following device groups are defined:

- Device address 00: MSCX
- Device address 03: MTCX

FI command		**XX_BR_CCP4_MSCX**
Row	**Column**	**Reply**
1	1	00
	2	Pressure roller drive
	3	SERCANS-A
	4	NO
	5	ON
	6	NO
	7	OFF
	8	4
	9	V24,COM2,19200,EVEN,RS232,TCOFF
	10	3500
	11	MSCX
	12	NONE
	13	NONE

FI command

Output of the configuration data of the device that is addressed via the specified device address.

BR_CCP5_(1) (Single Read)

(1) = Device address [00...63]

Structure of the reply

The following table shows the general structure of the reply of the FI command CCP5. The reply consists of one row with 13 columns.

Row 1...n	Column 1	...	Column 13

Value range/meanings of the columns

1 =	Device address	IND_DEV.INI entry: [DeviceAddrX]
2 =	Device name	IND_DEV.INI entry: DeviceName=
3 =	Device type	IND_DEV.INI entry: DeviceTyp=
4 =	PLC support	IND_DEV.INI entry: PLC=
5 =	Device status	IND_DEV.INI entry: DeviceStatus=
6 =	Allocation of a Simulation pair	IND_DEV.INI entry: DeviceAssign=
7 =	Device mode	IND_DEV.INI entry: MtvncMode=
8 =	Communication channel	IND_DEV.INI entry: [CommAddrX]
9 =	Description of the communication channel	IND_DEV.INI entry: CommStr=
10 =	Timeout value	IND_DEV.INI entry: Timeout=
11 =	Device group	(see chapter entitled „Elements of the FI command“, identifier)
12 =	Type of component	IND_DEV.INI entry: Componenttype1=
13 =	Type of component	IND_DEV.INI entry: Componenttype2=

Example CCP5

Read the configuration settings of the defined MSCX devices. Assumption: the following device groups are defined:

- Device address 00: MSCX
- Device address 03: MTCX

FI command		**XX_BR_CCP5_00**
Row	**Column**	**Reply**
1	1	00
	2	Pressure roller drive
	3	SERCANS-A
	4	NO
	5	ON
	6	NO
	7	OFF
	8	4
	9	V24,COM2,19200,EVEN,RS232,TCOFF
	10	3500
	11	MSCX
	12	NONE
	13	NONE

Removing function interface jobs: DFJ

Designation

DFJ **D**elete **F**unction-Interface **J**obs

Explanation

FI jobs are removed from the management structure of the function interface. These are jobs that have the „READY“ or „ERROR“ status. The FI command DFJ1 removes all interface jobs, while DFJ2 removes the selected job.

Note: See „NC download“ in the chapter entitled „FI commands of the MTCX device group“.

FI command

Remove all FI jobs from the management structure of the function interface.

XX_BR_DFJ1 (Single Read)

Structure of the reply

The following table shows the general structure of the reply of the FI command DFJ1. The reply consists of one to n=19 rows (n=19 maximum number of FI jobs) with two columns each.

Row 1...n	Column 1	Column 2

Value range/meanings of the columns

1 = Deleted job ID [01...20]

2 = FI command

Example DFJ1

Delete all FI jobs. Assumption:

An NC program has been successfully complied with the FI command „NPC“ and then transferred to the device (controller) with the FI command „NPD“ (see FI commands of the MTCX device group).

- Job ID of NC program compiling: 01
- Job ID of NC program downloads: 02

FI command		**XX_BR_DFJ1**
Row	**Column**	**Reply**
1	1	01
	2	02_BR_NPC1_1 /3
2	1	02
	2	02_BR_NPD1_1_1 /3

FI command Remove the selected FI job from he management structure of the function interface.

XX_BR_DFJ2_(1) (Single Read)

(1) = Job-ID [01...20]

Structure of the reply The following table shows the general structure of the reply of the FI command DFJ2. The reply consists of one row with two columns.

Row 1	Column 1	Column 2

Value range/meanings of the columns

1 = Deleted job ID [01...20]

2 = FI command string

Example DFJ2 Delete FI job 01. Assumption:

An NC program has been successfully complied with the FI command „NPC“ and then transferred to the device (controller) with the FI command „NPD“ (see FI commands of the MTCX device group).

- Job ID of NC program compiling: 01
- Job ID of NC program downloads: 02

FI command		**XX_BR_DFJ2_01**
Row	**Column**	**Reply**
1	1	01
	2	02_BR_NPC1_1 /3

Information about function interface jobs: IFJ

Designation **IFJ** **I**nformation about **F**unction-Interface **J**obs

Explanation Status information about active FI jobs can be read. For example, this status query makes it possible to realise a progress display during an NC download because, depending on the size of the NC program, it may take some time to execute in the background.

Note: See „NC download“ in the chapter entitled „FI commands of the MTCX device group“

FI command Returns status information about all active FI jobs.

XX_BR_IFJ1 (Single Read)

Structure of the reply The following table shows the general structure of the reply of the FI command IFJ1. The reply consists of one to n=19 rows (n=19 maximum number of FI jobs) with 16 columns each.

Row 1...n	Column 1	...	Column 16

Value range/meanings of the columns

1 =	Job-ID	[01...20]
2 =	FI command string	
3 =	Job classification	[1 = NC download, 2 = compile NC program package]
4 =	Job status	[RUN, READY, ERROR]
5 =	Number of error lines in the error info buffer	
6 =	Max. processing time [ms] up to a TIMEOUT	
7 =	Job start time	[hh:mm:ss:ms]
8 =	Elapsed processing time in ms	
9 =	Function interface login name of application	
10 =	Progress type	[1 = progress in %, 2 = absolute progress display]
11 =	Percentage progress	[value, --], dependent on column 10 „progress type“
12 =	Absolute progress	[value, --], dependent on column 10 „progress type“
13 =	Absolute final value	[value, --], dependent on column 10 „progress type“
14 =	Progress info buffer, contains display information, e.g. NC program line that is currently being transferred.	
15 =	FI job error code	(see chapter entitled „Error codes“)
16 =	Error info buffer	

Note: The results of the columns depend on the started FI job. See „NC download“ in the chapter entitled „FI commands of the MTCX device group“.

Example IFJ1

Read the status information of all active FI jobs. Assumption:

- The job with the ID 01 has been started with the FI command „NPC“ and successfully ended with the READY message.

FI command		**XX_BR_IFJ1**
Row	**Column**	**Reply**
1	1	01
	2	02_BR_NPC1_1 /3
	3	2
	4	READY
	5	0
	6	600000
	7	16:15:22:123
	8	120
	9	VBDemo
	10	1
	11	100 %
	12	--
	13	--
	14	--
	15	0
	16	--

FI command

Returns information about the selected, active FI job.

XX_BR_IFJ2_(1) **(Single Read)**

(1) = Job ID [01...20]

Note: Information about the structure of the reply can be found in the description of the aforementioned FI command „XX_BR_IFJ1“.

Activated language of the Indramat user interface: LNG

Designation **LNG** Activated **La**ng**NG**uage

Explanation The country code of the activated language of the Indramat user interface is output.

FI command

XX_BR_LNG **(Single Read)**

Structure of the reply The reply of the FI commands LNG consists of one row with one column for the country code belonging to the activate language.

Value range of the column 1 = Country code of the activated language [2 ASCII characters]

Country code	Language
EN	English (US/GB)
GR	German
ES	Spanish
IT	Italian
FR	French
HU	Hungarian
PT	Portuguese
SE	Swedish
CS	Czech

Example LNG Read the country code of the activated language in the Indramat user interface.

I command	XX_BR_LNG
Reply	
Row	Column 1
1	SE

Far configuration parameter: FCP

Designation

FCP **F**ar Device **C**onfiguration **P**arameter

Explanation

The list of FarDevice entries defined on the PC is read. If no network configuration data is available, or if the local PC is inactive (disabled) in the network configuration, the list of local devices is output.

FI command

Read available devices.

XX_BR_FCP1{_(1)) (Single Read)

(1) = device selection [L= only local, F= only FAR] ! Optional !

Structure of the reply

The following table shows the general structure of the FI command FCP1. The number of rows depends on the current configuration.

Result: network configuration available

Row 1...n	Column 1	...	Column 9

Value range/meanings of the columns

1 = Device address
2 = Device name
3 = Device type
4 = Local device address
5 = Logical address of the PC
6 = Local device [YES, NO]
7 = Device status
8 = Allocation of a Simulation pair
9 = Device group

Note: If the [DeviceOrder] exists in the „IND_DEV.INI“ file or in the „FAR_DEV.INI“ file, the entries (rows) are output according to the specified order. If there is no [DeviceOrder] entry, the devices are output according to the order of the sections in the file.

Example FCP1 Network configuration data available

Read the network configuration of all devices defined in the „FAR_DEV.INI“ and „IND_DEV.INI“ files.

Assumption: the following device types are defined:

- Device address 15: MTCNC
- Device address 11: MTVNC
- Device address 12: MTVNC

FI command		**XX_BR_FCP1**
Row	**Column**	**Reply**
1	1	15
	2	Drilling left
	3	MTCNC
	4	05
	5	02
	6	YES
	7	ON
	8	11
	9	MTCX
2	1	11
	2	Drilling left
	3	MTVNC
	4	01
	5	02
	6	YES
	7	ON
	8	15
	9	MTCX
3	1	12
	2	Drilling right
	3	MTVNC
	4	02
	5	03
	6	NO
	7	OFF
	8	NO
	9	MTCX

Example FCP1
NO network configuration data available

Read the network configuration of all devices defined in the „FAR_DEV.INI“ and „IND_DEV.INI“ files.

Assumption: the following device types are defined:

- Device address 05: MTCNC
- Device address 01: MTVNC

Note: No network configuration data available or local PC in the network not active.

FI command		**XX_BR_FCP1**
Row	**Column**	**Reply**
1	1	05
	2	Drilling left
	3	MTCNC
	4	05
	5	--
	6	YES
	7	ON
	8	NO
	9	MTCX
2	1	01
	2	Drilling left
	3	MTVNC
	4	01
	5	--
	6	YES
	7	ON
	8	NO
	9	MTCX

Note: In any case, the contents of column 1 can be used to address both the local and the remote devices. A generic application must specify the value as a device address within the FI command.

The command combines the „Disable“ attribute of the FarDevice# with the „Disable“ attribute of the affiliated PC.

Note: If a PC is PC disabled, its affiliated devices are also Disabled.

	FarDevice, Disable YES	**FarDevice, Disable NO**
PC, Disable YES	Disabled	Disabled
PC, Disable NO	Disabled	Enabled

FI command But only for devices of a specific device group:

XX_BR_FCP2_(1){_(2)} **(Single Read)**

(1) = Device group

(2) = Device selection [L= only local, F= only FAR] ! Optional !

FI command Read the available devices of a device type (see FCP1, but only for devices belonging to a specific device type).

XX_BR_FCP3_(1){_(2)} **(Single Read)**

(1) = Device type [MTCNC, MTC200-P, MTC200-R, MTVNC, SERCANS-A, SERCANS-P, ISP200-P, ISP200-R, TRA200-P, TRA200-R, MTA200-P]

(2) = Device selection [L= only local, F= only FAR] ! Optional !

Far Device configuration parameter: FDC

Designation **FDC** **F**ar **D**evice **C**onfiguration

Explanation Returns the general data of the PC network. If no network data is available, the information is taken from the local definitions of the devices.

FI command **XX_BR_FDC1** **(Single Read)**

Structure of the reply The following table shows the general structure of the reply of the FI command FDC1.

Value range/meanings of the rows/columns

Row 1; Column 1...4

1 = PC network available? [YES, NO]

2 = Name of PC network (string)

3 = Maximum number of PCs (Integer)

4 = Maximum number of devices (Integer)

Row 2; Column 1

1 = Logical address of PC (Integer)

Row 3; Column 1

1 = Host name (string)
Internet host name, possibly with added domain name

Row 4; Column 1

1 = Computer name (string)
NETBIOS NAME of the computer

Row 5; Column 1...4

1 = IP address of network card 1 (string)

... ...

4 = IP address of network card 4 (string)

Row 6; Column 1

1 = MasterPC (string)
[YES = PC is MasterPC (header PC), NO]

Example FDC1
Network configuration data available

Read the general data of the PC network.

Assumption: a PC with two network cards is defined:

- IP address of the 1st network card: 172.16.0.1
- IP address of the 2nd network card: 172.16.1.1

FI command		**XX_BR_FDC1**
Row	**Column**	**Reply**
1	1	YES
	2	Operation10
	3	20
	4	64
2	1	19
3	1	machine1.cell1
4	1	MACHINE1
5	1	172.16.0.1
	2	172.16.1.1
6	1	YES

Example FDC1 NO network configuration data available

Read the general data of the PC network.

Assumption: no PC in the network is active or defined.

FI command		**XX_BR_FDC1**
Row	**Column**	**Reply**
1	1	NO
	2	--
	3	1
	4	16
2	1	--
3	1	machine1.cell1
4	1	MACHINE1
5	1	172.16.0.1
6	1	--

Far PC configuration parameter: FPC

Designation

FPC **F**ar **P**C **C**onfiguration Parameter

Explanation

Outputs the list of the PCs that are defined in the function interface. The PCs are output in the order of the sections in the „FAR_DEV.INI“ file.

FI command

XX_BR_FPC1 **(Single Read)**

Structure of the reply

The following table shows the general structure of the reply of the FI command FPC1. The number of rows depends on the current configuration. Result: network configuration data available.

ow 1...n	**Column 1**	**...**	**Column 6**

Value range/meanings of the columns

1 = Number of the PC

2 = Port [IP address, host name]

3 = Name of the PC

4 = Local device
[YES = PC is the local PC, NO = PC is a remote PC]

5 = Device status;
corresponds to the „Disable“ entry of the „PC<pcnr>„ section
[OFF = PC is disabled (not addressable in the PC network);
ON = PC is enabled (addressable in the PC network)]

6 = Master;
corresponds to the „MasterPC“ entry of the „PC<pcnr>„ section
[YES = PC is MasterPC (header PC), NO]

Example FPC1 Network configuration data available

Read the list of the PCs that are defined in the function interface. Assumption: two PCs are defined:

- PC1 with the IP address: 192.4.4.91
- PC2 with the host name: st100103

FI command		XX_BR_FPC1
Row	**Column**	**Reply**
1	1	10
	2	192.4.4.91
	3	Drilling station-02
	4	YES
	5	OFF
	6	NO
2	1	20
	2	st100103
	3	Drilling station 03
	4	NO
	5	ON
	6	YES

Example FPC1 NO network configuration data available

Read the list of the PCs that are defined in the function interface. Assumption: no PCs are defined.

FI command		XX_BR_FPC1
Row	**Column**	**Reply**
1	1	XX
	2	--
	3	--
	4	YES
	5	ON
	6	YES

6.2 FI commands for the MTCX device group

The FI commands described in this chapter apply to the MTCX device group. The following types and the possible device addresses are listed in this device group:

Group	Affiliated types	Address
MTCX	MTCNC, MTC200-P, MTC200-R, MTVNC	[00...15]

Note: Pay attention to the fact that the device address must be placed before the respective FI command, e.g. 00_CR_AAC_0 (refer also the chapter entitled „Elements of the FI command“).

Active acceleration value: AAC

Designation

AAC **A**ctive **AC**celeration

Explanation

The current acceleration value of a CNC process is read. Within an NC program, an acceleration limit can be programmed with the „programmable acceleration ACC“ function. This is the case, for example, when the axes of the workpiece carrier are to be moved depending on the weight of the workpiece.

FI command

Output of the active acceleration value of a CNC process of the selected device in the MTCX device group.

CR_AAC1_(1) **(Single Read)**

CC_AAC1_(1) **(Cyclic Read)**

CB_AAC1_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply

The following table shows the general structure of the reply of the FI command AAC. A row with three columns is output for the NC command, the acceleration value and the unit.

Row 1	Column 1	....	Column 3

Value range of the columns

1 = NC command [ACC]

2 = Acceleration value [0...100]

3 = Unit [%]

Example AAC1 Read the active acceleration value in the CNC process 0 of device address 00.

FI command	00_CR_AAC1_0		
Reply			
Row	Column 1	Column 2	Column 3
1	ACC	50	%

References For further information about the acceleration value and the value range, refer to the Indramat documentation:

„NC programming instructions V17“,
Chapter 4.2 Interpolation conditions/Programmable acceleration „ACC“,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active angle dimension (RAD/DEG): AAD

Designation **AAD** **A**ctive **A**ngle **D**imension

Explanation The active angle dimension of a CNC process is read. The arguments of the angle functions SIN, COS, TAN and the results of the inverse functions of the angle functions ASIN, ACOS, ATAN can be specified or calculated in the unit Radiant (RAD) and in the unit degree (DEG).

FI command Output of the active angle unit of a CNC process of the selected device belonging to the MTCX device group.

CR_AAD_(1) **(Single Read)**

CC_AAD_(1) **(Cyclic Read)**

CB_AAD_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The reply of the FI command AAD consists of one row with one columns for the unit [RAD/DEG].

Example AAD Read the active angle unit in the CNC process 0 of the device address 00.

FI command	00_CR_AAD_0
Reply	
Row	Column 1
1	RAD

References Further information about the arguments of the angle functions can be found in the Indramat documentation:

„NC programming instructions V17“,
Chapter 10.2 Angle unit for angle functions „RAD“, „DEG“,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Actual axis speed (spindle speed): AAS

Designation **AAS** **A**ctual **A**xis **S**peed

Explanation The actual axis speeds and spindle speeds of a CNC process of the selected device are read. The FI command „AAS1“ refers to the CNC process number and the code of the axis meaning and, at the same time, the FI command „AAS2“ enables an output of the current speed that is referred to the physical axis number.

FI command Output of the actual axis speed that is referred to the CNC process number and the code of the axis meaning.

CR_AAS1_(1)_(2) (Single Read)

CC_AAS1_(1)_(2) (Cyclic Read)

CB_AAS1_(1)_(2) (Break Cyclic Read)

(1) = CNC process number [0...6]

(2) = Axis meaning [0...11; 20] (see chapter entitled „Data tables „)

Structure of the reply The following table shows the general structure of the reply of the FI command AAS. One row with three columns for the axis designation, the axis speed and the unit is output.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation [according to axis parameter settings]

2 = Speed [according to axis parameter settings]

3 = Unit [according to process parameter settings]

Example AAS1 Read the current axis speed of the Z axis in CNC process 0 of the device address 00.

FI command	00_CR_AAS1_0_2		
Reply			
Row	Column 1	Column 2	Column 3
1	Z	-158.2	[mm/min]

References For further information about the axis-specific speeds, refer to the Indramat documentation:

„NC programming instructions V17“, Chapter 4.3 Interpolations functions/line interpolation, rapid traversing „G00“,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E

„MT-CNC Numeric control for multi-axes, multi-process applications“, Chapter 2.6 Maximum path speed,
DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

FI command

Output of the axis speed of the selected device referred to the physical axis number.

CR_AAS2_(1)	**(Single Read)**
CC_AAS2_(1)	**(Cyclic Read)**
CB_AAS2_(1)	**(Break Cyclic Read)**

(1) = Physical axis number [1...32, according to system parameter settings]

Structure of the reply

The following table shows the general structure of the reply of the FI command AAS2. One row with three columns for the axis designation, speed and unit is output.

Row 1	**Column 1**	**...**	**Column 3**

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Speed	[according to axis parameter settings]
3 = Unit	[according to process parameter settings]

Example AAS2

Read the current speed of the Spindle S (physical axis number 4) of device address 00.

FI command	**00_CR_AAS2_4**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	S	4000.0	[1/min]

References

For further information about the axis-specific speeds, refer to the Indramat documentation:

„ MT-CNC Numeric control for multi-axes, multi-process applications „, Chapter 2.6 Maximum path speed, DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

Active NC block: ABI

Designation

ABI **A**ctive NC-**B**lock **I**nformation

Explanation

The active NC block or a user-defined NC block is read. This permits the creation of an NC block display with the active NC block and the number of previous and subsequent NC blocks.

FI command

Output of the active NC block and the previous and subsequent NC blocks of a CNC process of the selected device in the MTCX device group.

BR_ABI_(1){_(2)_(3)}	**(Single Read)**
BC_ABI_(1){_(2)_(3)}	**(Cyclic Read)**

BB_ABI_(1){_(2)_(3)} **(Break Cyclic Read)**

(1) = CNC process number [0...6]

(2) = Number of previous NCs Sets [1..4] ! Optional !

(3) = Number of subsequent NCs Sets [1..4] ! Optional !

Note: If the optional parameters are not specified, only the current NC block is output.

Structure of the reply

The number of rows of the reply (1 ... n=9) depends on the number of requested NC blocks. Each row consists of one column containing the relevant NC block.

Note: If there is no valid NC program in the device, the value of the column is [--] .

Example ABI

Read the active NC block and the two previous and subsequent NC blocks of the CNC process of device address 00.

FI command	**00_BR_ABI_0_2_2**
Reply	
Row	**Column 1**
1	N0000 .START
2	N0001 T13 BSR .M6
3	N0002 G90 G41 G54 G17 F2000.0 S3200.00 M003
4	N0003 G00 X 60.0000 Y -30.0000
5	N0004 Z -6.0000

References

For further information about the structure of an NC block, refer to the Indramat documentation:

„NC programming instructions V17“,
Chapter 2.4 Elements of an NC block,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active cutting speed of the reference spindle: ACS

Designation **ACS** **A**ctive **C**utting **S**peed

Explanation

Output of the active cutting speed of the reference spindle of a CNC process of the selected device belonging to the MTCX device group.

FI command

CR_ACS_(1) **(Single Read)**

CC_ACS_(1) **(Cyclic Read)**

CB_ACS_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply

The following table shows the general structure of the reply of the FI command ACS. One row with three columns for the S number of the reference spindle, cutting speed and the unit according to the system parameter settings is output.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = S number of reference spindle	[S1, S2, S3]
2 = Cutting speed	[Format according to parameter settings]
3 = Unit	[according to system parameter settings]

Note: If no reference spindle is defined in the selected CNC process, the value of column 1 is [*S], and the values of columns 2 and 3 is [--].

Example ACS

Read the active cutting speed in CNC process 0 of device address 00.

FI command	**00_CR_ACS_0**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	S1	200	[m/min]

References

For further information about the reference spindle and about NC programming of the cutting speed, refer to the Indramat documentation:

„CNC NC programming instructions V17“,
Chapter 4.5 Spindle speed, Constant cutting speed „G96“/Selecting the reference spindle „SPF“,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active D-correction number: ADN

Designation

ADN **A**ctive **D**-Correction **N**umber

Explanation

The active D-correction number of an NC process belonging to the MTCX device group is output. The D-corrections are additive registers with respect to the tool geometry data of tool management.

FI command

Output of the active D-correction number of a CNC process of the selected device belonging to the MTCX device group.

FI command

CR_ADN1_(1) **(Single Read)**

CC_ADN1_(1) **(Cyclic Read)**

CB_ADN1_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply A row with two columns for the active D-correction number of the specified CNC process is output. Meanings:

1 = Designator	[D]
2 = D-correction number	[0] = Cancel D-correction [1..99] = Select a D-correction

Example ADN Read the active D-correction of the CNC process 0 belonging to device address 00.

FI command	**00_CR_ADN1_0**	
Reply		
Row	**Column 1**	**Column 2**
1	D	0

References For further information about the D-corrections, refer to the Indramat documentation:

„NC programming instructions V17“, Chapter 5.8 D-corrections, DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active event monitoring: AEM

Designation **AEM** **A**ctive **E**vent **M**onitoring

Explanation The status of event monitoring of the specified NC process belonging to the MTCX device group is output. Events are binary variables that can be used by the NC program which, just like flags in the PLC program, represent any state defined by the programmer. By waiting for a defined state of an event, it is therefore possible to achieve process synchronisation.

FI command Output of the status of event monitoring of a CNC process belonging to the selected device of the MTCX device group.

CR_AEM_(1)	**(Single Read)**
CC_AEM_(1)	**(Cyclic Read)**
CB_AEM_(1)	**(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply A row and a column are output for the status of event monitoring. Meanings:

- EEV = Activate event monitoring
- DEV = Cancel event monitoring

Example AEM Read the status of event monitoring of the CNC process 0 belonging to device address 00.

FI command	**00_CR_AEM_0**
Reply	
Row	**Column 1**
1	EEV

References For further information about events and how to handle them, refer to the Indramat documentation:

„NC programming instructions V17“, Chapter 7 Events, DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active tool cutter number: AEN

Designation **AEN** **A**ctive **E**dge-**N**umber

Explanation The active tool cutter number of a CNC process is output. A change in the active cutter in the NC program results in provision of the affiliated correction and tool life data, which tool management then accesses during the subsequent machining process.

FI command Output of the active tool cutter number of a CNC process belonging to the selected device of the MTCX device group.

CR_AEN_(1) **(Single Read)**

CC_AEN_(1) **(Cyclic Read)**

CB_AEN_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply A row with two columns for the designator „E = Edge“ of the active tool cutter number is output. The numeric value assigned to the address letter „E“ by way of a single-digit decimal number [1...9] corresponds to the active cutter.

Example AEN Read the active tool cutter number of the CNC process 0 belonging to the device address 00.

FI command	**00_CR_AEN_0**	
Reply		
Row	**Column 1**	**Column 2**
1	E	1

References For further information about commands for tool management, refer to the Indramat documentation:

„NC programming instructions V17“,
Chapter 8 Commands for tool management/cutter selection „E“,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active feed rate override: AFO

Designation **AFO** **A**ctive **F**eedrate **O**verride

Explanation The current value of the feed rate override of a CNC process is output. The override is evaluated in the NC regardless of the mode and takes effect with all axis movements (except reference travel of digital axes).

FI command Output of the current value of the feed rate override of a CNC process belonging to the selected device of the MTCX device group.

CR_AFO1_(1)	**(Single Read)**
CC_AFO1_(1)	**(Cyclic Read)**
CB_AFO1_(1)	**(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The following table shows the general structure of the reply of the FI command AFO. A row with 3 columns for designator, the current value of the feed rate override and the unit [%] is output.

Row 1	Column 1	....	Column 3

Value range/meaning of the columns

1 = Designator	[OVR=Override]
2 = Current value of the feed rate override	[0...255]
3 = Unit	[%]

Note: The permissible range of override evaluation by the PLC program is between 0 and 255 %. The NC limits the axis or process speed to the maximum values defined in the parameters if too high an override value is set.

Example AFO1 Read the current value of the feed rate override in the CNC process 0 of device address 00.

FI command	**00_CR_AFO1_0**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	OVR	60	%

References For further information about the feed rate override, refer to the Indramat documentation:

„CNC/PLC Interface description 18VRS“,
Chapter 2.7 Feed rate and spindle override „PxxCSOVRD“,
DOK-MTC200-SPS*GWY*V18-ANW1-DE-E.

Actual feed rate: AFR

Designation **AFR** **A**ctual **F**eed**R**ate

Explanation The current value of the feed rate of a CNC process is output. The feed rate entry in an NC program is expressed as a feed rate word with the address letter „F“ and a feed rate that is specified directly as a constant or by way of an expression.

FI command Output of the current value of the feed rate of a CNC process belonging to the selected device of the MTCX device group.

CR_AFR_(1)	**(Single Read)**
CC_AFR_(1)	**(Cyclic Read)**
CB_AFR_(1)	**(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The following table shows the general structure of the reply of the FI command AFR. A row with three columns for the designator, the current feed rate value and the unit [%] is output.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Designator	[F = Feedrate]
2 = Value	[Format according to parameter settings]
3 = Unit	[Depending on process parameter settings]

Example AFR Read the current feed rate in the CNC process 0 of device address 00.

FI command	**00_CR_AFR_0**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	F	5000.0	[mm/min]

References For further information about the feed rate, refer to the Indramat documentation:

„CNC NC programming instructions V17“, Chapter 4.4 Feed rate,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active G functions: AGF

Designation **AGF** **A**ctive **G**-**F**unction

Explanation The active G functions of a CNC process belonging to the selected device of the MTCX device group are read.

FI command

CR_AGF_(1){_(2)} **(Single Read)**
CC_AGF_(1){_(2)} **(Cyclic Read)**
CB_AGF_(1){_(2)} **(Break Cyclic Read)**

(1) = CNC process number [0...6]
(2) = G code group [1...21] ! Optional !

Note: If the optional parameter is not specified, all active G codes of all G code groups are output.

Structure of the reply One row is output, and the number of columns depends on the number of requested G code groups. If the optional parameter has not been specified, the reply consists of a row with 21 columns. If the optional parameter has been specified, the reply consists of a row with one column containing the active G function of the selected G code group.

Note: In the event that no G function of the selected G code group is active, the reply consists of the character [--].

Example AGF Read the active G functions of G code group 17 in the CNC process 0 or device address 00.

FI command	**00_CR_AGF_0_17**
Reply	
Row	**Column 1**
1	G30

References For further information about the operating principle of the G functions and the breakdown of the G code groups, refer to the Indramat documentation:

„NC programming instructions V17“, Chapter 14.1 Table of G code groups, DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active M functions: AMF

Designation

AMF **A**ctive **M**-**F**unction

Explanation

The active M functions of a CNC process belonging to the selected device of the MTCX device group are read.

FI command

CR_AMF_(1){_(2)} **(Single Read)**
CC_AMF_(1){_(2)} **(Cyclic Read)**
CB_AMF_(1){_(2)} **(Break Cyclic Read)**

(1) = CNC process number [0...6]
(2) = M function group [1...16] ! Optional !

Note: If the optional parameter is not specified, all active M functions of all M function groups are output.

Structure of the reply

One row is output, and the number of columns depends on the number of requested M function groups. If the optional parameter has not been specified, the reply consists of a row with 21 columns. If the optional parameter has been specified, the reply consists of a row with one column containing the active M function of the selected M function group.

Note: In the event that no M function of the selected M function group is active, the reply consists of the character [--].

Example AMF

Read the active M functions of the M function group 2 in the CNC process 0 of the device address 00.

FI command	**00_CR_AMF_0_2**
Reply	
Row	**Column 1**
1	M005

References

For further information about the operating principle of the M functions and the breakdown of the M function groups, refer to the Indramat documentation:

„NC programming instructions V17“,
Chapter 14.2 Table of M function groups,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active mechanism messages: AMM

Designation

AMM **A**ctive **M**echanism **M**essages

Explanation

Messages pertaining to active mechanism faults and mechanism diagnostics are output. These messages are assigned to a specific mechanism or process. Depending on the FI command, the device address, the device name, the mechanism number, the mechanism name, the message type, the message source, the message number, a short text and a more detailed text are output.

FI command

Output of the currently applied mechanism messages of all active devices.

BR_AMM1	**(Single Read)**
BC_AMM1	**(Cyclic Read)**
BB_AMM1	**(Break Cyclic Read)**

Note: The FI command AMM1 refers to all devices within the device group. This is why any valid device address can be specified in the command line (see example AMM1).

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM1. The reply consists of one to a maximum of n=512 rows (n=16 devices x 32 mechanisms = 512) with 11 columns each.

Row 1..n	Column 1	...	Column 11

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0...31]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Short text	[max. 54 ASCII characters]
10 =	Detailed text	[x= available, -- = not available]
11 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see AMM5)

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“ Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE .

Example AMM1

Read the current mechanism messages of all active devices.
Assumption: the following device address and mechanisms are defined:

- Device address 01 with 2 mechanisms 0 and 1 and
- Device address 03 with one mechanism 0.

FI command		03_BR_AMM1
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	0
	4	Station 1
	5	D
	6	CNC
	7	S
	8	79
	9	Station waits until tool change command ended.
	10	x
	11	0
2	1	01
	2	Drilling centre
	3	1
	4	Station 2
	5	F
	6	CNC
	7	O
	8	1
	9	No external 24V power supply.
	10	x
	11	0
3	1	03
	2	Milling centre
	3	0
	4	Camshaft30.40.25.0S
	5	D
	6	CNC
	7	S
	8	71
	9	Circular interpolation
	10	x
	11	0

FI command

Output of the currently applied mechanism messages of the selected device.

BR_AMM2	**(Single Read)**
BC_AMM2	**(Cyclic Read)**
BB_AMM2	**(Break Cyclic Read)**

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM2. The reply consists of one to n=31 rows with 11 columns each.

Row 1..n	Column 1	...	Column 11

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[32 ASCII characters]
3 =	Mechanism number	[0...31]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Short text	[max. 54 ASCII characters]
10 =	Detailed text	[x= available, -- = not available]
11 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see AMM5)

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“ Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Example AMM2 Read the current mechanism messages of device address 01.
Assumption: device address 01 with 2 defined mechanisms 0 and 1.

FI command		**01_BR_AMM2**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	0
	4	Station 1
	5	D
	6	CNC
	7	S
	8	79
	9	Station waits until tool change command ended.
	10	x
	11	0
2	1	01
	2	Drilling centre
	3	1
	4	Station 2
	5	F
	6	CNC
	7	O
	8	1
	9	No external 24V power supply
	10	x
	11	0

FI command

Output of the currently applied messages of the listed mechanisms for the selected device of the MTCX device group.

BR_AMM3_(1)	**(Single Read)**
BC_AMM3_(1)	**(Cyclic Read)**
BB_AMM3_(1)	**(Break Cyclic Read)**

(1) = Selection list of up to 10 mechanisms [0_1_2_ ... _31]

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM3. The number of rows (1...n=32) depends on the number of requested mechanism messages. Each row, in turn, consists of 11 columns.

Row 1..n	Column 1	...	Column 11

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0...31]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Short text	[max. 54 ASCII characters]
10 =	Detailed text	[x= available, -- = not available]
11 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see AMM5)

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“ Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Example AMM3 Read the current messages of mechanisms 0 and 1 or device address 01.
Assumption: device address 01 with 2 defined mechanisms 0 and 1

FI command		**01_BR_AMM3_0_1**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	0
	4	Station 1
	5	D
	6	CNC
	7	S
	8	79
	9	Station waits until tool change command ended
	10	x
	11	0
2	1	01
	2	Drilling centre
	3	1
	4	Station 2
	5	F
	6	CNC
	7	O
	8	1
	9	No external 24V power supply
	10	x
	11	0

FI command

Output of the currently applied messages of the listed mechanisms for the devices of the MTCX device group.

BR_AMM4_(1)	**(Single Read)**
BC_AMM4_(1)	**(Cyclic Read)**
BB_AMM4_(1)	**(Break Cyclic Read)**

(1) = Selection list of max. 10 mechanisms [Format: x.y]

Format x.y	Value range
x	Device address [00...15]
y	Mechanism number [0...31]

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM4. The number of rows n=512 rows (n=16 devices x 32 mechanisms = 512) depends on the number of requested mechanism messages. Each row, in turn, consists of 11 columns.

Row 1..n	Column 1	...	Column 11

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0...31]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Short text	[max. 54 ASCII characters]
10 =	Detailed text	[x= available, -- = not available]
11 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see AMM5)

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“ Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Example AMM4

Read the current messages of mechanisms 0 and 1 of device address 01 and the messages of mechanism 0 of device address 03.

Assumption: the following device addresses and mechanisms are defined:

- Device address 01 with 2 mechanisms 0 and 1 and
- Device address 03 with one mechanism 0.

FI command		**01_BR_AMM4_01.0_01.1_03.0**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	0
	4	Station 1
	5	D
	6	CNC
	7	S
	8	79
	9	Station waits until tool change command ended
	10	x
	11	0
2	1	01
	2	Drilling centre
	3	1
	4	Station 2
	5	F
	6	CNC
	7	O
	8	1
	9	No external 24V power supply
	10	x
	11	0
3	1	03
	2	Milling centre
	3	0
	4	Camshaft30.40.25.0S
	5	D
	6	CNC
	7	S
	8	71
	9	Circular interpolation
	10	x
	11	0

FI command

Device and mechanism-related output of the detailed text for the selected message number of the devices belonging to the MTCX device group.

BR_AMM5_(1)_(2)_(3) **(Single Read)**

(1) = Mechanism number [0...31]

(2) = Message number [0...600]

(3) = 2 bytes additional information on the message number

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM5. The number of rows n=512 rows (n=16 devices x 32 mechanisms = 512) depends on the number of requested mechanism messages. Each row, in turn, consists of 9 columns.

Row 1..n	Column 1	...	Column 9

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0...31]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Detailed text	[max. 14 rows with max. 78 characters/row]

Example AMM5

Read the detailed text for the required message number 79 of the selected mechanism 0 for the selected device 01.

FI command		**01_BR_AMM5_0_79_0**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	0
	4	Station 1
	5	D
	6	CNC
	7	--
	8	79
	9	Station waits until the active tool change command has been executed completely.

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“ Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Active machine parameter index: API

Designation

API **A**ctive Machine-**P**arameter **I**ndex

Explanation

Information about the active machine parameter sets of all defined devices of the MTCX device group is output. The device address, the index, display of the BOF/GBO, the designation, the size, the date and details of the defined processes of the active machine parameter set are output.

FI command

BR_API1 **(Single Read)**

BC_API1 **(Cyclic Read)**

BB_API1 **(Break Cyclic Read)**

Note: The FI command API1 refers to all devices within the device group. This is why any valid device address can be specified in the command line (see example API1).

Structure of the reply

The following table shows the general structure of the reply of the FI command API1. The reply consists of one to n=16 rows with 8 columns each.

Row 1...n	Column 1	...	Column 8

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Index of the active parameter set	[1...99]
3 =	Display of BOF/GBO	[max. 62 ASCII characters]
4 =	Designation of the parameter set	[max. 32 ASCII characters]
5 =	Size of parameter set [bytes]	[max. 7 ASCII characters]
6 =	Date of creation or last change of parameter set	[8 ASCII characters in format: DD.MM.YY]
7 =	Time of creation or last change of parameter set	[8 ASCII characters in format: HH:MM:SS]
8 =	Details of defined processes	[7 ASCII characters]

Note: In the event that there is no active machine parameter set in the device or the active machine parameter set has been modified, column 1 is assigned the device address and columns 2 to 8 the value [--].

Example API1

Read the information about the active machine parameter sets of all defined devices.

Assumption: the following device addresses of the MTCX device group are defined:

- Device address 00: MTC200-P,
- Device address 01: MTCNC and
- Device address 02: MTVNC.

FI command		**01_BR_API1**
Row	**Column**	**Reply**
1	1	00
	2	1
	3	00 MSD 0209-15 15625 28.01.99 13:29:10 M123456
	4	MSD 0209-15
	5	15625
	6	28.01.99
	7	13:29:10
	8	M123456
2	1	01
	2	88
	3	88 PCI 12.45.12.34 10584 11.11.98 11:11:11 M12
	4	PCI 12.45.12.34
	5	10584
	6	11.11.98
	7	11:11:11
	8	M12
3	1	02
	2	11
	3	11 Lab 5 DRV 24464 01.03.99 14:25:10 M13456
	4	Lab 5 DRV
	5	24464
	6	01.03.99
	7	14:25:10
	8	M13456

Explanation

Output of the device address, index, display of BOF/GBO, designation, size, date and time of creation or the last change and details of defined processes of the active machine parameter set of the selected device.

FI command

BR_API2 (Single Read)

BC_API2 (Cyclic Read)

BB_API2 (Break Cyclic Read)

Structure of the reply

The following table shows the general structure of the reply of the FI command API2. The reply consists of one of 8 columns.

Row 1	Column 1	...	Column 8

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Index of the active parameter set	[1...99]
3 =	Display of BOF/GBO	[max. 62 ASCII characters]
4 =	Designation of the parameter set	[max. 32 ASCII characters]
5 =	Size of parameter set [bytes]	[max. 7 ASCII characters]
6 =	Date of creation or last change of parameter set	[8 ASCII characters in format: DD.MM.YY]
7 =	Time of creation or last change of parameter set	[8 ASCII characters in format: HH:MM:SS]
8 =	Details of defined processes	[7 ASCII characters]

Note: Columns 2 to 8 are assigned the value [--] in the event that there is no active machine parameter set in the device or if the active machine parameter set has been modified.

Example API2

Read the information about the active machine parameter set of device address 02.

Assumption: the following device addresses of the MTCX device group are defined:

- Device address 00: MTC200-P,
- Device address 01: MTCNC and
- Device address 02: MTVNC.

FI command		**00_BR_API2**
Row	**Column**	**Reply**
1	1	00
	2	1
	3	00 MSD 0209-15 15625 28.01.99 13:29:10 M123456
	4	MSD 0209-15
	5	15625
	6	28.01.99
	7	13:29:10
	8	M123456

References For further information about the machine parameters and their classification according to system, process, axis and APR-Sercos parameters, refer to the Indramat documentation:

„MTC200/MT-CNC MCI operating instructions 18VRS“,
Chapter 1.4 Machine parameters,
DOK-MTC200-GBO*MCI*V18-ANW1-DE-E
„Parameter description“,
DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

Active part program message in the NC program (part program message and NC block number): APM

Designation **APM** **A**ctive Part-**P**rogram **M**essage

Explanation The active part program message of the NC block and the NC block number of a CNC process of the MTCX device group are output. Each NC block can contain a part program message which is displayed in the diagnostics menu of the Indramat user interface after execution of the NC block. The part program message in the diagnostics line remains active until it is overwritten by a new part program message (see also: Active part program message in the NC program (NC block number only): APN).

FI command

CR_APM_(1) **(Single Read)**

CC_APM_(1) **(Cyclic Read)**

CB_APM_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The following table shows the general structure of the reply of the FI command APM. One row with two columns for the NC block number and the NC part program message is output.

Value range/meaning of the columns

1 = NC block number of part program message [0000...9999]

2 = Part program message [max. 48 ASCII characters]

Note: If the current NC program does not contain a **part program message**, the result of column 1 is [0000] and the result of column 2 is [--].

Example APM Read the active part program message in CNC process 0 of device address 00.

FI command	**00_CR_APM_0**	
Reply		
Row	**Column 1**	**Column 2**
1	0002	Process data instructions

References For further information about the elements of an NC block and the part program message, refer to the Indramat documentation:

„CNC NC programming instructions V17“, Chapter 2.5 NC word,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active part program message in the NC program (NC block number only): APN

Designation **APN** **A**ctive Part-**P**rogram Message-**N**umber

Explanation The NC block number of the active part program message of a CNC process of the MTCX device group is output. Each NC block can contain a part program message which is displayed in the diagnostics menu of the Indramat user interface after execution of the NC block. The part program message in the diagnostics line remains active until it is overwritten by a new part program message (see also: Active part program message in the NC program (part program message and NC block number): APM).

FI command

CR_APN_(1)	**(Single Read)**
CC_APN_(1)	**(Cyclic Read)**
CB_APN_(1)	**(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply A row with one column for the NC block number of the active part program message is output.

Value range/meaning of the columns 1 = NC block number of part program message [0000...9999]

Note: If the current NC program does not contain a **part program message**, the result of column 1 is [0000].

Example APN Read the NC block number of the active part program message in CNC process 0 of device address 00.

FI command	**00_CR_APN_0**
Reply	
Row	**Column 1**
1	0002

References For further information about the elements of an NC block and the part program message, refer to the Indramat documentation:

„CNC NC programming instructions V17“, Chapter 2.5 NC word, DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Actual position value of an axis: APO

Designation **APO** **A**ctual Axis **PO**sition

Explanation The actual position value of a selected axis of the MTCX device group is read. The FI command „APO1“ returns the position of an axis referred to

the code of the axis meaning. By contrast, the FI command „APO2“ returns the position of an axis referred to the physical axis number.

FI command

Output of the position value of the selected axis of the specified device that is referred to the code of the axis meaning.

CR_APO1_(1)_(2)_(3)	**(Single Read)**
CC_APO1_(1)_(2)_(3)	**(Cyclic Read)**
CB_APO1_(1)_(2)_(3)	**(Break Cyclic Read)**

(1) = CNC process number	[0...6]
(2) = Axis meaning	[0...11; 20]; (see chapter entitled „Data tables“)
(3) = Coordinate system	[1 = Machine coordinates 2 = Program coordinates]

Structure of the reply

The following table shows the general structure of the reply of the FI command APO1. A row with three columns for the axis designation, the position value and the unit according to the process parameter settings is output.

Row 1	**Column 1**	**...**	**Column 3**

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Position value	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and there is no column 3.

Example APO1

Read the current position value of the Z axis in machine coordinates in the CNC process 0 of device address 00.

FI command	**00_CR_APO1_0_2_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-5.9897	[mm]

FI command

Output, referred to the physical axis number, of the position value of the selected axis of the specified device.

CR_APO2_(1)_(2)	**(Single Read)**
CC_APO2_(1)_(2)	**(Cyclic Read)**
CB_APO2_(1)_(2)	**(Break Cyclic Read)**

(1) = Physical axis number	[1...32]
(2) = Coordinate system	[1 = Machine coordinates 2 = Program coordinates]

Structure of the reply

The following table shows the general structure of the reply of the FI command APO2. One row with three columns for the axis designation, position value and the unit according to process parameter settings is output.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Position value	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of columns is [*X], column 2 is [--] and there is not column 3.

Example APO2

Read the current position value of the Z axis (physical axis number = 3) in machine coordinates at device address 00.

FI command	**00_CR_APO2_3_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-5.9897	[mm]

Active NC part program number: APP

Designation

APP **A**ctive **P**art-**P**rogram number

Explanation

The active NC part program number of a CNC process is read.

FI command

CR_APP_(1)	**(Single Read)**
CC_APP_(1)	**(Cyclic Read)**
CB_APP_(1)	**(Break Cyclic Read)**

Structure of the reply

(1) = CNC process number [0...6]

The following table shows the general structure of the reply of the FI command APP. One row with 2 columns for the NC memory and the NC part program number is output.

Row 1	Column 1	Column 2

Value range/meaning of the columns

1 = NC memory [A = memory A; B = memory B]

2 = NC part program number [01...99]

Example APP Read the active NC part program number in the CNC process 0.

FI command	00_CR_APP_0	
Reply		
Row	Column 1	Column 2
1	A	01

References For further information about the NC data structure, refer to the Indramat documentation:

„NC programming instructions V17“,
Chapter 1.2 Program and data organisation,
DOK-MTC200-NC**PRO*V17-ANW1-DE-P.

Actual rapid traversing override: ARO

Designation **ARO** **A**ctual **R**apid **O**verride

Explanation The actual value of rapid traversing override of a CNC process of the MTCX device group is output. This override value is evaluated by the NC in the event of all axis movements that are executed with „G00“. The permissible range of override evaluation by the PLC program is 0 - 255 %.

FI command Output of the current value of the rapid traversing override of a CNC process belonging to the selected device of the MTCX device group.

CR_ARO1_(1) **(Single Read)**

CC_ARO1_(1) **(Cyclic Read)**

CB_ARO1_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The following table shows the general structure of the reply of the FI command ARO. One row with three columns for the designator, the current value of the rapid traversing override and the unit [%] is output.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Designator	[ROV= Rapid Override]
2 = Current rapid override value	[0...255]
3 = Unit	[%]

Note: The permissible range of override evaluation by the PLC program is between 0 and 255 %. The NC limits the axis or process speed to the maximum values defined in the parameters if too large an override is set.

Example ARO1 Read the current value of the rapid override in the CNC process of device address 00.

FI command	**00_CR_ARO1_0**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	ROV	100	%

References For further information about the rapid override, refer to the Indramat documentation:

„CNC/PLC Interface description 18VRS“,
Chapter 2.7 Feed rate and spindle override; rapid override „PxxCROVRD“,
DOK-MTC200-SPS*GWY*V18-ANW1-DE-E.

Selected reference spindle: ASF

Designation **ASF** **A**ctive **S**pindle **F**or Process

Explanation The selected reference spindle of the selected CNC process is output. As there may be several spindles in one CNC process, it is necessary for certain NC functions, e.g. G96 (constant cutting speed), to have them act on others than the first spindle. The following NC functions depend on the selected main spindle:

- G33 Thread cutting
- G63/G64 Thread drilling
- G65 Thread drilling; spindle as guide axis
- G95 Feed rate per revolution and
- G96 Constant cutting speed.

FI command

CR_ASF_(1) **(Single Read)**

CC_ASF_(1) **(Cyclic Read)**

CB_ASF_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The reply of the FI command ASF consists of one row with one column for the selected reference spindle.

Selected reference spindle [S1, S2, S3, *S]

Note: If no reference spindle is selected in the CNC process, the reply of column 1 is [*S].

Example ASF

Read the selected reference spindle in the CNC process 0 of the device address 00. Assumption:

- A rotary axis-compatible main spindle (S1) is defined in CNC process 0,
- The spindle has been selected as the reference spindle with the NC command „SPF 1“ and
- The G function „G96“ is active in the NC program.

FI command	**00_CR_ASF_0**
Reply	
Row	**Column 1**
1	S1

References

For further information about selection of the reference spindle in the NC program, refer to the Indramat documentation:

„CNC NC programming instructions V17“, Application description, Chapter 4.5 Spindle speed; Selection of the reference spindle „SPF“, DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Actual spindle gear: ASG

Designation

ASG **A**ctual **S**pindle **G**ear

Explanation

The actual spindle gear of a CNC process belonging to the selected device of the MTCX device group is read. The control signals for gear speed selection are only evaluated by the CNC if a gear with at least two speeds is entered in the axis parameters.

FI command

CR_ASG_(1)_(2) **(Single Read)**

CC_ASG_(1)_(2) **(Cyclic Read)**

CB_ASG_(1)_(2) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

(2) = Spindle number [1...3]

Structure of the reply

The reply of the FI command ASG consists of one row with two columns for the designator and the current spindle gear speed of the selected CNC process.

Row 1	**Column 1**	**Column 2**

Value range/meaning of the columns

1 = Designator	[g= gear]
2 = Current spindle gear speed	[1...3, -]

Note: If no current spindle gear speed is selected in the CNC process or in the NC program, column 1 is assigned the value [g] and column 2 the value [-].

Example ASG

Read the current spindle gear speed of spindle number 1 in the CNC process of device address 00.

FI command	**00_CR_ASG_0_1**	
Reply		
Row	**Column 1**	**Column 2**
1	g	1

References

For further information about selection of the spindle gear speed in the NC program, refer to the Indramat documentation:

„NC programming instructions V17“,
Chapter 6.2 Modal functions „M“/Gear changeover,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active system fault messages: ASM

Designation

ASM **A**ctive **S**ystem **M**essages

Explanation

The active system fault messages concerning the operation of the complete electrical device are output. Depending on the FI command, the device address, the device name, the message number, the message type, the short text and the detailed text are output.

FI command

Output of the currently applied system fault messages of all active devices in the MTCX device group.

BR_ASM1	**(Single Read)**
BC_ASM1	**(Cyclic Read)**
BB_ASM1	**(Break Cyclic Read)**

Note: The FI command ASM1 refers to all devices within the device group. This is why any valid device address can be specified in the command line (see example ASM1).

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM1. The number of rows (1... n=15) depends on the number of defined devices. Each row consists of 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM1

Read the current system fault messages of all defined devices of the MTCX device group.

Assumption: the following three devices are defined:

- Device address 01
- Device address 07 and
- Device address 10.

FI command		**07_BR_ASM1**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
2	1	07
	2	Milling centre 1
	3	74
	4	F
	5	SLM timeout
	6	X
	7	0
3	1	10
	2	Milling centre 2
	3	1
	4	D
	5	Fault has been remedied.
	6	X
	7	0

FI command

Output of the currently applied system fault message of the selected device belonging to the MTCX device group.

BR_ASM2	**(Single Read)**
BC_ASM2	**(Cyclic Read)**
BB_ASM2	**(Break Cyclic Read)**

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM2. The reply consists of a row with 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM2

Read the current system fault messages of device address 01.

Assumption: the following three devices are defined:

- Device address 01
- Device address 07 and
- Device address 10

FI command		**01_BR_ASM2**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0

FI command

Output of the current system fault messages of the listed devices belonging to the MTCX device group.

BR_ASM3_(1)	**(Single Read)**
BC_ASM3_(1)	**(Cyclic Read)**
BB_ASM3_(1)	**(Break Cyclic Read)**

(1) = Selection list of up to 10 MTCX devices [00_01_ ... _15]

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM3. The number of rows (1... n=15) depends on the number of listed MTCX devices. Each row consists of 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM3

Read the current system fault messages of the selected MTCX devices. Assumption: the following device addresses are defined:

- Device address 01,
- Device address 07 and
- Device address 10.

FI command		**01_BR_ASM3_01_10**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
2	1	10
	2	Milling centre 2
	3	1
	4	D
	5	Fault has been remedied.
	6	X

	7	0

FI command

Output of the current system fault messages of all defined devices (according to the system configuration) of the MTCX device group.

BR_ASM4_(1)	**(Single Read)**
BC_ASM4_(1)	**(Cyclic Read)**
BB_ASM4_(1)	**(Break Cyclic Read)**
(1) = Device group	[MTCX, MISX]

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM4. The number of rows (1... n=15) depends on the number of defined MTCX devices. Each row consists of 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM4

Read the current system fault messages of all defined devices in the MTCX device group. Assumption: the following device addresses are defined:

- Device address 01 and
- Device address 10.

FI command		**01_BR_ASM4_MTCX**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
2	1	10
	2	Milling centre 2
	3	1
	4	D
	5	Fault has been remedied.
	6	X

	7	0

FI command

Device and message number-related output of the detailed text of the currently applied system message.

BR_ASM5_(1)_(2) **(Single Read)**

(1) = Message number [0...150]

(2) = 2 bytes additional information on the message number

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM5. The reply consists of a row with 5 columns for the device address, device name, message number and detailed text.

Row 1..n	Column 1	...	Column 5

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
6 =	Detailed text	[max. 14 Rows with max. 78 characters/row]

Example ASM5

Read the detailed text of the system fault with the message number 74 of the device address 01.

FI command		**03_BR_ASM5_74_0**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	74
	4	F
	5	Replace the SLM module on the PLC card (slot 3).

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“, Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Actual NC block number: ASN

Designation **ASN** **A**ctual **S**equence **N**umber

Explanation The active NC block number of a CNC process belonging to the selected device of the MTCX device group is output.

FI command

CR_ASN_(1)	**(Single Read)**
CC_ASN_(1)	**(Cyclic Read)**
CB_ASN_(1)	**(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The reply of the FI command ASN consists of one row with one column for the active NC block number [N0000...N9999].

Note: If there is not valid NC program, column 1 is assigned the value [N0000].

Example ASN Read the active NC block number of the CNC process 0 of device address 00.

FI command	**00_CR_ASN_0**
Reply	
Row	**Column 1**
1	N0002

References For further information about the structure of an NC block, refer to the Indramat documentation:

„NC programming instructions V17“,
Chapter 2.4 Elements of an NC block,
DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Actual spindle override: ASO

Designation **ASO** **A**ctual **S**pindle **O**verride

Explanation The current value of the spindle override of a CNC process belonging to the MTCX device group is output. The override is valid for non-interpolating axes, i.e. for spindle axes and for magazine axes. The override is evaluated in the NC regardless of mode and takes effect for all axis movements (except in the case of reference travel of digital axes).

FI command CNC process and spindle number-related output of the current value of the override of the selected device belonging to the MTCX device group.

CR_ASO1_(1)_(2)	**(Single Read)**
CC_ASO1_(1)_(2)	**(Cyclic Read)**
CB_ASO1_(1)_(2)	**(Break Cyclic Read)**

(1) = CNC process number	[0...6]
(2) = Spindle number	[1...3]

Structure of the reply

The following table shows the general structure of the reply of the FI command ASO1. One row with three columns for the designator, the current value of the override and the unit [%].

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Designator	[S= Spindle]
2 = Current override value with unit	[0...255]
3 = Unit	[%]

Note: The permissible range of override evaluation by the PLC program is between 0 and 255 %. The NC limits the axis or process speed to the maximum values defined in the parameters if too high an override value is set.

Example ASO1

Read the current value of the override of spindle 1 in the CNC process 0 of device address 00.

FI command	**00_CR_ASO1_0_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	S	60	%

References

For further information about the spindle override, refer to the Indramat documentation:

„CNC/PLC Interface description 18VRS“,
Chapter 2.7 Feed rate and spindle override „PxxCSOVRD“,
DOK-MTC200-SPS*GWY*V18-ANW1-DE-E.

Actual spindle speed: ASS

Designation **ASS** **A**ctual **S**pindle **S**peed

Explanation

The actual spindle speed (axis speed) of a CNC process belonging to the selected device of the MTCX device group is read.

FI command

Output of the current spindle speed of a CNC process referred to the spindle number.

CR_ASS_(1)_(2)	**(Single Read)**

CC_ASS_(1)_(2)	**(Cyclic Read)**
CB_ASS_(1)_(2)	**(Break Cyclic Read)**

(1) = CNC process number [0...6]

(2) = Spindle number [1...3]

Structure of the reply

The following table shows the general structure of the reply of the FI command ASS. One row with three columns for the axis designation, the axis speed and the unit according to process parameter settings is output.

Row 1	**Column 1**	**...**	**Column 3**

Value range/meaning of the columns

1 = Axis designation	[*X, S, S1, S2, S3]
2 = Spindle speed	[according to axis parameter settings]
3 = Unit	[1/min; according to parameter setting]

Note: If no spindle in the selected CNC process is defined, column 1 is assigned the value [*X], column 2 the value [0.0] and column 3 the value [1/min; according to parameter settings].

Example ASS

Read the current axis speed of the 1st spindle in the CNC process 0 of device address 00.

FI command	**00_CR_ASS_0_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	S1	4000.0	[1/min]

References

For further information about the axis-specific speeds, refer to the Indramat documentation:

„MT-CNC Numeric control for multi-axes, multi-process applications”, Chapter 2.6 Maximum path speed, DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

Active tool number: ATN

Designation

ATN **A**ctive **T**ool-**N**umber

Explanation

The number of the active tool of a CNC process belonging to the selected device of the MTCX device group is read.

FI command

CR_ATN_(1)	**(Single Read)**
CC_ATN_(1)	**(Cyclic Read)**
CB_ATN_(1)	**(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply

The reply of the FI command ATN consists of one row with two columns for the designator and the number of the active tool.

	Column 1	**Column 2**

Value range/meaning of the columns

1 = Designator [T = Tool]

2 = Number of the active tool [1...9999999]

Note: If no tool in the selected CNC process is active, column 1 is assigned the value [T] and column 2 the value [0].

Example ATN

Read the number of the active tool in the CNC process 0 of device address 00.

FI command	**00_CR_ATN_0**	
Reply		
Row	**Column 1**	**Column 2**
1	T	4

References

For further information about the structure and elements of the tool data, refer to the Indramat documentation:

„CNC NC programming instructions V17 Application description“, Chapter 11.4 Access to toll data from the NC program „TLD“, DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Active zero offset bank: AZB

Designation

AZB **A**ctive **Z**ero-Offset **B**ank

Explanation

The number of the active zero offset bank of a CNC process belonging to the selected device of the MTCX device group is read.

The zero offsets permit shifting of the source of a coordinate axis, by a specific value referred to the machine datum. These offsets are recorded in the zero offset banks.

FI command

CR_AZB1_(1) **(Single Read)**

CC_AZB1_(1) **(Cyclic Read)**

CB_AZB1_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply

The reply of the FI command AZB1 consists of one row with two columns for the designator (O = Offset) and the number of the active offset bank [0...2].

Example AZB Read the number of the active offset bank in the CNC process of device address 00.

FI command	00_CR_AZB1_0	
Reply		
Row	Column 1	Column 2
1	O	2

References For further information about the use of zero offsets and zero offset banks, refer to the Indramat documentation:

„CNC NC programming instructions V17“, Application description, Chapter 3.5 Zero offsets, zero offset banks „O“, DOK-MTC200-NC**PRO*V17-ANW1-DE-E.

Position setpoint of an axis: CPO

Designation **CPO** **C**ommand **PO**sition

Explanation The current position setpoint of an axis of the MTCX device group selected according to the axis meaning is read.

FI command

CR_CPO1_(1)_(2)_(3)	**(Single Read)**
CC_CPO1_(1)_(2)_(3)	**(Cyclic Read)**
CB_CPO1_(1)_(2)_(3)	**(Break Cyclic Read)**

(1) = CNC process number	[0...6]
(2) = Axis meaning	[0...11; 20] (see chapter entitled, „Data tables“)
(3) = Coordinate system	[1 = Machine coordinates 2 = Program coordinates]

Structure of the reply The following table shows the general structure of the reply of the FI command CPO1. One row with three columns for the axis designation, the position value and the unit according to the process parameter settings is output.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Position value	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and there is no column 3.

Example CPO1 Read the current position setpoint of the Z axis in machine coordinates in the CNC process 0 of device address 00.

FI command	**00_CR_CPO1_0_2_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-5.9897	[mm]

FI command

Output of the position setpoint of an axis of the specified device referred to the physical axis number.

CR_CPO2_(1)_(2)	**(Single Read)**
CC_CPO2_(1)_(2)	**(Cyclic Read)**
CB_CPO2_(1)_(2)	**(Break Cyclic Read)**

(1) = Physical axis number	[1...32]
(2) = Coordinate system	[1 = Machine coordinates 2 = Program coordinates]

Structure of the reply

The following table shows the general structure of the reply of the FI command CPO2. One row with three columns for the axis designation, the position value and the unit according to process parameter settings is output.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Position value	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and there is no column 3.

Example CPO2

Read the current position setpoint of the Z axis (physical axis number = 3) in machine coordinates in the CNC process 0 of device address 00.

FI command	**00_CR_CPO2_3_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-5.9897	[mm]

Control Reset: CRT

Designation

CRT **C**ontrol **R**ese**T**

Explanation

The control reset permits reset of the selected device after a system fault. If there is no system fault, the job is ignored.

FI command

CW_CRT **(Single Write)**

Value to be written

Reset 0 = Trigger reset

Note: The value to be written is passed on in the „DataTransfer“ routine in the „acValue“ parameter.

Structure of the reply

A row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge Reset executed

Example CRT

Trigger a control reset on the selected device.

FI command	**00_CW_CRT** **Value to be written: 0**
Reply	
Row	**Column 1**
1	(P_ACK)

References

For further information about reset of the device, refer to the Indramat documentation:

„PLC programming instructions 18VRS“,
Chapter 11.5 Commissioning / Start,
DOK-CONTRL-SPS*PRO*V18-ANW1-DE-E.

Device axis configuration parameter: DAC

Designation

DAC **D**evice **A**xis **C**onfiguration Parameter

Explanation

The configuration of the device axes that are configured in the active machine parameter set is read. The configuration data of the device axes includes the axis number, the allocated processes, the axis type, the APR number, the APR axis number, the main axis meaning, the secondary axis meaning, the main axis designation, the secondary axis designation and the affiliated axis number.

FI command

Output of the current parameters of all configured device axes.

BR_DAC1 **(Single Read)**

Structure of the reply

The following table shows the general structure of the reply of the FI command DAC1. The number of reply rows [1...32 depending on the CNC process] depends on the number of configured device axes. Each row consists of 11 columns.

Row 1...n	Column 1	...	Column 11

Note: Columns [1...11] are omitted if there is no active machine parameter set in the device.

Value range/meaning of the columns

1 =	Axis number	[1...32]
2 =	Process affiliation	[0...6]
3 =	Allocated processes	[0...6,--]
4 =	Axis type	[see chapter entitled, „Data tables“]
5 =	APR number	[1...5]
6 =	APR axis number	[1...8]
7 =	Main axis meaning	[see chapter entitled „Data tables“]
8 =	Secondary axis meaning	[see chapter entitled „Data tables“]
9 =	Main axis designation	[Xi, Yi, Z,i Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
10 =	Secondary axis designation	[Xi, Yi, Z,i Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
11 =	Allocated axis number	[1...32, --]

References

For further information about configuration of the device axes, refer to the Indramat documentation:

„MT-CNC Numeric control for multi-axes, multi-process applications“, Chapter 1, „System parameters“ and Chapter 3, „Axis parameters“, DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

Example DAC1 Read the current parameters of all configured device axes of the active machine parameter set of device address 00.

Assumption: the following three device axes are defined:

- Digital linear axis (Axis number 1)
- Digital linear axis (Axis number 2)
- Rotary axis-compatible main spindle (Axis number 3).

FI command		**00_BR_DAC1**
Row	**Column**	**Reply**
1	1	1
	2	0
	3	--
	4	81
	5	1
	6	1
	7	0
	8	--
	9	X
	10	--
	11	--
2	1	2
	2	0
	3	--
	4	81
	5	1
	6	2
	7	1
	8	--
	9	Y
	10	--
	11	--
3	1	3
	2	0
	3	--
	4	85
	5	1
	6	4
	7	8
	8	--
	9	S
	10	--
	11	--

FI command

Output of the current parameters of the selected device axis type.

BR_DAC2_(1) **(Single Read)**

(1) = Axis type [1 = only digital axes, 2 = only analog axes]

Structure of the reply

The following table shows the general structure of the reply of the FI command DAC2. The number of reply rows [1...32] depends on the number of configured device axes. Each row consists of 11 columns.

Row 1..n	Column 1	...	Column 11

Note: Columns [1...11] are omitted if there is no active machine parameter set in the device.

Value range/meaning of the columns

1 =	Axis number	[1...32]
2 =	Process affiliation	[0...6]
3 =	Allocated processes	[0...6,--]
4 =	Axis type	[see chapter entitled „Data tables“, axis types]
5 =	APR number	[1...5]
6 =	APR axis number	[1...8]
7 =	Main axis meaning	[see chapter entitled „Data tables“, axis meanings]
8 =	Secondary axis meaning	[see chapter entitled „Data tables“, axis meanings]
9 =	Main axis designation	[Xi, Yi, Z,i Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
10 =	Secondary axis designation	[Xi, Yi, Z,i Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
11 =	Allocated axis number	[1...32, --]

Example DAC2

Read the current parameters of all configured digital device axes of the active machine parameter set of device address 00.
Assumption: a digital linear axis with the axis number 1 is defined.

FI command		**00_BR_DAC2**
Row	**Column**	**Reply**
1	1	1
	2	0
	3	--
	4	81
	5	1
	6	1
	7	0
	8	--
	9	X
	10	--
	11	--

Device configuration parameter: DCP

Designation

DCP **D**evice **C**onfiguration **P**arameter

Explanation

The device configuration parameters that are entered in the active machine parameter set and in the „IND_DEV.INI“ file are output. The configuration parameters of the devices include the device address, the device designation, the device type, the mechanism number, the mechanism name, and the process type.

FI command

Output of the configuration parameters of all defined devices.

BR_DCP1 **(Single Read)**

Note: the FI command DCP1 refers to all defined devices. This is why any valid device address can be specified in the command line (see example DCP1).

Structure of the reply

The following table shows the general structure of the reply of the FI command DCP1. The reply consists of one to n=512 rows (n=16 devices x 32 mechanisms = 512) with 7 columns each.

Row 1...n	Column 1	...	Column 7

Note: Columns [1...7] are omitted if there is no active machine parameter set in the device.

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device designation	[max. 32 ASCII characters]
3 =	Device type	[MTCNC, MTC200-P, MTC200-R, MTVNC, MTRA-P, MTRA-R]
4 =	Mechanism number	[0...31]
5 =	Mechanism name	[max. 28 ASCII characters]
6 =	Process kind	[1= internal, 2 = external process]
7 =	Process type	[1 = CNC process, 2 = PLC process]

Example DCP1 Read the configuration parameters of all defined devices.

Assumption: three devices are defined

- Device address 00 (MTCNC)
- Device address 01 (MTC200-P) and
- Device address 02 (MTC200-P).

FI command		00_BR_DCP1
Row	**Column**	**Reply**
1	1	00
	2	Revolving transfer machine
	3	MTCNC
	4	1
	5	Master
	6	1
	7	2
2	1	01
	2	0
	3	MTC200-P
	4	0
	5	Milling cutter 01
	6	1
	7	1
3	1	02
	2	0
	3	MTC200-P
	4	1
	5	Fräse02
	6	1
	7	1

References For further information about process parameters and their functions and value ranges, refer to the Indramat documentation:

„CNC/PLC Interface description 18VRS Application description, Chapter 3.1 „External mechanisms“
DOK-MTC200-SPS*GWY*V18-ANW1-DE-E.

FI command

Output of the configuration parameters of the selected device.

BR_DCP2 (Single Read)

Structure of the reply

The following table shows the general structure of the reply of the FI command DCP2. The reply consists of one row with 7 columns each.

Row 1	Column 1	...	Column 7

Note: Columns [1...7] of the respective device are omitted if there is no active machine parameter set in the device.

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device designation	[max. 32 ASCII characters]
3 =	Device type	[MTCNC, MTC200-P, MTC200-R, MTVNC, MTRA-P, MTRA-R]
4 =	Mechanism number	[0...31]
5 =	Mechanism name	[max. 28 ASCII characters]
6 =	Process kind	[1= internal, 2 = external process]
7 =	Process type	[1 = CNC process, 2 = PLC process]

Example DCP2

Read the device configuration parameters of the selected device (device address 01).

Assumption: three devices are defined

- Device address 00 (MTCNC)
- Device address 01 (MTC200-P)
- Device address 02 (MTC200-P)

FI command		01_BR_DCP2
Row	**Column**	**Reply**
1	1	01
	2	0
	3	MTC200-P
	4	Machining centre
	5	Fräse01
	6	1
	7	1

References

For further information about process parameters and their functions and value ranges, refer to the Indramat documentation:

„CNC/PLC Interface description 18VRS Application description, Chapter 3.1 „External mechanisms“
DOK-MTC200-SPS*GWY*V18-ANW1-DE-E.

D-correction register: DCR

Designation **DCR** **D**-**C**orrection **R**ecord

Explanation The values of the D-correction registers of the selected CNC process are read or written.

The D-corrections are registers that act additively to the tool geometry data of tool management, i.e. they are additive to the existing geometry registers L1, L2, L3 and R.

99 D-correction numbers are available for each of the seven CNC processes. Each D-correction number therefore contains the registers L1, L2, L3 and R. The value allocations of the D-correction registers are realised via the INDRAMAT user interface (BOF/GBO) or via the function interface.

FI command Output of the D-correction register values of a CNC process of the selected device.

CR_DCR1_(1)_(2){_(3)}	**(Single Read)**
CC_DCR1_(1)_(2){_(3)}	**(Cyclic Read)**
CB_DCR1_(1)_(2){_(3)}	**(Break Cyclic Read)**

(1) = CNC process number	[0...6]
(2) = D-correction number	[1...99]
(3) = Number of the D-correction register	[1=L1, 2=L2, 3=L3, 4=R]; ! Optional !

Note: If the optional parameter is not specified, the values of all D-correction registers (length corrections L1 to L3 and radius correction R) of the selected D-correction numbers are output.

Structure of the reply The number of rows depends on the number of requested D-correction registers.

If the optional parameter has been specified, the reply consists of one row with three columns each for the designator (length corrections L1 to L3 and radius correction R), the value of the requested D-correction register and the unit according to the settings of the process parameters.

If the optional parameter has <u>not</u> been specified, the reply consists of five rows with three columns each.

Row 1...n	Column 1	...	Column 3

Value range/meaning of the columns

1 =	Designator	[L1, L2, L3, R]
2 =	Value of D-correction	[Values formatted according to the process parameter settings]
3 =	Unit	[mm, inch; acc. parameter settings]

Note: If the requested D-correction number or the D-correction register is not assigned a value, the value 0 is output as the reply, formatted according to the settings in the process parameters.

Example DCR1 without optional parameter

Read the values of all D-correction registers at the device address 00 of the CNC process 0.

FI command	**00_CR_DCR1_0_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	L1	1.2586	[mm]
2	L2	3.5892	[mm]
3	L3	0.0000	[mm]
4	R	0.0860	[mm]

Example DCR1 with optional parameter

Read the value of D-correction register 4 (radius correction R) at the device address 00 in the CNC process 0.

FI command	**00_CR_DCR1_0_1_4**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	R	0.0860	[mm]

FI command

Write all D-correction register values of a CNC process of the selected device.

CW_DCR1_(1)_(2) **(Single Write)**

(1) = CNC process number [0...6]

(2) = D-correction number [1...99]

Value to be written

D-correction register [<L1 value > <L2 value > <L3 value > <R value >]

Note: The individual values are separated by blanks and the formatting must conform to the settings of the process parameters (see example DCR1: write D-correction register).

Structure of the reply

One row, with one column for the acknowledgement of successful execution of the FI command is output.

(P_ACK) = **P**ositive **ACK**nowledge Value has been written

Example DCR1
Write D-correction register

Write the following four values into all D-correction registers at the device address 00 of the CNC process 0 of D-correction number 1:

- 1st value L1: 1.2586
- 2nd value L2: 3.5892
- 3rd value L3: 0.0000 and
- 4th value R: 0.0860

Note: The values to be written are communicated in the „acValue“ parameter in the „DataTransfer“ routine and must be separated from one another by a blank „ „.

FI command		**00_CW_DCR1_0_1** **Values to write:** **1.2586 3.5892 0.0000 0.0860**
Row	**Column**	**Reply**
1	1	(P_ACK)

References

For further information about D-corrections, refer to the Indramat documentation:

„MTC200/MT-CNC NC programming instructions 18VRS“,
Chapter 5.7 D-corrections,
DOK-MTC200-NC**PRO*V18-AW01-DE-E.

Identification string of NC/PLC data: DIS

Designation **DIS** **D**ata **I**dentification **S**tring

FI command Output of the parameter identification.

CR_DIS1 (Single Read)

Output of the PLC program identification.

CR_DIS2 (Single Read)

Output of the NC program identification.

CR_DIS3_(1) (Single Read)

(1) = NC memory [1 = NC memory A; 2 = NC memory B]

Output of the tool list identification.

CR_DIS4_(1) (Single Read)

(1) = CNC process number [0...6]

Output of the machine data identification.

CR_DIS5 (Single Read)

Output of the NC program identification.

CR_DIS6_(1)_(2)_(3) (Single Read)

(1) = NC memory [1 = NC memory A;
2 = NC memory B]

(2) = CNC process number [0...6]

(3) = NC program package number [1...99]

Distance to go: DTG

Designation **DTG** **D**istance **T**o **G**o

FI command

CR_DTG1_(1)_(2)_(3)	**(Single Read)**
CC_DTG1_(1)_(2)_(3)	**(Cyclic Read)**
CB_DTG1_(1)_(2)_(3)	**(Break Cyclic Read)**

(1) = CNC process number	[0...6]
(2) = Axis meaning	[0...11; 20]; (see chapter entitled „Data tables“)
(3) = Coordinate system	[1 = Machine coordinates 2 = Program coordinates]

FI command

CR_DTG2_(1)_(2)	**(Single Read)**
CC_DTG2_(1)_(2)	**(Cyclic Read)**
CB_DTG2_(1)_(2)	**(Break Cyclic Read)**

(1) = Physical axis number	[0...6]
(2) = Coordinate system	[1 = Machine coordinates 2 = Program coordinates]

Read device group: DTY

Designation **DTY** **D**evice **TY**pe

FI command **CR_DTY1** **(Single Read)**

Structure of the reply One row is output with one column for the device type.

Example DTY1 Read the device type of the selected device address

FI command	**00_CR_DTY1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	MTC200-P	MTS-P	MTC-P

Position end value: EPO

Designation **EPO** **E**nd**PO**int

Explanation The position end value of a selected axis of the MTCX device group is read. The FI command „EPO1“ returns the position end value of an axis referred to the code of the axis meaning. By contrast, the FI command „EPO2“ returns the position end value of an axis referred to the physical axis number.

FI command Output of the position end value of the selected device referred to the code of the axis meaning.

CR_EPO1_(1)_(2)_(3)	**(Single Read)**
CC_EPO1_(1)_(2)_(3)	**(Cyclic Read)**
CB_EPO1_(1)_(2)_(3)	**(Break Cyclic Read)**

(1) = CNC process number	[0...6]
(2) = Axis meaning	[0...11; 20]; (see chapter entitled „Data tables“)
(3) = Coordinate system	[1 = Machine coordinates 2 = Program coordinates]

Structure of the reply The following table shows the general structure of the reply of the FI command EPO1. One row is output with three columns for the axis designation, the position value and the unit according to process parameter settings.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = End value	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and column 3 is omitted.

Example EPO1 Read the position end value of the Z axis in machine coordinates in the CNC process 0 of device address 00.

FI command	**00_CR_EPO1_0_2_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-5.9897	[mm]

FI command

Output of the position end value of the selected axis of the specified device referred to the physical axis number.

CR_EPO2_(1)_(2)	**(Single Read)**
CC_EPO2_(1)_(2)	**(Cyclic Read)**
CB_EPO2_(1)_(2)	**(Break Cyclic Read)**

(1) = Physical axis number	[1...32]
(2) = Coordinate system	[1 = Machine coordinates 2 = Program coordinates]

Structure of the reply

The following table shows the general structure of the reply of the FI command EPO2. One row with 3 columns for the axis designation, the position value and the unit according to process parameter settings is output.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Position value	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and column 3 is omitted.

Example EPO2

Read the position end value of the Z axis (Physical axis number = 3) in machine coordinates at the device address 00.

FI command	**00_CR_EPO2_3_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-5.9897	[mm]

Global process parameter configuration: GPC

Designation

GPC **G**lobal **P**rocess **C**onfiguration

Explanation

The configuration of the global process parameters of the active machine parameter set of the selected device belonging to the MTCX device group is read.

The global process parameters include programmable and displayed decimal places for distance dimensions, the designation of the CNC process, the basic programming unit, the max. zero offset bank number, D-corrections, basic setting necessary, reference necessary, transformation cartesian/polar possible, jogging of axis results in reset and repositioning of the tool store axis.

Note: The FI commands „GPPx“ (see Global process parameter: GPP) must be given preference over the FI commands „GPCx“ because the speed of access has been optimised in them.

FI command

Output of the configuration of the global process parameters of all defined CNC processes of the active machine parameter set.

BR_GPC1 **(Single Read)**

Structure of the reply

The following table shows the general structure of the reply of the FI command GPC1. The reply consists of one to n=7 rows (n= max. number of defined CNC processes [0...6] = 7) with 12 columns each.

Row 1	Column 1	...	Column 12

Note: Columns [1...12] are dropped if there is no active machine parameter set in the device.

Value range/meaning of the columns

1 =	CNC process number	[0...6]
2 =	Designation of the CNC process	[max. 20 ASCII characters]
3 =	Basic programming unit	[mm, inch]
4 =	Programmed decimal places	[4, 5]
5 =	Displayed decimal places	[0...4]
6 =	Max. zero offset bank number	[0...9]
7 =	D-corrections	[YES, NO]
8 =	Reset position necessary	[YES, NO]
9 =	Reference necessary	[YES, NO]
10 =	Cartesian/polar transformation possible	[YES, NO]
11 =	Axis jogging results in reset	[YES, NO]
12 =	Back-position tool store axis	[YES, NO]

Example GPC1

Read the configuration of the global process parameters of all defined CNC processes of the active machine parameter set of device address 00.
Assumption: the following three CNC processes are defined:

- Slide 1 (CNC process number 0),
- Turret 1 (CNC process number 1) and
- Turret 2 (CNC process number 3).

FI command		**00_BR_GPC1**
Row	**Column**	**Reply**
1	1	0
	2	Slide 1
	3	[mm]
	4	4
	5	3
	6	0
	7	YES
	8	NO
	9	NO
	10	NO
	11	YES
	12	NO
2	1	1
	2	Turret 1
	3	[mm]
	4	4
	5	3
	6	0
	7	NO
	8	YES
	9	YES
	10	NO
	11	YES
	12	NO
3	1	3
	2	Turret 2
	3	[mm]
	4	4
	5	3
	6	0
	7	NO
	8	YES
	9	NO
	10	NO
	11	YES
	12	NO

References

For further information about process parameters and their functions and value ranges, refer to the Indramat documentation:

„MT-CNC Numeric control for multi-axes, multi-process applications“, Chapter 2 „Process parameters“
DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

FI command

CNC process-related output of the configuration of the global process parameters of the active machine parameter set of the selected device.

BR_GPC2_(1) **(Single Read)**

(1) = CNC process number [0...6]

Structure of the reply

The following table shows the general structure of the reply of the FI command GPC2. The reply consists of one row with 12 columns.

Row 1	Column 1	...	Column 12

Note: If there is no active machine parameter set in the device or the selected CNC process is not defined, the columns [1...12] are dropped.

Value range/meaning of the columns

1 =	CNC process number	[0...6]
2 =	Designation of the CNC process	[max. 20 ASCII characters]
3 =	Basic programming unit	[mm, inch]
4 =	Programmed decimal places	[4, 5]
5 =	Displayed decimal places	[0...4]
6 =	Max. zero offset bank number	[0...9]
7 =	D-corrections	[YES, NO]
8 =	Reset position necessary	[YES, NO]
9 =	Reference necessary	[YES, NO]
10 =	Cartesian/polar transformation possible	[YES, NO]
11 =	Axis jogging results in reset	[YES, NO]
12 =	Back-position tool store axis	[YES, NO]

Example GPC2

Read the global process parameters in the CNC process 0 of the active machine parameter set of device address 00.
Assumption: the following three CNC processes are defined:

- Slide 1 (CNC process number 0),
- Turret 1 (CNC process number 1) and
- Turret 2 (CNC process number 3).

FI command		**00_BR_GPC2_0**
Row	**Column**	**Reply**
1	1	0
	2	Slide 1
	3	[mm]
	4	4
	5	3
	6	0
	7	YES
	8	NO
	9	NO
	10	NO
	11	YES
	12	NO

References

For further information about process parameters and their functions and value ranges, refer to the Indramat documentation:

„MT-CNC Numeric control for multi-axes, multi-process applications“, Chapter 2 „Process parameters“
DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

Global process parameter: GPP

Designation **GPP** **G**lobal **P**rocess **P**arameter

Explanation The global process parameters of the active machine parameter set of the selected device belonging to the MTCX device group are read. These include programmable and displayed decimal places for distances, the designation of the CNC process, the basic programming unit and the max. zero offset bank number.

Note: The FI commands „GPPx“ (see Global process parameter configuration: GPC) must be given preference over the FI commands „GPCx“ because the speed of access has been optimised in them.

FI command Output of the configuration of the global process parameters of all defined CNC processes of the active machine parameter set.

BR_GPP1 (Single Read)

Structure of the reply The following table shows the general structure of the reply of the FI command GPC1. The reply consists of one to n=7 rows (n= max. number of defined CNC processes [0...6] = 7) with six columns each.

Row 1	Column 1	...	Column 6

Note: Columns [1...6] are dropped if there is no active machine parameter set in the device.

Value range/meaning of the columns

1 =	CNC process number	[0...6]
2 =	Designation of the CNC process	[max. 20 ASCII characters]
3 =	Basic programming unit	[mm, inch]
4 =	Programmed decimal places	[4, 5]
5 =	Displayed decimal places	[0...4]
6 =	Max. zero offset bank number	[0...9]

Example GPP1

Read the configuration of the global process parameters of all defined CNC processes of the active machine parameter set of device address 00.
Assumption: the following three CNC processes are defined:

- Slide 1 (CNC process number 0),
- Turret 1 (CNC process number 1) and
- Turret 2 (CNC process number 3).

FI command		**00_BR_GPP1**
Row	**Column**	**Reply**
1	1	0
	2	Slide 1
	3	[mm]
	4	4
	5	3
	6	0
2	1	1
	2	Turret 1
	3	[mm]
	4	4
	5	3
	6	0
3	1	3
	2	Turret 2
	3	[mm]
	4	4
	5	3
	6	0

References

For further information about process parameters and their functions and value ranges, refer to the Indramat documentation:

„MT-CNC Numeric control for multi-axes, multi-process applications“,
Chapter 2 „Process parameters“
DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

FI command

CNC process-related output of the configuration of the global process parameters of the active machine parameter set of the selected device.

BR_GPP2_(1) **(Single Read)**

(1) = CNC process number [0...6]

Structure of the reply

The following table shows the general structure of the reply of the FI command GPP2. The reply consists of one row with six columns.

Row 1	Column 1	...	Column 6

Note: If there is no active machine parameter set in the device or the selected CNC process is not defined, the columns [1...6] are dropped.

Value range/meaning of the columns

1 =	CNC process number	[0...6]
2 =	Designation of the CNC process	[max. 20 ASCII characters]
3 =	Basic programming unit	[mm, inch]
4 =	Programmed decimal places	[4, 5]
5 =	Displayed decimal places	[0...4]
6 =	Max. zero offset bank number	[0...9]

Example GPP2

Read the global process parameters in the CNC process 0 of the active machine parameter set of the device address 00.

Assumption: the following three CNC processes are defined:

- Slide 1 (CNC process number 0),
- Turret 1 (CNC process number 1) and
- Turret 2 (CNC process number 3).

FI command		**00_BR_GPP2_0**
Row	**Column**	**Reply**
1	1	0
	2	Slide 1
	3	[mm]
	4	4
	5	3
	6	0

References

For further information about process parameters and their functions and value ranges, refer to the Indramat documentation:

„MT-CNC Numeric control for multi-axes, multi-process applications“, Chapter 2 „Process parameters“
DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

Module assignment of a process: MAP

Designation **MAP** **M**odul **A**ssign of **P**rocess

Explanation The module to which a specific process is assigned is read out of the „Moduldef.ini'„ file. This file is located in the [Drive]:\MT-CNC\CONFIG directory and contains all module configuration data.

The process data is located in the sections: [DeviceAddrX\ModulY\Process] and „X“ stands for the addressed device and „Y“ for the configured module numbers.

FI command Determine the module to which the process belongs. Information is read from the module configuration of the MTCX device group.

BR_MAP1_(1) **(Single Read)**

BC_MAP1_(1) **(Cyclic Read)**

BB_MAP1_(1) **(Break Cyclic Read)**

(1) = Mechanism number [0...31]

Structure of the reply The following table shows the general structure of the reply of the FI command MAP1. One row with one column for the determined module number is output.

Value range of the columns 1 = Module number [0...99]

Example MAP1 Read the module number assigned to the CNC process number 4 out of the module configuration.

Assumption: the module to which CNC process 4 is assigned has the module number 5.

FI command	**03_BR_MAP1_4**
Reply	
Row	**Column 1**
1	5

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the Indramat documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

Device data of the module configuration: MCD

Designation **MCD** **M**odule **C**onfiguration: **D**evice Information

Explanation All device data of the module configuration is read out of the „Moduldef.ini“ file, which is located in the „[Drive]:\MT-CNC\CONFIG“ directory. The device data is located in the [DeviceAddrX] sections, and „X“ stands for the configured device addresses.

FI command Read the device data within the module configuration of the MTCX device group.

BR_MCD1	**(Single Read)**
BC_MCD1	**(Cyclic Read)**
BB_MCD1	**(Break Cyclic Read)**

Note: The FI command MCD1 refers to all devices within the MTCX device group. This is why any valid device address can be specified in the command line (see example MCD1).

Structure of the reply The following table shows the general structure of the reply of the FI command MCD1. The number of rows depends on the number of configured devices. Each row consists of four columns for the device address and the PLC FB names for the provision of setup diagnostics, warning information and start prerequisites.

Row 1	Column 1	...	Column 4

Value range of the columns

1 = Device address	[0...15]
2 = PLC FB name for setup diagnostics	[max. 9 ASCII characters]
3 = PLC FB name for warning information	[max. 9 ASCII characters]
4 = PLC FB name for start prerequisites	[max. 9 ASCII characters]

Example MCD1

Read all device data of the module configuration

Assumption: the following devices of the MTCX device group are defined:

- Device address 01 (MTC200-P)
- Device address 03 (MT-CNC)

FI command	03_BR_MCD1			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	01	PVSetup_1	PVWarn_1	PVStart_1
2	03	PVSetup_3	PVWarn_3	PVStart_3

References

For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the Indramat documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

Module data of the module configuration: MCM

Designation

MCM **M**odul **C**onfiguration: **M**odul Information

Explanation

All module data of a specific device is read out of the „Moduldef.ini“ file, which is located in the „[Drive]:\MT-CNC\CONFIG“ directory and which contains all module configuration data. The module data is located in the [DeviceAddrX\ModulY] sections; „X“ stands for the addressed device and „Y“ for the configured module numbers.

FI command

Read module data out of the module configuration with regard to a device of the MTCX device group.

BR_MCM1	**(Single Read)**
BC_MCM1	**(Cyclic Read)**
BB_MCM1	**(Break Cyclic Read)**

Structure of the reply

The following table shows the general structure of the reply of the FI command MCM1. The number of rows depends on the number of configured modules of a device. Each row consists of four columns for the module number, the module name and PLC FB names for general module faults and module faults.

Row 1	Column 1	...	Column 4

Value range of the columns

1 = Module number	[0...99]
2 = Module name	[max. 28 ASCII characters]
3 = PLC FB name for gen. module faults	[max. 9 ASCII characters]
4 = PLC FB name for messages from module	[max. 9 ASCII characters]

Example MCM1

Read the module data of device 03 from the module configuration:

Assumption: the following modules are defined:

- Module number 5
- Module number 7

FI command	**03_BR_MCM1**			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	5	Module 5 – Milling	PVError_5	PVMsg_5
2	7	Module 7 - Drilling	PVError_7	PVMsg_7

References

For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the Indramat documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P

Process data of the module configuration: MCP

Designation

MCP **M**odul **C**onfiguration: **P**rocess Information

Explanation

All process data of a specific module is read out of the „Moduldef.ini'„ file. This file is located in the „[Drive]:\MT-CNC\CONFIG“ directory and contains all module configuration data. The process data is located in the [DeviceAddrX\ModulY\Process] sections; „X“ stands for the addressed device and „Y“ for the selected module number.

BR_MCP1_(1) **(Single Read)**

BC_MCP1_(1) **(Cyclic Read)**

BB_MCP1_(1) **(Break Cyclic Read)**

(1) = Module number [0...99]

Structure of the reply

The reply of the FI command MCP1 consists of one to n=32 rows with one column for the number of the CNC process or of external mechanisms.

Value range of the columns

(1) = Mechanism number [0...31]

Example MCP1 Read the CNC process number of module 5 of device 03 of the module configuration.

Assumption: the following CNC processes are defined:

- CNC process number 1
- CNC process number 4

FI command	**03_BR_MCP1_5**
Reply	
Row	**Column 1**
1	1
2	4

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the Indramat documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

SFC data of the module configuration: MCS

Designation **MCS** **M**odul **C**onfiguration: **S**FC Information

Explanation All SFC data of a specific module is read out of the „Moduldef.ini'„ file. This file is located in the „[Drive]:\MT-CNC\CONFIG“ directory and contains all module configuration data. The SFC data is contained in the [DeviceAddrX\ModulY\Sfc] sections; „X“ stands for the addressed device and „Y“ for the selected module number.

FI command Read the SFC data with regard to the module of a device out of the module configuration of the MTCX device group.

BR_MCS_(1) **(Single Read)**

BC_MCS_(1) **(Cyclic Read)**

BB_MCS_(1) **(Break Cyclic Read)**

(1) = Module number [0...99]

Structure of the reply The number of rows depends on the number of configured Indrastep sequencers of a module. Each row has a column for the names of the Indrastep sequencers.

Value range of the columns 1 = Name of the Indrastep sequencer [Format W.X.Y.Z]

Format W.X.Y.Z	Value range
W	max. 9 ASCII characters
X	max. 9 ASCII characters ! OPTIONAL !
Y	max. 9 ASCII characters ! OPTIONAL !
Z	max. 9 ASCII characters ! OPTIONAL !

Example MCS1 Read the names of the Indrastep sequencers of module 5 of device 03 of the module configuration.

Assumption: the following Indrastep sequencers are defined:

- ISFB_1
- FB_US.ISFB_3
- FB_US.ISFB_3.SW1.ABBA
- FB_US.ISFB_3.SW1.ABBA

FI command	03_BR_MCS1_5
Reply	
Row	Column 1
1	ISFB_1
2	FB_US.ISFB_3
3	FB_US.ISFB_3.SW1
4	FB_US.ISFB_3.SW1.ABBA

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the INDRAMAT documentation::

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

Maximum feed rate override: MFO

Designation **MFO** **M**aximal **F**eedrate **O**verride

FI command

CR_MFO1_(1) **(Single Read)**

CC_MFO1_(1) **(Cyclic Read)**

CB_MFO1_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The following table shows the general structure of the reply of the FI command MFO1. One row with three columns for the designator, the current value of the maximum feed rate override and the unit [%] is output.

Row 1	Column 1	....	Column 3

Value range/meaning of the columns

1 = Designator	[MAX]
2 = Value of max. feed rate override	[0...100]
3 = Unit	[%]

Example MFO1

Read the maximum value of the feed rate override in the CNC process 0 of device address 00.

FI command	**00_CR_MFO1_0**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	MAX	100	%

Maximum feed rate: MFR

Designation **MFR** **M**aximal **F**eed**R**ate

FI command

CR_MFR_(1) **(Single Read)**

CC_MFR_(1) **(Cyclic Read)**

CB_MFR_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The following table shows the general structure of the reply of the FI command MFR. One row with three columns for the designator, the current value of the feed rate override and the unit according to the parameter settings is output.

Row 1	Column 1	....	Column 3

Value range/meaning of the columns

1 = Designator	[F = Feedrate]
2 = Feedrate-Overrides	[Format according to parameter settings,--]
3 = Unit	[according to parameter settings]

Example MFR Read the value of the feed rate override in the CNC process 0 of device address 00.

FI command	**00_CR_MFR_0**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	F	120000.0	[mm/min]

Maximum rapid traversing override: MRO

Designation **MRO** **M**aximal **R**apid **O**verride

FI command

CR_MRO1_(1) **(Single Read)**

CC_MRO1_(1) **(Cyclic Read)**

CB_MRO1_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The following table shows the general structure of the reply of the FI command MRO1. One row with three columns for the designator, the value of the maximum rapid traversing override and the unit [%] is output.

Row 1	Column 1	....	Column 3

Value range/meaning of the columns

1 = Designator	[RMAX]
2 = Value of max. rapid override	[0...100]
3 = Unit	[%]

Example MRO1 Read the maximum value of the rapid override in the CNC process 0 of device address 00.

FI command	**00_CR_MRO1_0**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	RMAX	100	%

Maximum spindle override: MSO

Designation **MSO** **M**aximal **S**pindle **O**verride

FI command

CR_MSO1_(1)_(2) (Single Read)

CC_MSO1_(1)_(2) (Cyclic Read)

CB_MSO1_(1)_(2) (Break Cyclic Read)

(1) = CNC process number [0...6]

(2) = Number of the spindle [1...3]

Structure of the reply The following table shows the general structure of the reply of the FI command MSO1. One row with three columns for the designator, the value of the maximum spindle override and the unit [%] is output.

Row 1	Column 1	....	Column 3

Value range/meaning of the columns

1 = Designator	[SMAX]
2 = Value of the max. spindle override	[0...100]
3 = Unit	[%]

Example MSO1 Read the maximum value of the spindle override of the 1st spindle in the CNC process 0 of device address 00.

FI command	00_CR_MSO1_0_1		
Reply			
Row	Column 1	Column 2	Column 3
1	SMAX	100	%

Maximum spindle speed: MSS

Designation

MSS **M**aximal **S**pindle **S**peed

FI command

CR_MSS_(1)_(2) **(Single Read)**

CC_MSS_(1)_(2) **(Cyclic Read)**

CB_MSS_(1)_(2) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

(2) = Number of spindle [1...3]

Structure of the reply

The following table shows the general structure of the reply of the FI command MSS. One row with three columns for the designator, the speed and the unit [1/min] is output.

Row 1	Column 1	....	Column 3

Value range/meaning of the columns

1 = Designator [S = Spindle]

2 = Speed [Format according to parameter settings]

3 = Unit [1/min]

Example MSS

Read the maximum value of the speed of the 1st spindle in the CNC process 0 of device address 00.

FI command	00_CR_MSS_0_1		
Reply			
Row	Column 1	Column 2	Column 3
1	S	7500.0	[1/min]

User machine data: MTD

Designation **MTD** **M**achine **T**able **D**ata

FI command Write user machine data.

CW_MTD_(1)_(2)_(3)_(4)_(5) (Single Write)

(1) = Page number [1...299]
(2) = Run variable 1 [-1000 ... +1000]
(3) = Run variable 2 [-1000 ... +1000]
(4) = Element number [1...1000]
(5) = Code of designator [1...13]

Code	Designator	Byte	Min. value	Max. value
1	BOOL	1	0	
2	BYTE	1	0	
3	WORD	2	0	
4	DWORD	4	0	
5	STRING	max. 220 Bytes		
6	SINT	1	- 128	127
7	INT	2	- 32768	32767
8	DINT	4	- 2147483648	2147483647
9	USINT	1	0	255
10	UINT	2	0	65535
11	UDINT	4	0	4294967295
12	REAL	4	- 3.4 E38	3.4 E38
13	DREAL	8	- 1.7 E308	1.7 E308

Value to be written Variable value [according to BOF display format]

Note: The value to be written is communicated in the „DataTransfer“ routine in the „acValue“ parameter.

Structure of the reply One row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge Value transferred successfully

FI command Output of user machine data

CR_MTD_(1)_(2)_(3)_(4)_(5) (Single Read)

(1) = Page number [1...299]
(2) = Run variable 1 [-1000 ... +1000]
(3) = Run variable 2 [-1000 ... +1000]
(4) = Element number [1...1000]
(5) = Designation [1...13]

Reply

Data element
10110100
Read the 13 element of the type UDINT out of MD page 152 via LV1: 0 and LV2: 1 CR_MTD_152_0_1_13_8 150

Status of NC events: NEV

Designation **NEV** **N**C-**EV**ent

FI command Read the status of an NC event.

CR_NEV_(1){_(2)} (Single Read)

(1) = CNC process number [0...6]
(2) = Number of the NC event [0...31] ! Optional !

Note: If the optional parameter is not specified, the status of all NC events is output.

Structure of the reply One row is output; the number of columns depends on the number of requested event statuses. If the optional parameter has not been specified, the reply consists of one row with 32 columns. If the optional parameter has been specified, the reply consists of one row with one column which contains the status [0] or [1] of the requested NC event.

Example NEV Read the status of the 17th event in the CNC process 0 of device address 00.

FI command	00_CR_NEV_0_17
Reply	
Row	Column 1
1	0

FI command Write the status of an NC event.

CW_NEV_(1)_(2) (Single Write)

(1) = CNC process number [0...6]
(2) = Event number [0...31]

Value to be written Status of the NC event 0 = Delete NC event; 1 = Set NC event

Structure of the reply One row with one column for the acknowledgement is output to indicate whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge NC event has been deleted or set

Example NEV Set the 17th NC event in the CNC process 0 at the device address 00.

FI command	**00_CW_NEV_0_17** **Value to be written: 1**
Reply	
Row	**Column 1**
1	(P_ACK)

Selecting the NC memory: NMM

Designation **NMM** **N**C-**M**e**M**ory

Explanation Serves to select the NC memory for NC program editing. In the NC, NC programs are managed in two NC memories. During execution of one NC program, e.g. in NC memory A, it is possible to transfer a different NC program package to NC memory B. Both NC memories (A and B) have an identical structure and are completely equivalent, but only ever one NC memory can be active.

FI command **CW_NMM** **(Single Write)**

Value to be written NC memory [1 = Memory A; 2 = Memory B]

Note: The NC memory can only be selected if the NC is ready to start or is in the reset position. Otherwise, the request is acknowledged with an error. The Value to be written is communicated in the „acValue“ parameter in the „DataTransfer“ routine.

Structure of the reply One row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge NC memory has been selected

Example NMM Select NC memory B on device 00 for NC program editing.

FI command	**00_CW_NMM** **Value to be written: 2**
Reply	
Row	**Column 1**
1	(P_ACK)

References For further information about selection of the NC memory, refer to the Indramat documentation:

„MTC200/MT-CNC Graphical user interface 18VRS“,
Chapter 8.5 Overview of operation: managing NC programs,
DOK-MTC200-GBO*GEN*V18-ANW1-DE-E.

NC parameter: NPA

Designation **NPA** **N**C-**PA**rameter

FI command Output of a parameter line.

BR_NPA1_(1)_(2) (Single Read)

(1) = Parameter set number
(2) = Parameter number

Output of several parameter lines.

BR_NPA2_(1)_(2)_(3) (Single Read)

(1) = Parameter set number
(2) = Parameter number [from]
(3) = Parameter number [to]

Output of a specific element of a parameter line.

BR_NPA3_(1)_(2)_(3) (Single Read)

(1) = Parameter set number
(2) = Parameter number
(3) = Element number

Output of all elements of a parameter line.

BR_NPA4(1)_(2) (Single Read)

(1) = Parameter set number
(2) = Parameter number

NC download commands: NPI, IPP, PPD, PPN, NPC, NPD

By way of example, the following figure shows the sequence that is needed for a complete NC download.

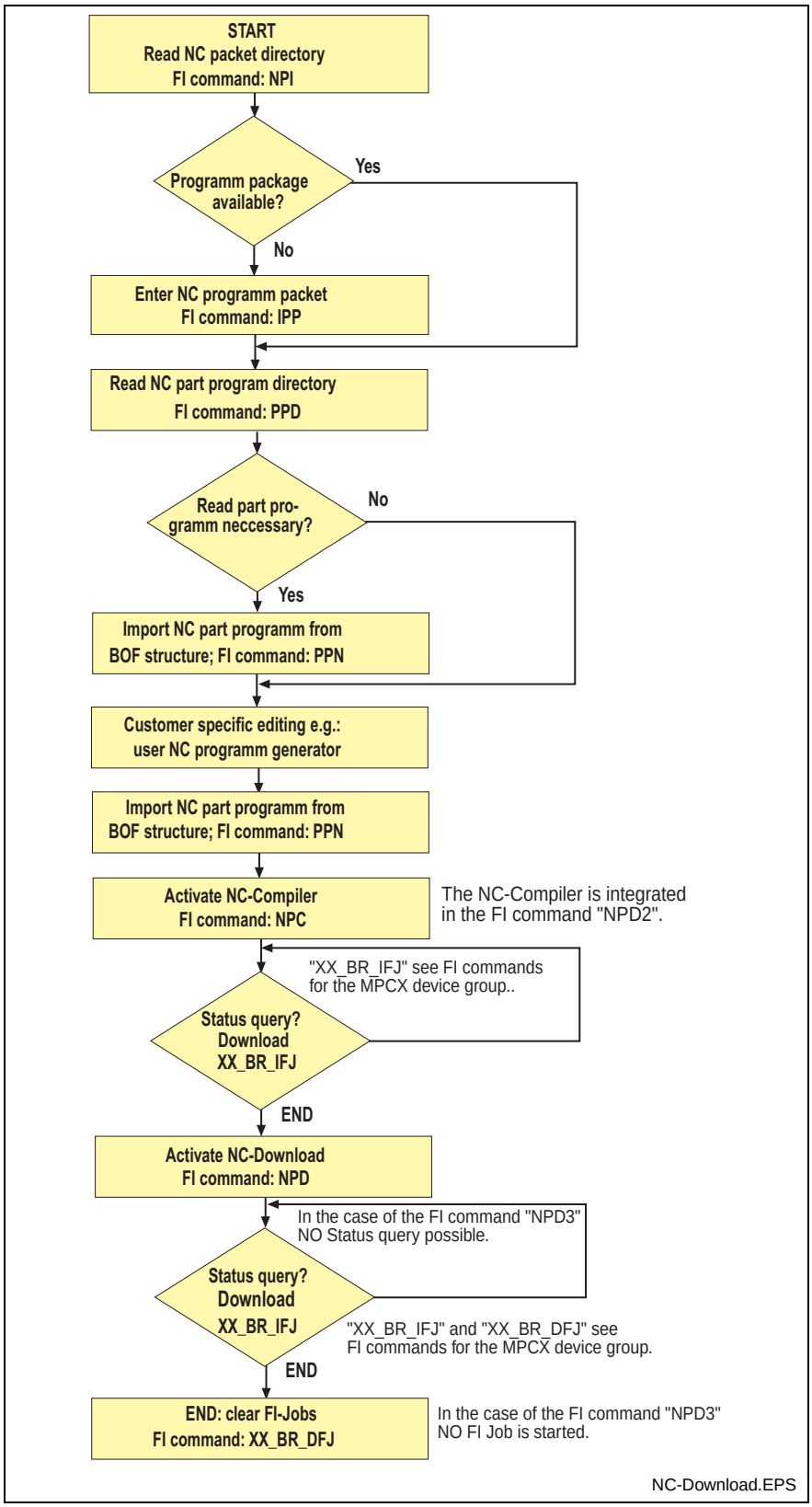


Fig. 6-1: Structure during NC download

Read NC package directory NPI

Designation **NPI** **N**C-**P**ackage D**I**rectory

Explanation Reads the entries of the NC package directories of the user interface.

FI command **BR_NPI** **(Single Read)**

Structure of the reply The following table shows the general structure of the reply of the FI command NPI. The reply consists of one to n=99 rows with five columns each.

Row 1...n	Column 1	...	Column 5

Value range/meaning of the columns

1 =	NC package number	[00...99]
2 =	Designation of the NC package	[max. 32 ASCII characters]
3 =	Length of the NC package	[Byte]
4 =	Date of creation/modification of the NC package	[DD.MM.YY]
5 =	Time of creation/modification of the NC package	[HH:MM:SS]

Example NPI Read the entries in the NC package directory at device address 00.

FI command		**00_BR_NPI**
Row	**Column**	**Reply**
1	1	01
	2	KEY1
	3	3579
	4	16.05.99
	5	10:41:08
2	1	10
	2	KEY2
	3	4589
	4	18.05.99
	5	10:12:10

Insert NC program package: IPP

Designation **IPP** **I**nsert NC-**P**rogram **P**ackage

Explanation Enters an NC program package in the NC package directory of the user interface structure.

FI command **BW_IPP_(1){_(2)}** **(Single Write)**

(1) = Number in the NC package directory [1...99]

(2) = Is the NC package directory entry blank? [0 = no check (default); 1 = with check] ! Optional !

Note: If an NC program package already exists at the selected number of the user interface NC package directory, an error is returned - if the check has been set.

Value to be written Designation of the NC package [max. 32 ASCII characters]

Note: The value to be written is communicated in the „acValue“ parameter as an ASCII value in the „DataTransfer“ routine.

Structure of the reply One row is output with one column for acknowledgement of the FI command sent. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge NC package has been entered.

Example IPP Enter the NC program package with the designation „KEY1“ in the NC package directory of the user interface structure in number 1. Assumption:

- A check is to be made as to whether the selected entry in the NC package directory is blank.

FI command	**00_BW_IPP_1_1** **Value to be written: SCHLUESSEL1**
Reply	
Row	**Column 1**
1	(P_ACK)

Read NC part program directory: PPD

Designation **PPD** **P**art-**P**rogram **D**irectory

FI command Reads the entries of the user interface's NC part program directory.

BR_PPD_(1)_(2) (Single Read)

(1) = Number in the NC package directory [1...99]
(2) = CNC process number [0...6]

Structure of the reply The following table shows the general structure of the reply of the FI command PPD. The reply consists of one to n=99 rows with five columns each.

Row 1...n	Column 1	...	Column 5

Value range/meaning of the columns

1 =	NC part program number	[00...99]
2 =	Designation of the part program	[max. 32 ASCII characters]
3 =	Length of the part program	[Byte]
4 =	Date of creation/last modification of the part program	[DD.MM.YY]
5 =	Time of creation/last modification of the part program	[HH:MM:SS]

Example PPD Read the entries in the NC part program directory of NC package number 1 of the CNC process 0 at device address 00.

FI command		**00_BR_PPD_1_0**
Row	**Column**	**Reply**
1	1	01
	2	TransAM
	3	3579
	4	16.05.99
	5	10:41:08
2	1	10
	2	BMW 3er
	3	4589
	4	18.05.99
	5	10:12:10

Import NC part programs from user interface structure: PPN

Designation **PPN** **P**art-**P**rogram **N**C

FI command Imports NC part programs from the user interface structure to a file.

BR_PPN_(1)_(2)_(3)_(4) (Single Read)

(1) = Number in the NC package directory	[1...99]
(2) = CNC process number	[0...6]
(3) = Number of the NC part program	[1...99]
(4) = NC block numbering	[0 = no number; 1 = with numbers]

Structure of the reply The reply of the FI command PPN consists of one row and column to specify the drive, the directory and the file that contains the NC part program.

Example PPN Import, without NC block numbering, the NC program with the NC part program number 1 of the 2nd NC package of the CNC process 0 at device address 00 to a file.

FI command		**00_BR_PPN_2_0_1_0**
Row	**Column**	**Reply**
1	1	C:\MT-CNC\ANLAGE01\MT_TEMP\T1010001.TMP

Dump from the file „C:\MT-CNC\ANLAGE01\MT_TEMP\T1010001.TMP“:

```
START

SPF 1 [Select reference spindle]
T1 BSR .M6
G90 G96 G54 S1 2000 F5000 M03

G00 X60 Y-30
Z-6 [Feed motion]
G01 X60 Y0 F2000
X5 Y0
Z100
M05 [Spindle halt]

T0 BSR .M6
BST .START
PROGRAM END
```

Export NC part programs from the user interface structure: PPN

Designation **PPN** **P**art-**P**rogram **N**C

FI command Exports an NC part program from the selected file to the user interface structure.

BW_PPN_(1)_(2)_(3)_(4)_(5) **(Single Write)**

(1) = Number in NC package directory	[1...99]
(2) = CNC process number	[0...6]
(3) = Number of NC part program	[1...99]
(4) = NC block numbering	[0 = without number; 1 = with numbers]
(5) = Is NC package directory entry blank?	[0 = without check (default); 1 = with check] ! Optional !
(6) = Complete entry of the directory	[Drive:\..\X.Y]

Note: This FI command does not have a „value to be written“.

Structure of the reply One row is output with one column for acknowledgement of the FI command sent. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge NC part program has been exported.

Example PPN Export the NC program to the NC part program number 1 of the 2nd NC package of the CNC process 0 at device address 00 from the file „C:\Data\T1010001.TMP“.

FI command		**00_BW_PPN_2_0_1_0_1_C:\Data\T1010001.TMP**
Row	**Column**	**Reply**
1	1	(P_ACK)

Activate NC compiler: NPC

Designation **NPC** **N**C-**P**ackage **C**ompiling

FI command Compiles a selected NC package.

BR_NPC1_(1) **(Single Read)**

(1) = Number in the NC package directory [1...99]

Structure of the reply The reply of the FI command NPC1 consists of one row and column for the job ID [01...20] (see chapter entitled „FI commands for the MPCX device group“, IFJ).

Structure of the reply One row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge NC package has been compiled.

Example NPC Compile the 2nd package.

FI command	**00_BR_NPC1_2**
Reply	
Row	**Column 1**
1	01

Activate NC download: NPD

Designation **NPD** **N**C-**P**ackage **D**ownload

FI command Loads the selected NC package **without setup lists** to the specified device.

BW_NPD1_(1)_(2) **(Single Write)**

(1) = NC memory [1= NC memory A, 2= NC memory B]

(2) = Number in the NC package directory [0...6]

Structure of the reply The reply of the FI command NPD1 consists f one row and column for the job ID [01...20] of the download job (see chapter entitled „FI commands for the MPCX device group“, IFJ).

Example NPD1 Load the 2nd NC package (**without setup lists**) into NC memory A of the device with the device address 00.

FI command	**00_BR_NPD1_1_2**
Reply	
Row	**Column 1**
1	02

FI command Loads the selected NC package **with setup lists** into the specified device.

BW_NPD2_(1)_(2) **(Single Write)**

(1) = NC memory [1= NC memory A, 2= NC memory B]

(2) = Number in the NC package directory [0...6]

Structure of the reply The reply of the FI command NPD2 consists of one row and column for the job ID [01...20] of the download job (see chapter entitled „FI commands for the MPCX device group“, IFJ).

Example NPD2 Load the 3rd NC package (**with setup lists**) into NC memory B of the device with the device address 00.

FI command	**00_BR_NPD2_1_2**
Reply	
Row	**Column 1**
1	03

Remarks on the FI command „NPD3“

This FI command has been speed-optimised. It is particularly suitable for transfer of very small NC programs (reference value: up to 100 NC program lines). As the transfer time of small NC programs is less than two seconds, there is no point in a status query. This is why function interface job management has been dispensed with in the case of the FI command „NPD3“ (see chapter entitled „FI commands for the MPCX device group“, IFJ).

FI command

Loads the selected NC package **without setup lists** into the specified device.

BW_NPD3_(1)_(2) **(Single Write)**

(1) = NC memory	[1= NC memory A, 2= NC memory B]
(2) = Number in he NC package directory	[0...6]

Note: No status query is possible in the case of the FI command „NPD3“. The „DataTransfer“ routine dwells for the complete transfer time (dwell time = transfer time).

Example NPD3

Load the 2nd NC package (**without setup lists**) into NC memory A of the device with the device address 00.

FI command	**00_BR_NPD3_1_2**

Selecting the NC program in the active NC memory: NPS

Designation **NPS** **N**C-**P**rogram **S**election

Explanation Serves to select the NC program in the active NC memory for editing. In the NC, NC programs are managed in two NC memories. During execution of one NC program, e.g. in NC memory A, it is possible to transfer a different NC program package to NC memory B. Both NC memories (A and B) have an identical structure and are completely equivalent, but only ever one NC memory can be active.

FI command **CW_NPS_(1)** **(Single Write)**

(1) = CNC process number [0...6]

Value to be written NC program number [1...99]

Note: The NC program can only be selected if there is a valid NC program in the active NC memory. Otherwise, the request is acknowledged with an error. The value to be written is communicated as an ASCII value in the „acValue“ parameter in the „DataTransfer“ routine.

Structure of the reply One row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge NC program has been selected.

Example NPS Select NC program 01 in the active NC memory of CNC process number 0. Assumption:

- There is a valid program package in the active NC memory.

FI command	**00_CW_NPS_0** **Value to be written: 1**
Reply	
Row	**Column 1**
1	(P_ACK)

References For further information about selection of the NC part program and the NC memory, refer to the Indramat documentation:

„MTC200/MT-CNC Graphical user interface 18VRS“,
Chapter 8.5 Overview of operation: managing NC programs,
DOK-MTC200-GBO*GEN*V18-ANW1-DE-E.

Next selected tool number: NTN

Designation **NTN** **N**ext **T**ool-**N**umber

FI command

CR_NTN_(1) **(Single Read)**

CC_NTN_(1) **(Cyclic Read)**

CB_NTN_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply One row with two columns for the designator [T= Tool] and the next selected tool number is output.

Example NTN Read the next selected tool number in the CNC process 0 of device address 00.

FI command	**00_CR_NTN_0**	
Reply		
Row	**Column 1**	**Column 2**
1	T	1

Reading and writing NC variables: NVS

Designation **NVS** **N**C-**V**ariable **S**ingle

FI command Read NC variables.

CR_NVS_(1)_(2){_(3)} **(Single Read)**
CC_NVS_(1)_(2){_(3)} **(Cyclic Read)**
CB_NVS_(1)_(2){_(3)} **(Break Cyclic Read)**

(1) = CNC process number [0...6]
(2) = NC variable number {from} [0...255]
(3) = NC variable number {to} [0...255] !Optional !

Note: If the optional parameter is specified, up to 20 NC variables are output.

Structure of the reply One row with a maximum of 20 columns containing the respective value of the requested NC variable is output.

Note: If the requested NC variable does not exist, [--] is entered in the relevant column.

Example NVS without optional parameter Read the value of the NC variable with the number 1 at the device address 00 in the CNC process 0.

FI command	**00_CR_NVS_0_1**
Reply	
Row	**Column 1**
1	1.111000

Example NVS with optional parameter Read the values of the 1st to 3rd NC variables at the device address 00 in the CNC process 0. Assumption: the 2nd NC variable is not defined.

FI command	**00_CR_NVS_0_1_3**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	1.111000	--	23.100000

FI command Write an NC variable.

CW_NVS_(1)_(2) **(Single Write)**

(1) = CNC process number [0...6]
(2) = NC variable number [0...255]

Value to be written NC variable [Format according to parameter settings]

Note: Only defined NC variables can be written .The value to be written is communicated as an ASCII value in the „acValue“ parameter in the „DataTransfer“ routine.

Structure of the reply One row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge Variable has been written

Example NVS Write the value 1.111000 into the 1st NC variable in the CNC process 0 a the device address 00.

FI command	**00_CW_NVS_0_1** **Value to be written: 1.111000**
Reply	
Row	**Column 1**
1	(P_ACK)

Optimal position distance of axes: OPD

Designation

OPD **O**ptimal **P**osition **D**istance

Explanation

The optimal position distance of a selected axis of the MTCX device group is read. The FI command „OPD1“ returns the position distance of an axis referred to the code of the axis meaning. The FI command „OPD2“, by contrast, returns the position distance of an axis referred to the physical axis number.

FI command

Output of the position distance of the selected axis of the specified device referred to the code of the axis meaning.

CR_OPD1_(1)_(2)	**(Single Read)**
CC_OPD1_(1)_(2)	**(Cyclic Read)**
CB_OPD1_(1)_(2)	**(Break Cyclic Read)**

(1) = CNC process number	[0...6]
(2) = Axis meaning	[0...11; 20]; (see chapter entitled „Data tables“)

Structure of the reply

The following table shows the general structure of the reply of the FI command OPD1. One row is output with three columns for the axis designation, the value of the optimal position distance and the unit according to process parameter settings.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Value	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and column 3 is dropped.

Example OPD1

Read the optimal position distance of the Z axis in the CNC process 0 of device address 00.

FI command	**00_CR_OPD1_0_2**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-5.9897	[mm]

FI command

Output of the optimal position distance of the selected axis of the specified device referred to the physical axis number.

CR_OPD2_(1)	**(Single Read)**
CC_OPD2_(1)	**(Cyclic Read)**
CB_OPD2_(1)	**(Break Cyclic Read)**

(1) = Physical axis number [1...32]

Structure of the reply

The following table shows the general structure of the reply of the FI command OPD2. One row is output with three columns for the axis designation, the position value and the unit according to process parameter settings.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Value	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and column 3 is dropped.

Example OPD2

Read the optimal position distance of the Z axis at device address 00 (physical axis number = 3).

FI command	**00_CR_OPD2_3**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-5.9897	[mm]

Programmed feed rate: PFR

Designation **PFR** **P**rogrammed **F**eed**R**ate

FI command

CR_PFR_(1)	**(Single Read)**
CC_PFR_(1)	**(Cyclic Read)**
CB_PFR_(1)	**(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply The following table shows the general structure of the reply of the FI command PFR. One row is output with three columns for the designator, the current value of the programmed feed rate and the unit according to the parameter settings.

Row 1	Column 1	....	Column 3

Value range/meaning of the columns

1 = Designator	[F = Feedrate]
2 = Feed rate	[Format according to parameter settings]
3 = Unit	[according to parameter settings]

Example PFR Read the programmed feed rate in the CNC process 0 of device address 00.

FI command	**00_CR_PFR_0**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	F	120000.0	[mm/min]

Read an NC block: PPS

Designation **PPS** **P**art **P**rogram **S**equence

Explanation An NC block of an NC part program is read.

FI command **CR_PPS_(1)_(2)_(3)_(4)** **(Single Read)**

(1) = NC memory	[1=Memory A, 2=Memory B]
(2) = CNC process number	[0...6]
(3) = NC part program number	[0...99]
(4) = NC block number	[1...1000]

Structure of the reply One row is output with one column containing the requested NC block.

Example PPS Read the NC block number 2 out of the NC program memory 1 of CNC process number 0 or NC part program number 1.

FI command		**00_CR_PPS_1_0_1_2**
Row	**Column**	**Reply**
1	1	N0002 G01 X50.0000 Y50.0000 Z20.0000 F2500.0

References For further information about the structure of an NC block, refer to the Indramat documentation:

„NC programming instructions V17“,
Chapter 2.4 Elements of an NC block
DOK-MTC200-NC**PRO*V17-ANW1-DE-P.

Programmed spindle speed: PSS

Designation **PSS** **P**rogrammed **S**pindle **S**peed

FI command

CR_PSS_(1)_(2) **(Single Read)**

CC_PSS_(1)_(2) **(Cyclic Read)**

CB_PSS_(1)_(2) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

(2) = Number of spindle [1...3]

Structure of the reply The following table shows the general structure of the reply of the FI command PSS. One row is output with three columns for the axis designation, the speed and the unit [1/min].

Row 1	Column 1	....	Column 3

Value range/meaning of the columns

1 = Designator [S = Spindle]

2 = Speed [Format according to parameter settings]

3 = Unit [1/min]

Example PSS Read the speed of the 1st spindle in the CNC process 0 of device address 00.

FI command	00_CR_PSS_0_1		
Reply			
Row	Column 1	Column 2	Column 3
1	S	7500.0	[1/min]

References For further information about the structure of an NC block, refer to the Indramat documentation:

„NC programming instructions V17“,
Chapter 2.4 Elements of an NC block,
DOK-MTC200-NC**PRO*V17-ANW1-DE-P.

Read and write PLC variables: PVS

Designation **PVS** **P**LC-**V**ariable **S**ingle

Explanation The following types of PLC variables can be read or written:

- BOOL, BYTE, SINT, USINT, CHAR, WORD, INT, UINT, STRING, DWORD, DINT, UDINT, TIME, REAL and imported structures and arrays.

FI command Read PLC variables.

CR_PVS_(1)	**(Single Read)**
CC_PVS_(1)	**(Cyclic Read)**
CB_PVS_(1)	**(Break Cyclic Read)**

(1) = Designator of the PLC variable

Note: Addressing a variable that has not been declared leads to an error message. The data length must not exceed 240 bytes (see also chapter entitled „Programming, Guidelines“).

Example PVS Read the value of the PLC variable with the designator „IB_EXT24“ at device address 00 in the CNC process 0.

Assumption:

- The PLC variable with the designator „IB_EXT24“ is of the „BOOL“ type.

FI command	**00_CR_PVS_IB_EXT24**
Reply	
Row	**Column 1**
1	1

FI command Write a PLC variable.

CW_PVS_(1) (Single Write)

(1) = Designator of the PLC variable

Value to be written PLC variable [Format according to the type in the PLC program]

Note: Only defined PLC variables can be written. Addressing a variable that has not been declared leads to an error message. The data length must not exceed 240 bytes (see also chapter entitled „Programming, Guidelines“). The value to be written is communicated as an ASCII value in the „acValue“ parameter in the „DataTransfer“ routine.

Structure of the reply One row is output with one column for acknowledgement of the FI command sent. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge Variable has been written

Example PVS Write the value 1 into the PLC variable with the designator „IB_EXT24“ at device address 00.

Assumption:

- The PLC variable with the designator „IB_EXT24“ is of the „BOOL“ type.

FI command	**00_CW_PVS_IB_EXT24** **Value to be written: 1**
Reply	
Row	**Column 1**
1	(P_ACK)

Servo lag of an axis: SLA

Designation **SLA** **S**ervo**LA**g

Explanation The current servo lag of a selected axis of the MTCX device group is read. The FI command „SLA1“ returns the servo lag of an axis referred to the code of the axis meaning. The FI command „SLA2“, by contrast, returns the servo lag of an axis referred to the physical axis number.

FI command Output, referred to the code of the axis meaning, of the servo lag of the selected axis belonging to the specified device.

CR_SLA1_(1)_(2)	**(Single Read)**
CC_SLA1_(1)_(2)	**(Cyclic Read)**
CB_SLA1_(1)_(2)	**(Break Cyclic Read)**

(1) = CNC process number	[0...6]
(2) = Axis meaning	[0...11; 20]; (see chapter entitled „Data tables“)

Structure of the reply The following table shows the general structure of the reply of the FI command SLA1. One row is output with three columns for the axis designation, the servo lag and the unit according to process parameter settings.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Servo lag	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and column 3 is dropped.

Example SLA1 Read the servo lag of the Z axis in the CNC process 0 of device address 00.

FI command	**00_CR_SLA1_0_2**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	2.9124	[mm]

FI command

Output, referred to the physical axis number, of the selected axis belonging to the specified device.

CR_SLA2_(1)	**(Single Read)**
CC_SLA2_(1)	**(Cyclic Read)**
CB_SLA2_(1)	**(Break Cyclic Read)**

(1) = Physical axis number [1...32]

Structure of the reply

The following table shows the general structure of the reply of the FI command SLA2. One row is output with three columns for the axis designation, the servo lag and the unit according to process parameter settings.

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Servo lag	[according to process parameter settings]
3 = Unit	[according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and column 3 is dropped.

Example SLA2

Read the servo lag of the Z axis (physical axis number = 3) at device address 00.

FI command	**00_CR_SLA2_3**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	2.9124	[mm]

Sercos parameters: SPA1

Designation **SPA1** **S**ercos **Pa**rameter

Explanation A Sercos parameter of a drive is read or written .In each case, the parameter consists of seven elements and any combination of elements can be selected by element coding.

FI command

BR_SPA1_(1)_(2)_(3)	**(Single Read)**
BC_SPA1_(1)_(2)_(3)	**(Cyclic Read)**
BB_SPA1_(1)_(2)_(3)	**(Break Cyclic Read)**
BW_SPA1_(1)_(2)_(3)	**(Single Write)**

(1) = Drive address	[0...254]
(2) = Parameter No.	in format: X-Y-ZZZZ
(3) = Element coding	[Standard, Advanced-Format]

Parameter No.

Format X-Y-ZZZZ	Value range
X	S = Standard data P = Product data
Y	[0...15] = Parameter set
Z	[0...4095] = Data block No.

Element coding Element coding in the standard format permits requesting of individual elements such as the operation date. If several elements are to be read in a query, the element coding can be ORed in the advanced format, e.g., ORed, the operation date (0x40) and the unit (0x08) result in (0x48) → 48

The advanced format 0x80 has priority over 0x40.

Element	Standard format	Advanced format	Format	Example
Data status	S	01H	Hexadecimal word	0x0000
Name	N	02H	String	NC cycle time (TNcyc)
Attribute	A	04H	Hexadecimal double word	0x60110001
Unit	U	08H	String	µs
Min. input value	L	10H	Decimal word	2000
Max. input value	H	20H	Decimal word	20000
Operation date	D	40H	(see Display of the operation date)	
Min. input value if no list		80H		

Display of the operation date

The display of the operation date depends on the requested parameter number.

Decimal

Decimal values are specified in floating point notation, e.g. 1.5. Leading blanks, zeroes, plus and minus signs and lagging blanks are permitted.

Hexadecimal

Hexadecimal values are represented by „0x...“, e.g. 0x80. A maximum of 8 positions is permitted. Leading or lagging blanks are allowed. Further leading zeroes or plus and minus signs are not allowed.

Binary (max. 32 characters)

Leading or lagging blanks are allowed. The decimal point is the separator: e.g.1111.0000.1010.1100.1111.0000.1010.1100

Note: Further leading zeroes or plus and minus signs are not allowed.

Ident number

The following table shows the general notation of the Ident number:

Format X-Y-ZZZZ	Value range
X	S = Standard data P = Product data
Y	[0...7] = Parameter set
Z	[0...4095] = Data block No.

(see example SPA1/write).

Lists of variable length

Lists always begin with two decimal numbers for the actual length and the maximum length of the list. The length entry refers to the length of the list in the drive, i.e. it designates the number of bytes for storage. The number of elements in the list can be calculated with the aid of the attribute. The list elements are represented according to the attribute. All components of the list are separated from one another by a line feed („\n“).

Example

Parameter S-0-0017, IDN list of all parameters

„400\n400\nS-0-0001\nS-0-0002\n...“

ASCII list

ASCII lists are a special form of the lists of variable length. The individual characters of the string are not separated with „line feed“. a distinction is made between the standard and the advanced formats. In the standard format, the pure character string is returned, whereas in the advanced format the actual and maximum lengths of the list (string) are also transferred.

Example

Parameter S-0-0030, operation date
Standard format: „DKC2.1-SSE-01V09“
Advanced format: „16\n16\nDKC2.1-SSE-01V09“

References

For further information about the functions of the standard and product-specific Sercos parameters (S and P), refer to the Indramat documentation:

„DIAX04 Drive with servo function“, Appendix A Parameter description, DOK-DIAX04-SSE-02VRS**-FKB1-DE-P.

Structure of the reply

The following table shows the general structure of the reply of the FI command SPA1. Row 1 is output both when reading and when writing. Further rows are only output when reading, depending on the element coding.

Note: The first row is dropped completely, if the element coding has been requested in the standard format.

Note: Row 1 is a status line which either contains the Sercos error or indicates successful execution of the FI command. In the event of successful execution, columns 1 and 3 contain the value [0x0000].

The number of the drive that reported the Sercos error is output in the second column of the first row.

Row	Column 1	Column 2	Column 3	Column 4
1	<Sercos error >	<Drive No. Sercos error>	0x0000	0x0000
2	Read: 1st element according to the element coding.			
...	...			
n	Read: (n-1)th element according to the element coding.			

Example SPA1/ read Read the parameters S-0-0003 of the 3rd drive (element coding 0x48)

FI command	**00_BR_SPA1_3_S-0-0003_48**			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	0x0000	0x0000	0x0000	0x0000
2	µs			
3	2000			

Example SPA1/ write Write the Ident number P-0-0037 into the parameter S-0-0305 of the 3rd drive (element coding 0x40).

Technical background:

- the real time status bit 1 should be assigned the trigger status word of the oscilloscope function of a DIAX04 drive.

FI command	**00_BW_SPA1_3_S-0-0305_40** **Value to be written: P-0-0037**			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	0x0000	0x0003	0x0000	0x0000

Change over active Sercos phase: SPH

Designation **SPH** **S**ercos **Ph**ase

Explanation All drives in a Sercos ring are in the same communication phase. The phase status can be read or modified by this command.

FI command

CR_SPH_(1)	**(Single Read)**
CC_SPH_(1)	**(Cyclic Read)**
CB_SPH_(1)	**(Break Cyclic Read)**
CW_SPH_(1)	**(Single Write)**
(1) = Physical axis number	[1...32]; (see chapter entitled „Data tables“)

Value to be written Phase [0...4]

Note: The value to be written is communicated in the „acValue“ parameter in the „DataTransfer“ routine.

Example SPH Read Sercos phase

Read the active phase of the first axis at device address 00.

FI command	00_CR_SPH_1
Reply	
Row	Column 1
1	2

Example SPH Write Sercos phase

Switch over the first axis (write) after phase 4; phase 2 is active.

FI command	00_CW_SPH_1 Value to be written: 4
Reply	
Row	Column 1
1	(P_ACK)

References

For further information about the communication phases, refer to the Indramat documentation:

„DIAX04 Drive with servo function“, General notes on commissioning, DOK-DIAX04-SSE-02VRS**-FKB1-DE-P.

Selected NC part program: SPP

Designation

SPP **S**elected **P**art-**P**rogram number

FI command

CR_SPP_(1) **(Single Read)**

CC_SPP_(1) **(Cyclic Read)**

CB_SPP_(1) **(Break Cyclic Read)**

(1) = CNC process number [0...6]

Structure of the reply

The reply of the FI command SPP consists of one row with two columns for the designator of the NC memory and the number of the selected NC part program.

Row 1	Column 1	Column 2

Value range/meaning of the columns

1 = NC memory	[A = NC memory A; B = NC memory B]
2 = Number of the selected NC part program	[according to process parameter settings]

Example SPP

Read the selected NC part program in CNC process 0 of device address 00.

FI command	00_CR_SPP_0	
Reply		
Row	Column 1	Column 2
1	B	55

Torque: TQE

Designation **TQE** TorQuE

Explanation The torque at a selected axis of the MTCX device group is read. The FI command „TQE1“ returns the torque of an axis referred to the code of the axis meaning. The FI command „TQE2“, by contrast, returns the torque of an axis that is referred to the physical axis number.

FI command Output of the torque of the selected axis that is referred to the code of the axis meaning.

CR_TQE1_(1)_(2)	**(Single Read)**
CC_TQE1_(1)_(2)	**(Cyclic Read)**
CB_TQE1_(1)_(2)	**(Break Cyclic Read)**

(1) = CNC process number	[0...6]
(2) = Axis meaning	[0...11; 20]; (see chapter entitled „Data tables“)

Structure of the reply The following table shows the general structure of the reply of the FI command TQE1. One row is output with three columns for the axis designation, the torque and the unit [%].

Row 1	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Torque	[Format according to process parameter settings]
3 = Unit	[%]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and column 3 is dropped.

Example TQE1 Read the torque at the Z axis in the CNC process 0 of device address 00.

FI command	**00_CR_TQE1_0_2**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-25.6	[%]

FI command

Output of the torque at the selected axis of the specified device that is referred to the physical axis number.

CR_TQE2_(1)	**(Single Read)**
CC_TQE2_(1)	**(Cyclic Read)**
CB_TQE2_(1)	**(Break Cyclic Read)**

(1) = Physical axis number [1...32 see chapter entitled „Data tables“]

Structure of the reply

The following table shows the general structure of the reply of the FI command TQE2. One row is output with three columns for the axis designation, the torque and the unit [%]

Row 1	**Column 1**	**...**	**Column 3**

Value range/meaning of the columns

1 = Axis designation	[according to axis parameter settings]
2 = Torque	[Format according to process parameter settings]
3 = Unit	[%]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and column 3 is dropped.

Example TQE2

Read the torque on the Z axis at the device address 00 (physical axis number = 3).

FI command	**00_CR_TQE2_3**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-25.6	[%]

Data of the zero offset table: ZOD

Designation **ZOD** **Z**ero **O**ffset **D**ata

Explanation The data of the zero offset table can be read and written. The zero offsets make it possible to shift the source of a coordinate axis, referred to the machine datum point, by a specific value. The position of the machine datum is permanently stored in the CNC controller and is not altered by the zero offset.

The following offset types are available in the CNC:

- Programmable absolute zero offset G50,
- Programmable incremental zero offset G51,
- Programmable workpiece zero point G52,
- Adjustable zero offsets G54 ... G59 and
- Adjustable general offset in the zero point table.

With the aid of the zero offsets G50, G51 and G54 ... G59 and the workpiece zero point G52, the coordinate zero point of each NC axis can be placed at any coordinate position within or also outside of the relevant travel area. Thus, it is possible to run an identical NC program at different machine positions. The position of the machine datum of each axis is specified in the drive parameters as the difference with respect to the reference point, and the value entered in the drive parameters correspond to the coordinate value of the reference point in the machine coordinate system.

FI command Write a zero offset.

CW_ZOD_(1)_(2)_(3)_(4)_(5)	**(Single Write)**
(1) = NC memory	[1 = Memory A; 2 = Memory B]
(2) = CNC process number	[0...6]
(3) = Number of the zero point bank	[0...9]
(4) = Offset type	[Code of the offset type]
(5) = Code of axis meanings	[0...8] Axis meanings [9] Torsion angle „PHI“

Value to be written Zero offset [Axes: format according to the parameter settings]
[Torsion angle PHI always in the format Y.XXXX]

Note: The value to be written is communicated in the „acValue“ parameter in the „DataTransfer“ routine.

Offset type

Code	Meaning	Explanation
3	Gen. Offset	Acts in addition to all offset types
4	G54	Adjustable zero offset
....	....	....
9	G59	Adjustable zero offset

Note: The axis meanings can be found in the chapter entitled „Data tables“.

Structure of the reply

One row is output with one column for acknowledgement of the FI command sent. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge Value has been written

Example ZOD

Write in the zero point bank 2 the value of the general offset of the axes X in the NC memory A of the CNC process number 0 at device address 00.

Assumption:

- There is a valid parameter set in the device and
- The axes X, Y, Z are defined.

FI command	**00_CW_ZOD_1_0_2_3_0** **Value to be written: 0.111**
Reply	
Row	**Column 1**
1	(P_ACK)

References

For further information about the zero offsets, refer to the Indramat documentation:

„MTC200/MT-CNC NC programming instructions 18VRS“,
Chapter 3.5 Zero offsets and Chapter 11.3 Reading and writing the zero offset data by the NC program „OTD“,
DOK-MTC200-NC**PRO*V18-AW01-DE-E.

FI command

The zero offset values of all defined axes are output for the selected offset type.

CR_ZOD1_(1)_(2)_(3)_(4)	**(Single Read)**
CC_ZOD1_(1)_(2)_(3)_(4)	**(Cyclic Read)**
CB_ZOD1_(1)_(2)_(3)_(4)	**(Break Cyclic Read)**

(1) = NC memory	[1 = Memory A; 2 = Memory B]
(2) = CNC process number	[0...6]
(3) = Number of the zero point bank	[0...9]
(4) = Offset type	[Code of the offset type]

Offset type

Code	Meaning	Explanation
0	All	Sum of all active offset values
1	G50/G51	Programmable/incremental zero offset
2	G52	Programmable workpiece zero point
3	Gen. offset	Acts in addition to all offset types
4	G54	Adjustable zero offset
....	....	....
9	G59	Adjustable zero offset

Structure of the reply

The following table shows the general structure of the reply of the FI command ZOD1. The reply consists of one to n=10 rows with three columns each for the axis designations, the value of the zero offset and the unit.

Row 1...n	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designations	[acc. axis parameter settings; PHI] [Xi, Yi, Z,i Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
2 = Value	[Format acc. parameter settings] [Torsion angle PHI always in the format Y.XXXX]
3 = Unit	[mm, inch] [Torsion angle PHI: deg]

Example ZOD1

Read in the zero point bank 2 the values of the general offset of all defined axes in the NC memory A of CNC process number 0 at device address 00. Assumption:

- There is a valid parameter set in the device and
- The axes X, Y, Z (temporarily assigned) are defined.

FI command	**00_CR_ZOD1_1_0_2_3**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	X	0.111	[mm]
2	Y	0.000	[mm]
3	*Z	0.000	[mm]
4	PHI	0.0000	[deg]

References

For further information about data of the zero offsets, refer to the Indramat documentation:

„MTC200/MT-CNC NC programming instructions 18VRS“,
Chapter 3.5 Zero offsets and Chapter 11.3 Reading and writing zero offset data by the NC program „OTD“,
DOK-MTC200-NC**PRO*V18-AW01-DE-E.

FI command

Output of all values of the zero offsets for the axes selected in a list.

CR_ZOD2_(1)_(2)_(3)_(4)_(5) **(Single Read)**

CC_ZOD2_(1)_(2)_(3)_(4)_(5) **(Cyclic Read)**

CB_ZOD2_(1)_(2)_(3)_(4)_(5) **(Break Cyclic Read)**

(1) = NC memory	[1 = Memory A; 2 = Memory B]
(2) = CNC process number	[0...6]
(3) = Number of the zero bank	[0...9]
(4) = Offset type	[Code of the offset type]
(5) = Selection list of up to 10 elements	[0...8] Axis meanings [9] Torsion angle „PHI“

Offset type

Code	Meaning	Explanation
0	All	Sum of all active offset values
1	G50/G51	Programmable/incremental zero offset
2	G52	Programmable workpiece zero point
3	Gen. offset	Acts in addition to all offset types
4	G54	Adjustable zero offset
....	....	....
9	G59	Adjustable zero offset

Note: The axis meanings can be found in the chapter entitled „Data tables“.

Structure of the reply

The following table shows the general structure of the reply of the FI command ZOD2. The reply consists of one to n=10 rows with three columns each for the code of the axis meanings, the value of the zero offset and the unit. The number of rows depends on the number of list elements.

Row 1...n	Column 1	...	Column 3

Value range/meaning of the columns

1 = Axis designations	[acc. axis parameter settings; PHI] [Xi, Yi, Z,i Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
2 = Value	[Format acc. parameter settings] [Torsion angle PHI always in the format Y.XXXX]
3 = Unit	[mm, inch] [Torsion angle PHI: deg]

Note: If a requested axis is not defined, the value of columns one to three is [--]. If the axis designation is preceded by „*“, e.g. „*Z“, then this axis is only temporarily assigned to the process (GAX/FAX).

Example ZOD2

Read in the zero point database 2 the values of the general offset of the axes X, Y, Z and U and the torsion angle „PHI“ in the NC memory A of the CNC process number 0 at device address 00.

Assumption:

- There is a valid parameter set in the device and
- The axes X, Y, Z (temporarily assigned) are defined.

FI command	**00_CR_ZOD2_1_0_2_3_0_1_2_3_9**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	X	0.111	[mm]
2	Y	0.000	[mm]
3	*Z	0.000	[mm]
4	--	--	--
5	PHI	0.0000	[deg]

References

For further information about data of the zero offsets, refer to the Indramat documentation:

„MTC200/MT-CNC NC programming instructions 18VRS“,
Chapter 3.5 Zero offsets and Chapter 11.3 Reading and writing zero offset data by the NC program „OTD“,
DOK-MTC200-NC**PRO*V18-AW01-DE-E.

Sercos errors

Code	**Error messages in the serial log**
0x0000	No error in the NC/MMI service channel
0x0001	NC/MMI service channel not open
0x0009	Invalid access to element 0
0x1001	IDN does not exist
0x1009	Invalid access to element 1
0x13E8	Transfer error
0x13E9	Drive does not exist.
0x13EA	Requested data transfer aborted
0x13EB	Requested data channel closed.
0x13EC	System error
0x2001	Name does not exist.
0x2002	Name transferred too short.
0x2003	Name transferred too long.
0x2004	Name cannot be modified.
0x2005	Name currently write-protected.
0x3002	Attribute transferred too short.
0x3003	Attribute transferred too long

Code	Error messages in the serial log
0x3004	Attribute cannot be modified
0x3005	Attribute currently write-protected
0x4001	Unit does not exist
0x4002	Unit transferred too short
0x4003	Unit transferred too long
0x4004	Unit cannot be modified
0x4005	Unit currently write-protected
0x5001	Minimum input value does not exist
0x5002	Minimum input value transferred too short
0x5003	Minimum input value transferred too long
0x5004	Minimum input value cannot be modified
0x5005	Minimum input value currently write-protected
0x6001	Maximum input value does not exist
0x6002	Maximum input value transferred too short
0x6003	Maximum input value transferred too long
0x6004	Maximum input value cannot be modified
0x6005	Maximum input value currently write-protected
0x7002	Data item transferred too short
0x7003	Data item transferred too long
0x7004	Data item cannot be modified
0x7005	Data item currently write-protected
0x7006	Data item less than min. input value
0x7007	Data item more than max. input value
0x7008	Data item not correct
0x7009	Data item password-protected

6.3 FI commands for the MSCX device

The FI commands described in this section apply to the MSCX device group. The device types belonging to this device group are listed in the following table:

Group	Affiliated device type	Address
MSCX	SERCANS-A, SERCANS-P	[00]

Note: Pay attention to the fact that the device address must be placed before the respective FI command (refer also the chapter entitled „Elements of the FI command“), e.g. 00_BR_SPA1_3_S-0-0003_48

Reading and writing Sercos parameters: SPA1

Designation **SPA1** **S**ercos **Pa**rameter

Explanation A Sercos parameter of a drive or a SERCANS parameter is read or written. The parameter consists of 7 elements, and any combination of the elements can be selected by element coding.

FI command

BR_SPA1_(1)_(2)_(3)	**(Single Read)**
BC_SPA1_(1)_(2)_(3)	**(Cyclic Read)**
BB_SPA1_(1)_(2)_(3)	**(Break Cyclic Read)**
BW_SPA1_(1)_(2)_(3)	**(Single Write)**

(1) = Drive address	[0...254]
(2) = Parameter No.	in format: X-Y-ZZZZ
(3) = Element coding	[Standard, advanced format]

Parameter No.

Format X-Y-ZZZZ	Value range
X	S = Standard data P = Product data Y = SERCANS parameter
Y	[0...15] = Parameter set
Z	[0...4095] = Data block No.

Element coding Element coding in the standard format permits requesting of individual elements such as the operation date. If several elements are to be read in a query, the element coding can be ORed in the advanced format, e.g., ORed, the operation date (0x40) and the unit (0x08) result in (0x48) → 48
The advanced format 0x80 has priority over 0x40.

Element	Standard format	Advanced format	Format	Example
Data status	S	01H	Hexadecimal word	0x0000
Name	N	02H	String	NC cycle time (TNcyc)
Attribute	A	04H	Hexadecimal double word	0x60110001
Unit	U	08H	String	µs
Min. input value	L	10H	Decimal word	2000
Max. input value	H	20H	Decimal word	20000
Operation date	D	40H	(see Display of the operation date.)	
Operation date if no list		80H		

Display of the operation date

The display of the operation date depends on the requested parameter number.

Decimal

Decimal values are specified in floating point notation, e.g. 1.5. Leading blanks, zeroes, plus and minus signs and lagging blanks are permitted.

Hexadecimal

Hexadecimal values are represented by „0x...“, e.g. 0x80. A maximum of 8 positions is permitted. Leading or lagging blanks are allowed. Further leading zeroes or plus and minus signs are not allowed.

Binary (max. 32 characters)

Leading or lagging blanks are allowed. The decimal point is the separator:

e.g.1111.0000.1010.1100.1111.0000.1010.1100

Note: Further leading zeroes or plus and minus signs are not allowed.

Ident number

The following table shows the general notation of the Ident number:

Format X-Y-ZZZZ	Value range
X	S = Standard data P = Product data
Y	[0...7] = Parameter set
Z	[0...4095] = Data block No.

(see example SPA1/write).

Lists of variable length

Lists always begin with two decimal numbers for the actual length and the maximum length of the list. The length entry refers to the length of the list in the drive, i.e. it designates the number of bytes for storage. The number of elements in the list can be calculated with the aid of the attribute. The list elements are represented according to the attribute. All components of the list are separated from one another by a line feed („\n“).

Example

Parameter S-0-0017, IDN list of all parameters

„400\n400\nS-0-0001\nS-0-0002\n...“

ASCII list

ASCII lists are a special form of the lists of variable length. The individual characters of the string are not separated with „line feed“. a distinction is made between the standard and the advanced formats. In the standard format, the pure character string is returned, whereas in the advanced format the actual and maximum lengths of the list (string) are also transferred.

Example

Parameter S-0-0030, operation date
Standard format: „DKC2.1-SSE-01V09“
Advanced format: „16\n16\nDKC2.1-SSE-01V09“

Note: When SERCANS parameters are requested any drive address in the range [0..254] is possible.

References

For further information about the functions of the standard and product-specific Sercos parameters (S and P), refer to the Indramat documentation:

„DIAX04 Drive with servo function“, Appendix A Parameter description, DOK-DIAX04-SSE-02VRS**-FKB1-DE-P.

For further information about the functions of the SERCANS system parameters (Y), refer to the Indramat documentation:

„SERCANS SERCOS interface modules“,
Chapter 10 Parameter description,
DOK-SERCAN-SER-V04VRS**-ANW1-DE-P.

Structure of the reply

The following table shows the general structure of the reply of the FI command SPA1. Row 1 is output both when reading and when writing. Further rows are only output when reading, depending on the element coding.

Note: The first row is dropped completely, if the element coding has been requested in the standard format.

Note: Row 1 is a status line which either contains the Sercos/SERCANS Sercos error or indicates successful execution of the FI command. In the event of successful execution, columns 1 and 3 contain the value [0x0000].

The number of the drive that reported the Sercos error or the global SERCANS error is output in the first row, column 2 or column 4. Not all global SERCANS errors can be directly assigned to a drive. In this case, the single result „Drive No.“ is set to 0x0000.

Row	Column 1	Column 2	Column 3	Column 4
1	<Sercos error >	<Drive No. Sercos error>	<Global SERCANS error>	<Drive No. Global SERCANS error>
2	Read: 1st element according to the element coding.			
...	...			
n	Read: (n-1)th element according to the element coding.			

Example SPA1/ read Read the parameters S-0-0003 of the 3rd drive (element coding 0x48)

FI command	**00_BR_SPA1_3_S-0-0003_48**			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	0x0000	0x0000	0x0000	0x0000
2	µs			
3	2000			

Example SPA1/ write Write the Ident number P-0-0037 into the parameter S-0-0305 of the 3rd drive (element coding 0x40).

Technical background:

- the real time status bit 1 should be assigned the trigger status word of the oscilloscope function of a DIAX 04 drive.

FI command	**00_BW_SPA1_3_S-0-0305_40** **Value to be written: P-0-0037**			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	0x0000	0x0003	0x0000	0x0000

Read and write the Sercos phase: SPH

Designation **SPH** **S**ercos **Ph**ase

Explanation All drives in a Sercos ring are in the same communication phase. The phase status can be read or modified by this command.

FI command

CR_SPH	**(Single Read)**
CC_SPH	**(Cyclic Read)**
CB_SPH	**(Break Cyclic Read)**
CW_SPH	**(Single Write)**

Value to be written/ result The allowed phase states are represented by the digits [0...4].

References For further information about the communication phases, refer to the Indramat documentation:

„DIAX04 Drive with servo function“, General notes on commissioning, DOK-DIAX04-SSE-02VRS**-FKB1-DE-P.

Structure of the reply The following table shows the general structure of the reply of the FI command SPH. The number of the drive reporting the Sercos error or the global SERCANS error is output in the first row, column 2 or 4. Not all global SERCANS errors can be directly assigned to a drive. In this case, the single result „Drive No.“ is set to 0x0000.

Row	Column 1	Column 2	Column 3	Column 4
1	<Sercos error >	<Drive No. Sercos error>	<Global SERCANS error>	<Drive No. that is causing the global SERCANS error>
2	Read: current phase Write: previous phase			

Example SPH Switch over (write) to phase 4; phase 2 is active.

FI command	**00_CW_SPH with value 4**			
		Reply		
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	0x0000	0x0000	0x0000	0x0000
2	2			

Determining the current SERCANS error: ASE

Designation **ASE** **A**ctual **S**ERCANS **E**rror

Explanation The global SERCANS error is read; the reply 0x0000 refers to an error-free state of the SERCANS card.

FI command

CR_ASE	**(Single Read)**
CC_ASE	**(Cyclic Read)**
CB_ASE	**(Break Cyclic Read)**

Structure of the reply The following table shows the general structure of the reply of the FI command ASE. The number of the drive reporting the global SERCANS error is output in row 1, column 4. Not all global SERCANS errors can be directly assigned to a drive. In this case, the single result „Drive No.“ is set to 0x0000.

Row 1	Column 1	...	Column 4

Value range/meanings of the columns

1 = 0x0000
2 = 0x0000
3 = Global SERCANS error [see Global SERCANS]
4 = Drive No.

Example ASE Reading the current SERCANS error results in interruption of the fibre-optic ring.

FI command		00_CR_ASE
Row	**Column**	**Reply**
1	1	0x0000
	2	0x0000
	3	0x8009
	4	0x0000

References For further information about the communication phases, refer to the Indramat documentation:

„DIAX04 Drive with servo function“, General notes on commissioning, DOK-DIAX04-SSE-02VRS**-FKB1-DE-P.

Clearing a SERCANS error: CSE

Designation **CSE** **Cl**ear **S**ERCANS **E**rror

Explanation An error reported by the SERCANS card is cleared.

FI command **CW_CSE** **(Single Write)**

Value to be written	The contents of the values parameter are not evaluated.

Structure of the reply The following table shows the general structure of the reply of the FI command CSE. The number of the drive reporting the global SERCANS error is output in row 1, column 4. Not all global SERCANS errors can be directly assigned to a drive. In this case, the single result „Drive No.“ is set to 0x0000.

Row 1	Column 1	...	Column 4

Value range/meanings of the columns

1 = 0x0000
2 = 0x0000
3 = Global SERCANS error [see Global SERCANS]
4 = Drive No..

Example CSE Clearing the SERCANS error.

FI command		00_CW_CSE
Row	**Column**	**Reply**
1	1	0x0000
	2	0x0000
	3	0x0000
	4	0x0000

References For further information about SERCANS errors, refer to the Indramat documentation:

„SERCANS /SERCVME SERCOS interface modules with universal uP interface or VMEbus“, application description, system structure and axis structure.

Global SERCANS errors

The global SERCANS errors are not directly related to the communicated message. These are fatal communication errors that lead to a breakdown of communication with one or several drive(s).

The following global SERCANS errors are defined:

Code	Error messages in the serial log
0x8006	HS timeout
0x8007	2-fold AT failure.
0x8008	Fibre-optic ring not closed.
0x8009	Fibre-optic ring interrupted.
0x800A	„Test operation: zero bit stream or continuous light“. A test mode is set on the SERCANS module to check the optical transmission link of the SERCOS interface.
0xC001	Invalid command control word.
0xC002	IDN is not a command.
0xC003	Command channel can currently not be activated.
0xD001	Drive error (state class 1, S-0-0011).
0xD004	Command cannot be executed in the drive
0xF001	„Configuration error“. An error has on configuration of the setpoint or actual value channel: a) too many setpoints or actual values configured b) the configured setpoints or actual values are not supported
0xF002	„Time slot calculation error „ a) Configured telegrams are too long b) Communication cycle time too short
0xF003	Invalid phase input from the NC
0xF004	„Life counter error“. The controller is no longer accessing the SERCANS DPR.
0xF005	SERCANS: internal error
0xF006	„Copy time too long“. Together, the copy times of the setpoints and actual values is greater than the time between the end of the last AT and the beginning of MDTs.
0xF007	Checksum error (Y parameter).
0xF008	Failure of the input signal SYNCIN
0xF009	Error storing the system parameter or system parameters have been modified. Check of min./max. values unsuccessful
0xF00A	Parameter is write-protected.

Sercos errors

Code	Error messages in the serial log
0x0000	No error in the NC/MMI service channel
0x0001	NC/MMI service channel not open
0x0009	Invalid access to element 0
0x1001	IDN does not exist
0x1009	Invalid access to element 1
0x2001	Name does not exist.
0x2002	Name transferred too short.
0x2003	Name transferred too long.
0x2004	Name cannot be modified.
0x2005	Name currently write-protected.
0x3002	Attribute transferred too short.
0x3003	Attribute transferred too long
0x3004	Attribute cannot be modified
0x3005	Attribute currently write-protected
0x4001	Unit does not exist
0x4002	Unit transferred too short
0x4003	Unit transferred too long
0x4004	Unit cannot be modified
0x4005	Unit currently write-protected
0x5001	Minimum input value does not exist
0x5002	Minimum input value transferred too short
0x5003	Minimum input value transferred too long
0x5004	Minimum input value cannot be modified
0x5005	Minimum input value currently write-protected
0x6001	Maximum input value does not exist
0x6002	Maximum input value transferred too short
0x6003	Maximum input value transferred too long
0x6004	Maximum input value cannot be modified
0x6005	Maximum input value currently write-protected
0x7002	Data item transferred too short
0x7003	Data item transferred too long
0x7004	Data item cannot be modified
0x7005	Data item currently write-protected
0x7006	Data item less than min. input value
0x7007	Data item more than max. input value
0x7008	Data item not correct
0x7009	Data item password-protected

Code	Error messages in the serial log
0x8001	„Service channel currently busy“. The required access is currently not possible because the controller has occupied the service channel.
0x8002	„Fault in the service channel“. The required drive can currently not be accessed.
0x800B	Transfer aborted by the controller because it has to communicate with the same drive (higher priority).
0x800C	Unallowed access (service channel still active); last access not yet ended and a new request has been started.

6.4 FI commands for the MISX device group

The FI commands described in this chapter apply to the MISX device group. The following types and the possible device addresses are listed in this device group:

Group	Affiliated types	Address
MISX	ISP200-P, ISP200-R	[00...63]

Note: Pay attention to the fact that the device address must be placed before the respective FI command, e.g. 00_BR_AMM1 (refer also the chapter entitled „Elements of the FI command“).

Active mechanism messages: AMM

Designation **AMM** **A**ctive **M**echanism **M**essages

Explanation Messages pertaining to active mechanism faults and mechanism diagnostics are output. In the case of the ISP200, these messages refer to mechanism 0. Depending on the FI command, the device address, the device name, the mechanism number, the mechanism name, the message type, the message source, the message number, a short text and a more detailed text are output.

FI command Output of the currently applied mechanism messages of all active devices.

BR_AMM1 **(Single Read)**

BC_AMM1 **(Cyclic Read)**

BB_AMM1 **(Break Cyclic Read)**

Note: The FI command AMM1 refers to all devices within the device group. This is why any valid device address can be specified in the command line (see example AMM1).

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM1. The reply consists of one to a maximum of n=16 rows (n=16 devices x 1 mechanism = 16) with 11 columns each.

Row 1..n	Column 1	...	Column 11

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Short text	[max. 54 ASCII characters]
10 =	Detailed text	[x= available, -- = not available]
11 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see AMM5)

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“ Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Example AMM1

Read the current mechanism messages of all active devices.
Assumption: the following device address and mechanisms are defined:

- Device address 01 with one mechanism 0 and
- Device address 03 with one mechanism 0.

FI command		**03_BR_AMM1**
Row	**Column**	**Reply**
1	1	01
	2	Header station 1
	3	0
	4	Station 1
	5	D
	6	PLC
	7	S
	8	03
	9	EMERGENCY STOP key on station pressed.
	10	x
	11	0
2	1	03
	2	Header station 2
	3	0
	4	Station 2
	5	D
	6	PLC
	7	S
	8	03
	9	EMERGENCY STOP key on station pressed.
	10	x
	11	0

FI command

Output of the currently applied mechanism messages of the selected device.

BR_AMM2	**(Single Read)**
BC_AMM2	**(Cyclic Read)**
BB_AMM2	**(Break Cyclic Read)**

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM2. The reply consists of one row with 11 columns.

Row 1	Column 1	...	Column 11

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[32 ASCII characters]
3 =	Mechanism number	[0]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Short text	[max. 54 ASCII characters]
10 =	Detailed text	[x= available, -- = not available]
11 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see AMM5)

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“ Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Example AMM2

Read the current mechanism messages of device address 01.
Assumption: device address 01 with one mechanism 0.

FI command		**01_BR_AMM2**
Row	**Column**	**Reply**
1	1	01
	2	Header station 3
	3	0
	4	Station 3
	5	D
	6	PLC
	7	S
	8	03
	9	EMERGENCY STOP key on station pressed.
	10	x
	11	0

FI command

Output of the currently applied messages of the listed mechanisms for the selected device of the MISX device group.

BR_AMM3_(1)	**(Single Read)**
BC_AMM3_(1)	**(Cyclic Read)**
BB_AMM3_(1)	**(Break Cyclic Read)**

(1) = Mechanism number [0]

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM3. The number of rows (1...n=32) depends on the number of requested mechanism messages. Each row, in turn, consists of 11 columns.

Row 1..n	Column 1	...	Column 11

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Short text	[max. 54 ASCII characters]
10 =	Detailed text	[x= available, -- = not available]
11 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see AMM5)

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“ Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Example AMM3

Read the current messages of mechanism 0 of device address 01.
Assumption: device address 01 with one mechanism 0.

FI command		**01_BR_AMM3_0_1**
Row	**Column**	**Reply**
1	1	01
	2	Header station 4
	3	0
	4	Station 4
	5	D
	6	PLC
	7	S
	8	03
	9	EMERGENCY STOP key on station pressed.
	10	x
	11	0

FI command

Output of the currently applied messages of the listed mechanisms for the devices of the MISX device group.

BR_AMM4_(1) **(Single Read)**

BC_AMM4_(1) **(Cyclic Read)**

BB_AMM4_(1) **(Break Cyclic Read)**

(1) = Selection list of max. 10 mechanisms [Format: x.y]

Format x.y	**Value range**
x	Device address [00...15]
y	Mechanism number [0]

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM4. The number of rows n=16 rows (n=16 devices x 1 mechanism = 16) depends on the number of requested mechanism messages. Each row, in turn, consists of 11 columns.

Row 1..n	**Column 1**	**...**	**Column 11**

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Short text	[max. 54 ASCII characters]
10 =	Detailed text	[x= available, -- = not available]
11 =	2 bytes additional information	needed to resolve the informa-

on the message number tion „@“ (see AMM5)

Example AMM4 Read the current messages of the mechanisms of device address 01 and device address 03.

Assumption: the following device addresses and mechanisms are defined:

- Device address 01 with 2 mechanism 0 and
- Device address 03 with mechanism 0.

FI command		**01_BR_AMM4_01.0_03.0**
Row	**Column**	**Reply**
1	1	01
	2	Header station 1
	3	0
	4	Station 1
	5	F
	6	PLC
	7	S
	8	03
	9	EMERGENCY STOP key on station pressed.
	10	x
	11	0
2	1	03
	2	Header station 2
	3	0
	4	Station 2
	5	F
	6	PLC
	7	S
	8	03
	9	EMERGENCY STOP key on station pressed.
	10	x
	11	0

FI command

Device and mechanism-related output of the detailed text for the selected message number of the devices belonging to the MISX device group.

BR_AMM5_(1)_(2)_(3) **(Single Read)**

(1) = Mechanism number [0]
(2) = Message number [0...600]
(3) = 2 bytes additional information on the message number

Structure of the reply

The following table shows the general structure of the reply of the FI command AMM5. The number of rows n=16 rows (n=16 devices x 1 mechanism = 16) depends on the number of requested mechanism messages. Each row, in turn, consists of 9 columns.

Row 1..n	Column 1	...	Column 9

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Message type	[F = Fault, D = Diagnostics]
6 =	Message source	[CNC, PLC]
7 =	Message type	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Detailed text	[max. 14 rows with max. 78 characters/row]

Example AMM5

Read the detailed text for the required message number 79 of the selected mechanism 0 for the selected device 01.

FI command		**01_BR_AMM5_0_79_0**
Row	**Column**	**Reply**
1	1	01
	2	Header station 4
	3	0
	4	Station 4
	5	F
	6	PLC
	7	--
	8	03
	9	EMERGENCY STOP key on station pressed.

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“ Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Active system fault messages: ASM

Designation **ASM** **A**ctive **S**ystem **M**essages

Explanation The active system fault messages concerning the operation of the complete electrical device are output. Depending on the FI command, the device address, the device name, the message number, the message type, the short text and the detailed text are output.

FI command Output of the currently applied system fault messages of all active devices in the MISX device group.

BR_ASM1	**(Single Read)**
BC_ASM1	**(Cyclic Read)**
BB_ASM1	**(Break Cyclic Read)**

Note: The FI command ASM1 refers to all devices within the device group. This is why any valid device address can be specified in the command line (see example ASM1).

Structure of the reply The following table shows the general structure of the reply of the FI command ASM1. The number of rows (1... n=15) depends on the number of defined devices. Each row consists of 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM1

Read the current system fault messages of all defined devices of the MISX device group.

Assumption: the following three devices are defined:

- Device address 01
- Device address 07 and
- Device address 10.

FI command		**07_BR_ASM1**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
2	1	07
	2	Drilling station1
	3	74
	4	F
	5	SLM timeout
	6	X
	7	0
3	1	10
	2	Drilling station2
	3	1
	4	D
	5	The fault has been remedied.
	6	X
	7	0

FI command

Output of the currently applied system fault message of the selected device belonging to the MISX device group.

BR_ASM2	**(Single Read)**
BC_ASM2	**(Cyclic Read)**
BB_ASM2	**(Break Cyclic Read)**

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM2. The reply consists of a row with 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM2

Read the current system fault messages of device address 01.

Assumption: the following three devices are defined:

- Device address 01
- Device address 07 and
- Device address 10

FI command		**01_BR_ASM2**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0

FI command

Output of the current system fault messages of the listed devices belonging to the MISX device group.

BR_ASM3_(1)	**(Single Read)**
BC_ASM3_(1)	**(Cyclic Read)**
BB_ASM3_(1)	**(Break Cyclic Read)**

(1) = Selection list of up to 10 MISX devices [00_01_02_ ... _15]

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM3. The number of rows (1... n=15) depends on the number of listed MISX devices. Each row consists of 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM3

Read the current system fault messages of the selected MISX devices. Assumption: the following device addresses are defined:

- Device address 01,
- Device address 07 and
- Device address 10.

FI command		**01_BR_ASM3_01_10**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low
	6	X
	7	0
2	1	10
	2	Drilling centre 2
	3	1
	4	D
	5	The fault has been remedied
	6	X
	7	0

FI command

Output of the current system fault messages of all defined devices (according to the system configuration) of the MISX device group.

BR_ASM4_(1)	**(Single Read)**
BC_ASM4_(1)	**(Cyclic Read)**
BB_ASM4_(1)	**(Break Cyclic Read)**
(1) = Device group	[MISX]

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM4. The number of rows (1... n=15) depends on the number of defined MISX devices. Each row consists of 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM4

Read the current system fault messages of all defined devices in the MISX device group.
Assumption: the following devices are defined:

- Device address 01 and
- Device address 10.

FI command		**01_BR_ASM4_MISX**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
2	1	10
	2	Drilling centre 2
	3	1
	4	D
	5	The fault has been remedied.
	6	X
	7	0

FI command

Device and message number-related output of the detailed text of the currently applied fault message.

BR_ASM5_(1)_(2) **(Single Read)**

(1) = Message number [0...150]

(2) = 2 bytes additional information on the message number

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM5. The reply consists of a row with 5 columns for the device address, device name, message number and detailed text.

Row 1..n	Column 1	...	Column 5

Value range/meaning of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
6 =	Detailed text	[max. 14 Rows with max. 78 characters/row]

Example ASM5

Read the detailed text of the system fault with the message number 74 of the device address 01.

FI command		**03_BR_ASM5_74_0**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	74
	4	F
	5	Replace the SLM module on the PLC card (slot 3).

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“, Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Control Reset: CRT

Designation **CRT** **C**ontrol **R**ese**T**

Explanation The control reset permits reset of the selected device after a system fault. If there is no system fault, the job is ignored.

FI command **CW_CRT** **(Single Write)**

Value to be written Reset 0 = Trigger reset

Note: The value to be written is passed on in the „DataTransfer“ routine in the „acValue“ parameter.

Structure of the reply A row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge Reset executed

Example CRT Trigger a control reset on the selected device.

FI command	**00_CW_CRT** **Value to be written: 0**
Reply	
Row	**Column 1**
1	(P_ACK)

References For further information about reset of the device, refer to the Indramat documentation:

„PLC programming instructions 18VRS“,
Chapter 11.5 Commissioning / Start,
DOK-CONTRL-PLC*PRO*V18-ANW1-DE-E.

Identification string of PLC data: DIS

Designation **DIS** **D**ata **I**dentification **S**tring

FI command Output of the PLC program identification.

CR_DIS2 **(Single Read)**

Read device group: DTY

Designation **DTY** **D**evice **TY**pe

FI command **CR_DTY1** **(Single Read)**

Structure of the reply One row is output with one column for the device type.

Example DTY1 Read the device type of the selected device address

FI command	**00_CR_DTY1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	MTC200-P	MTS-P	MTC-P

Module assignment of a process: MAP

Designation **MAP** **M**odul **A**ssign of **P**rocess

Explanation The module to which a specific process is assigned is read out of the „Moduldef.ini'„ file. This file is located in the [Drive]:\MT-CNC\CONFIG directory and contains all module configuration data.

The process data is located in the sections:
[DeviceAddrX\ModulY\Process]
and „X“ stands for the addressed device and „Y“ for the configured module numbers.

FI command Determine the module to which the process belongs. Information is read from the module configuration of the MISX device group.

BR_MAP1_(1) **(Single Read)**

BC_MAP1_(1) **(Cyclic Read)**

BB_MAP1_(1) **(Break Cyclic Read)**

(1) = Mechanism number [0...31]

Structure of the reply The following table shows the general structure of the reply of the FI command MAP1. One row with one column for the determined module number is output.

Value range of the columns 1 = Module number [0...99]

Example MAP1 Read the module number assigned to the CNC process number 4 out of the module configuration.
Assumption: the module to which CNC process 4 is assigned has the module number 5.

FI command	**03_BR_MAP1_4**
Reply	
Row	**Column 1**
1	5

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the following INDRAMAT documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

Device data of the module configuration: MCD

Designation **MCD** **M**odul **C**onfiguration: **D**evice Information

Explanation All device data of the module configuration is read out of the „Moduldef.ini“ file, which is located in the „[Drive]:\MT-CNC\CONFIG“ directory. The device data is located in the [DeviceAddrX] sections, and „X“ stands for the configured device addresses.

FI command Read the device data within the module configuration of the MISX device group.

BR_MCD1	**(Single Read)**
BC_MCD1	**(Cyclic Read)**
BB_MCD1	**(Break Cyclic Read)**

Note: The FI command MCD1 refers to all devices within the MISX device group. This is why any valid device address can be specified in the command line (see example MCD1).

Structure of the reply The following table shows the general structure of the reply of the FI command MCD1. The number of rows depends on the number of configured devices. Each row consists of four columns for the device address and the PLC FB names for the provision of setup diagnostics, warning information and start prerequisites.

Value range of the columns

1 = Device address	[0...15]
2 = PLC FB name for setup diagnostics	[max. 9 ASCII characters]
3 = PLC FB name for warning information	[max. 9 ASCII characters]
4 = PLC FB name for start prerequisites	[max. 9 ASCII characters]

Example MCD1 Read all device data of the module configuration

Assumption: the following devices of the MISX device group are defined:

- Device address 01 (MTC200-P)
- Device address 03 (MT-CNC)

FI command	**03_BR_MCD1**			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	01	PVSetup_1	PVWarn_1	PVStart_1
2	03	PVSetup_3	PVWarn_3	PVStart_3

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the following INDRAMAT documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

Module data of the module configuration: MCM

Designation **MCM** **M**odul **C**onfiguration: **M**odul Information

Explanation All module data of a specific device is read out of the „Moduldef.ini“ file, which is located in the „[Drive]:\MT-CNC\CONFIG“ directory and which contains all module configuration data. The module data is located in the [DeviceAddrX\ModulY] sections; „X“ stands for the addressed device and „Y“ for the configured module numbers.

FI command Read module data out of the module configuration with regard to a device of the MISX device group.

BR_MCM1	**(Single Read)**
BC_MCM1	**(Cyclic Read)**
BB_MCM1	**(Break Cyclic Read)**

Structure of the reply The following table shows the general structure of the reply of the FI command MCM1. The number of rows depends on the number of configured modules of a device. Each row consists of four columns for the module number, the module name and PLC FB names for general module faults and module faults.

Row 1	Column 1	...	Column 4

Value range of the columns

1 = Module number	[0...99]
2 = Module name	[max. 28 ASCII characters]
3 = PLC FB name for gen. module faults	[max. 9 ASCII characters]
4 = PLC FB name for messages from module	[max. 9 ASCII characters]

Example MCM1 Read the module data of device 03 from the module configuration:

Assumption: the following modules are defined:

- Module number 5
- Module number 7

FI command	**03_BR_MCM1**			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	5	Module 5 – Milling	PVError_5	PVMsg_5
2	7	Module 7 - Drilling	PVError_7	PVMsg_7

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the following INDRAMAT documentation:

Diagnostic and message system for HMI systems ProVi,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

Process data of the module configuration: MCP

Designation **MCP** **M**odul **C**onfiguration: **P**rocess Information

Explanation All process data of a specific module is read out of the „Moduldef.ini'„ file. This file is located in the „[Drive]:\MT-CNC\CONFIG“ directory and contains all module configuration data. The process data is located in the [DeviceAddrX\ModulY\Process] sections; „X“ stands for the addressed device and „Y“ for the selected module number.

BR_MCP1_(1) **(Single Read)**

BC_MCP1_(1) **(Cyclic Read)**

BB_MCP1_(1) **(Break Cyclic Read)**

(1) = Module number [0...99]

Structure of the reply The reply of the FI command MCP1 consists of one to n=32 rows with one column for the number of the CNC process or of external mechanisms.

Value range of the columns (1) = Mechanism number [0...31]

Example MCP1 Read the CNC process number of module 5 of device 03 of the module configuration.

Assumption: the following CNC processes are defined:

- CNC process number 1
- CNC process number 4

FI command	**03_BR_MCP1_5**
Reply	
Row	**Column 1**
1	1
2	4

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the following INDRAMAT documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

SFC data of the module configuration: MCS

Designation **MCS** **M**odul **C**onfiguration: **S**FC Information

Explanation All SFC data of a specific module is read out of the „Moduldef.ini'„ file. This file is located in the „[Drive]:\MT-CNC\CONFIG“ directory and contains all module configuration data.
The SFC data is contained in the [DeviceAddrX\ModulY\Sfc] sections; „X“ stands for the addressed device and „Y“ for the selected module number.

FI command Read the SFC data with regard to the module of a device out of the module configuration of the MISX device group.

BR_MCS_(1) **(Single Read)**

BC_MCS_(1) **(Cyclic Read)**

BB_MCS_(1) **(Break Cyclic Read)**

(1) = Module number [0...99]

Structure of the reply The number of rows depends on the number of configured Indrastep sequencers of a module. Each row has a column for the names of the Indrastep sequencers.

Value range of the columns 1 = Name of the Indrastep sequencer [Format W.X.Y.Z]

Format W.X.Y.Z	Value range
W	max. 9 ASCII characters
X	max. 9 ASCII characters ! OPTIONAL !
Y	max. 9 ASCII characters ! OPTIONAL !
Z	max. 9 ASCII characters ! OPTIONAL !

Example MCS1 Read the names of the Indrastep sequencers of module 5 of device 03 of the module configuration.
Assumption: the following Indrastep sequencers are defined:

- ISFB_1
- FB_US.ISFB_3
- FB_US.ISFB_3.SW1.ABBA
- FB_US.ISFB_3.SW1.ABBA

FI command	03_BR_MCS1_5
Reply	
Row	Column 1
1	ISFB_1
2	FB_US.ISFB_3
3	FB_US.ISFB_3.SW1
4	FB_US.ISFB_3.SW1.ABBA

Read and write PLC variables: PVS

Designation **PVS** **P**LC-**V**ariable **S**ingle

Explanation The following types of PLC variables can be read or written:

- BOOL, BYTE, SINT, USINT, CHAR, WORD, INT, UINT, STRING, DWORD, DINT, UDINT, TIME, and imported structures and arrays.

FI command Read PLC variables.

CR_PVS_(1)	**(Single Read)**
CC_PVS_(1)	**(Cyclic Read)**
CB_PVS_(1)	**(Break Cyclic Read)**

(1) = Designator of the PLC variable

Note: Addressing a variable that has not been declared leads to an error message. The data length must not exceed 240 bytes (see also chapter entitled „Programming, Guidelines“).

Example PVS Read the value of the PLC variable with the designator „IB_EXT24“ at device address 00 in the CNC process 0.

Assumption:

- The PLC variable with the designator „IB_EXT24“ is of the „BOOL“ type.

FI command	**00_CR_PVS_IB_EXT24**
Reply	
Row	**Column 1**
1	1

FI command Write a PLC variable.

CW_PVS_(1) **(Single Write)**

(1) = Designator of the PLC variable

Value to be written PLC variable [Format according to the type in the PLC program]

Note: Only defined PLC variables can be written. Addressing a variable that has not been declared leads to an error message. The data length must not exceed 240 bytes (see also chapter entitled „Programming, Guidelines“). The value to be written is communicated as an ASCII value in the „acValue“ parameter in the „DataTransfer“ routine.

Structure of the reply One row is output with one column for acknowledgement of the FI command sent. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge Variable has been written.

Example PVS Write the value 1 into the PLC variable with the designator „IB_EXT24“ at device address 00.

Assumption:

- The PLC variable with the designator „IB_EXT24“ is of the „BOOL“ type.

FI command	**00_CW_PVS_IB_EXT24** **Value to be written: 1**
Reply	
Row	**Column 1**
1	(P_ACK)

6.5 FI commands for the MTAX device group

The FI commands described in this chapter apply to the MTAX device group. The following types and the possible device addresses are listed in this device group:

Group	Affiliated types	Address
MTAX	MTA200-P (ANDRON controller)	[00...63]

Note: Pay attention to the fact that the device address must be placed before the respective FI command, e.g. 00_BR_ASM1 (refer also the chapter entitled „Elements of the FI command“).

Actual position value of an axis: APO

Designation **APO** **A**ctual Axis **PO**sition

Explanation The actual position value of a selected axis of the MTAX device group is read. The FI command „APO2“ returns the position of an axis referred to the physical axis number.

FI command Output of the position value of the selected axis of the specified device that is referred to the physical axis number.

CR_APO2_(1)_(2) **(Single Read)**

CC_APO2_(1)_(2) **(Cyclic Read)**

CB_APO2_(1)_(2) **(Break Cyclic Read)**

(1) = Physical axis number [1...32]

(2) = Coordinate system [1 = Machine coordinates
2 = Program coordinates]

Structure of the reply The following table shows the general structure of the reply of the FI command APO2. A row with three columns for the axis designation, the position value and the unit according to the process parameter settings is output.

Row 1	Column 1	...	Column 3

Value range/meanings of the columns

1 = Axis designation [according to axis parameter settings]

2 = Position value [according to process parameter settings]

3 = Unit [according to process parameter settings: mm, inch]

Note: If the specified axis is not defined in the CNC process, the reply of column 1 is [*X], column 2 is [--] and there is no column 3.

ExampleAPO2 Read the current position value of the Z (physical axis number = 3) axis in machine coordinates at the device address 00.

FI command	**00_CR_APO2_3_1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	Z	-5.9897	[mm]

Active system fault messages: ASM

Designation **ASM** **A**ctive **S**ystem **M**essages

Explanation The active system fault messages concerning the operation of the complete electrical device are output. Depending on the FI command, the device address, the device name, the message number, the message type, the short text and the detailed text are output. Accesses to system fault messages refer only to the PLC portion (ISP200).

FI command Output of the currently applied system fault messages of all active devices in the MTAX device group.

BR_ASM1	**(Single Read)**
BC_ASM1	**(Cyclic Read)**
BB_ASM1	**(Break Cyclic Read)**

Note: The FI command ASM1 refers to all devices within the device group. This is why any valid device address can be specified in the command line (see example ASM1).

Structure of the reply The following table shows the general structure of the reply of the FI command ASM1. The number of rows (1... n=15) depends on the number of defined devices. Each row consists of 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	**Column 1**	**...**	**Column 7**

Value range/meanings of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM1

Read the current system fault messages of all defined devices of the MTAX device group.

Assumption: the following three devices are defined:

- Device address 01
- Device address 07 and
- Device address 10.

FI command		**07_BR_ASM1**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
2	1	07
	2	Milling centre 1
	3	74
	4	F
	5	SLM timeout
	6	X
	7	0
3	1	10
	2	Milling centre 2
	3	1
	4	D
	5	The fault has been remedied.
	6	X
	7	0

FI command

Output of the currently applied system fault message of the selected device belonging to the MTAX device group.

BR_ASM2	**(Single Read)**
BC_ASM2	**(Cyclic Read)**
BB_ASM2	**(Break Cyclic Read)**

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM2. The reply consists of a row with 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meanings of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM2

Read the current system fault messages of device address 01.

Assumption: the following three devices are defined:

- Device address 01
- Device address 07 and
- Device address 10

FI command		**01_BR_ASM2**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0

FI command

Output of the current system fault messages of the listed devices belonging to the MTAX device group.

BR_ASM3_(1)	**(Single Read)**
BC_ASM3_(1)	**(Cyclic Read)**
BB_ASM3_(1)	**(Break Cyclic Read)**

(1) = Selection list of up to 10 MTAX devices [00_01_02_ ... _15]

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM3. The number of rows (1... n=15) depends on the number of listed MTAX devices. Each row consists of 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meanings of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM3

Read the current system fault messages of the selected MTAX devices. Assumption: the following device addresses are defined:

- Device address 01,
- Device address 07 and
- Device address 10.

FI command		**01_BR_ASM3_01_10**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low
	6	X
	7	0
2	1	10
	2	Milling centre 2
	3	1
	4	D
	5	The fault has been remedied
	6	X
	7	0

FI command

Output of the current system fault messages of all defined devices (according to the system configuration) of the MTAX device group.

BR_ASM4_(1)	**(Single Read)**
BC_ASM4_(1)	**(Cyclic Read)**
BB_ASM4_(1)	**(Break Cyclic Read)**

(1) = Device group [MTRX, MTCX, MISX, MTAX]

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM4. The number of rows (1... n=15) depends on the number of defined MTAX devices. Each row consists of 7 columns for the device address, the device name, the message number, the message status, the short text and a display as to whether or not a detailed text exists for the fault message.

Row 1..n	Column 1	...	Column 7

Value range/meanings of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
5 =	Short text	[max. 54 ASCII characters]
6 =	Detailed text	[x= available, -- = not available]
7 =	2 bytes additional information on the message number	needed to resolve the information „@“ (see ASM5)

Example ASM4

Read the current system fault messages of all defined devices in the MTAX device group.
Assumption: the following device addresses are defined:

- Device address 01 and
- Device address 10.

FI command		**01_BR_ASM4_MTAX**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
2	1	10
	2	Milling centre 2
	3	1
	4	D
	5	The fault has been remedied.
	6	X
	7	0

FI command

Device and message number-related output of the detailed text of the currently applied fault message.

BR_ASM5_(1)_(2) **(Single Read)**

(1) = Message number [0...150]

(2) = 2 bytes additional information on the message number

Structure of the reply

The following table shows the general structure of the reply of the FI command ASM5. The reply consists of a row with 5 columns for the device address, device name, message number and detailed text.

Row 1..n	Column 1	...	Column 5

Value range/meanings of the columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Message type	[F = Fault, D = Diagnostics]
6 =	Detailed text	[max. 14 Rows with max. 78 characters/row]

Example ASM5

Read the detailed text of the system fault with the message number 74 of the device address 01.

FI command		**03_BR_ASM5_74_0**
Row	**Column**	**Reply**
1	1	01
	2	Drilling centre
	3	74
	4	F
	5	Replace the SLM module on the PLC card (slot 3).

References

For further information about the diagnostic system and the affiliated message types, refer to the Indramat documentation:

„Graphical user interface 17VRS“, Application description,
Chapter 3 Diagnostics,
DOK-MTC200-GBO*GEN*V17-ANW1-DE.

Read CMOS RAM integer parameter: CMI

Designation **CMI** **CM**os Ram **I**nteger Parameter

FI command **CR_CMI_(1)** **(Single Read)**

(1) = Parameter number of the CMOS RAM integer parameter

Structure of the reply The following table shows the general structure of the reply of the FI commands CMI.

Example CMI Read a CMos Ram integer parameter 56.

FI command	**00_CR_CMI_56**
Reply	
Row 1	**Column 1**
	4454

Write CMOS RAM integer parameter: CMI

Designation **CMI** **CM**os Ram **I**nteger Parameter

Explanation The write command CMI makes it possible to write to a CMos Ram integer parameter.

FI command **CW_CMI_(1)** **(Single Write)**

(1) = Parameter number of the CMOS RAM integer parameter

Value to be written Integer variable

Note: The value to be written is communicated in the „acValue“ parameter in the „DataTransfer“ routine.

Structure of the reply One row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge	Value written successfully to the variable

Example CMI Write a value into the CMos Ram integer parameter 56

FI command	00_CW_CMI_56 Value to write: 4454
Reply	
Row	Column 1
1	(P_ACK)

Read CMOS RAM floating point parameter: CMF

Designation **CMF** **CM**os Ram **F**loatingpoint Parameter

FI command **CR_CMF_(1)** **(Single Read)**

(1) = Parameter number of the CMOS RAM floating point

Structure of the reply The following table shows the general structure of the reply of the FI command CMF.

Example CMF Read a CMos Ram FloatingPoint parameter 56

FI command	00_CR_CMF_56
Reply	
Row 1	Column 1
	4454.000

Write CMOS RAM floating point parameter: CMF

Designation **CMF** **CM**os Ram **F**loatingpoint Parameter

Explanation The write command CMF allows you to write a CMos Ram FloatingPoint parameter.

FI command **CW_CMF_(1)** **(Single Write)**

(1) = Parameter number of the CMOS RAM floating point

Value to be written FloatingPoint variable

Note: The value to be written is communicated in the „acValue“ parameter in the „DataTransfer“ routine.

Structure of the reply One row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge	Value written successfully to the variable

Example CMF Write a value to the CMos Ram FloatingPoint parameter 56.

FI command	**00_CW_CMF_56 Value to write: 4454**
Reply	
Row	**Column 1**
1	(P_ACK)

Read CMOS RAM ASCII parameter: CMA

Designation **CMA** **CM**os Ram **A**SCII Parameter

FI command **CR_CMA_(1)** **(Single Read)**

(1) = Parameter number of the CMOS RAM ASCII parameter

Example CMA Read a CMos Ram ASCII parameter 0.

FI command	**00_CR_CMA_0**
Reply	
Row 1	**Column 1**
	Wait for tool change

Write CMOS RAM ASCII parameter: CMA

Designation **CMA** **CM**os Ram **A**SCII Parameter

Explanation The write command CMA allows you to write a CMos Ram ASCII parameter.

FI command **CW_CMA_(1)** **(Single Write)**

(1) = Parameter number of the CMOS RAM ASCII parameter.

Value to be written ASCII variable

Note: The value to be written is communicated in the „acValue“ parameter in the „DataTransfer“ routine.

Structure of the reply One row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge — Value written successfully to the variable

Example CMA Write a value to the CMos Ram ASCII parameter 0.

FI command	**00_CW_CMA_0** **Value to write: Hallo World**
Reply	
Row	**Column 1**
1	(P_ACK)

Control Reset: CRT

Designation **CRT** **C**ontrol **R**ese**T**

Explanation The control reset permits reset of the selected device after a system fault. If there is no system fault, the job is ignored.

FI command **CW_CRT** **(Single Write)**

Value to be written Reset 0 = Trigger reset

Note: The value to be written is passed on in the „DataTransfer“ routine in the „acValue“ parameter.

Structure of the reply A row with one column for acknowledgement of the FI command sent is output. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge Reset executed

Example CRT Trigger a control reset on the selected device.

FI command	**00_CW_CRT** **Value to be written: 0**
Reply	
Row	**Column 1**
1	(P_ACK)

References For further information about reset of the device, refer to the Indramat documentation:

„PLC programming instructions 18VRS“,
Chapter 11.5 Commissioning / Start,
DOK-CONTRL-SPS*PRO*V18-ANW1-DE-E.

Device Configuration parameter: DCP

Designation

DCP **D**evice **C**onfiguration **P**arameter

Explanation

The device configuration parameters that are entered in the active machine parameter set and in the „IND_DEV.INI“ file are output. The configuration parameters of the devices include the device address, the device designation, the device type, the mechanism number, the mechanism name, and the process type.

FI command

Output of the configuration parameters of all defined devices.

BR_DCP1 **(Single Read)**

Note: The FI command DCP1 refers to all defined devices. This is why any valid device address can be specified in the command line (see example DCP1).

Structure of the reply

The following table shows the general structure of the reply of the FI command DCP1. The reply consists of one to n=16 rows (n=16 devices x 1 mechanism = 16) with 7 columns each.

Row 1...n	Column 1	...	Column 7

Note: Columns [1...7] are omitted if there is no active machine parameter set in the device.

Value range/meanings of the columns

1 =	Device address	[00...15]
2 =	Device designation	[max. 32 ASCII characters]
3 =	Device type	[MTCNC, MTC200-P, MTC200-R, MTVNC, MTRA-P, MTRA-R]
4 =	Mechanism number	[0...31]
5 =	Mechanism name	[max. 28 ASCII characters]
6 =	Process kind	[1= internal, 2 = external process]
7 =	Process type	[1 = CNC process, 2 = PLC process]

Example DCP1

Read the configuration parameters of all defined devices.

Assumption: three devices are defined

- Device address 00 (MTCNC)
- Device address 01 (MTC200-P) and
- Device address 02 (MTC200-P).

FI command		**00_BR_DCP1**
Row	**Column**	**Reply**
1	1	00
	2	Revolving transfer machine
	3	MTRA-P
	4	0
	5	Master
	6	1
	7	2
2	1	01
	2	0
	3	MTRA-P
	4	0
	5	Fräse01
	6	1
	7	1
3	1	02
	2	0
	3	MTRA-P
	4	0
	5	Milling cutter 02
	6	1
	7	1

References

For further information about process parameters and their functions and value ranges, refer to the Indramat documentation:

„CNC/PLC Interface description 18VRS Application description, Chapter 3.1 „External mechanisms“
DOK-MTC200-SPS*GWY*V18-ANW1-DE-E.

FI command

Output of the configuration parameters of the selected device.

BR_DCP2 (Single Read)

Structure of the reply

The following table shows the general structure of the reply of the FI command DCP2. The reply consists of one row with 7 columns each.

Row 1	Column 1	...	Column 7

Note: Columns [1...7] of the respective device are omitted if there is no active machine parameter set in the device.

Value range/meanings of the columns

1 =	Device address	[00...15]
2 =	Device designation	[max. 32 ASCII characters]
3 =	Device type	[MTCNC, MTC200-P, MTC200-R, MTVNC, MTRA-P, MTRA-R]
4 =	Mechanism number	[0...31]
5 =	Mechanism name	[max. 28 ASCII characters]
6 =	Process kind	[1= internal, 2 = external process]
7 =	Process type	[1 = CNC process, 2 = PLC process]

Example DCP2

Read the device configuration parameters of the selected device (device address 01).

Assumption: three devices are defined

- Device address 00 (MTCNC)
- Device address 01 (MTC200-P)
- Device address 02 (MTC200-P)

FI command		01_BR_DCP2
Row	**Column**	**Reply**
1	1	01
	2	0
	3	MTRA-P
	4	0
	5	Milling cutter 01
	6	1
	7	1

References

For further information about process parameters and their functions and value ranges, refer to the Indramat documentation:

„CNC/PLC Interface description 18VRS Application description, Chapter 3.1 „ External mechanisms“
DOK-MTC200-SPS*GWY*V18-ANW1-DE-E.

Identification string of NC/PLC data: DIS

Designation **DIS** **D**ata **I**dentification **S**tring

FI command Output of the PLC program identification.

CR_DIS2 **(Single Read)**

Distance to go: DTG

Designation **DTG** **D**istance **T**o **G**o

FI command

CR_DTG2_(1)_(2)	**(Single Read)**
CC_DTG2_(1)_(2)	**(Cyclic Read)**
CB_DTG2_(1)_(2)	**(Break Cyclic Read)**

(1) = Physical axis number	[0...6]
(2) = Coordinate system	[1 = Machine coordinates 2 = Program coordinates]

Read device group: DTY

Designation **DTY** **D**evice **TY**pe

FI command **CR_DTY1** **(Single Read)**

Structure of the reply One row is output with one column for the device type.

Example DTY1 Read the device type of the selected device address

FI command	**00_CR_DTY1**		
Reply			
Row	**Column 1**	**Column 2**	**Column 3**
1	MTC200-P	MTS-P	MTC-P

Module assignment of a process: MAP

Designation **MAP** **M**odul **A**ssign of **P**rocess

Explanation The module to which a specific process is assigned is read out of the „Moduldef.ini'„ file. This file is located in the [Drive]:\MT-CNC\CONFIG directory and contains all module configuration data.
The process data is located in the sections:
[DeviceAddrX\ModulY\Process]
and „X“ stands for the addressed device and „Y“ for the configured module numbers.

FI command Determine the module to which the process belongs. Information is read from the module configuration of the MTAX device group.

BR_MAP1_(1)	**(Single Read)**
BC_MAP1_(1)	**(Cyclic Read)**
BB_MAP1_(1)	**(Break Cyclic Read)**

(1) = Mechanism number [0...31]

Structure of the reply The following table shows the general structure of the reply of the FI command MAP1. One row with one column for the determined module number is output.

Value range of the columns 1 = Module number [0...99]

Example MAP1 Read the module number assigned to the CNC process number 4 out of the module configuration.

Assumption: the module to which CNC process 4 is assigned has the module number 5.

FI command	**03_BR_MAP1_4**
Reply	
Row	**Column 1**
1	5

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the Indramat documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

Device data of the module configuration: MCD

Designation

MCD **M**odul **C**onfiguration: **D**evice Information

Explanation

All device data of the module configuration is read out of the „Moduldef.ini“ file, which is located in the „[Drive]:\MT-CNC\CONFIG“ directory. The device data is located in the [DeviceAddrX] sections, and „X“ stands for the configured device addresses.

FI command

Read the device data within the module configuration of the MTAX device group.

BR_MCD1	**(Single Read)**
BC_MCD1	**(Cyclic Read)**
BB_MCD1	**(Break Cyclic Read)**

Note: The FI command MCD1 refers to all devices within the MTAX device group. This is why any valid device address can be specified in the command line (see example MCD1).

Structure of the reply

The following table shows the general structure of the reply of the FI command MCD1. The number of rows depends on the number of configured devices. Each row consists of four columns for the device address and the PLC FB names for the provision of setup diagnostics, warning information and start prerequisites.

Row 1	Column 1	...	Column 4

Value range of the columns

1 = Device address	[0...15]
2 = PLC FB name for setup diagnostics	[max. 9 ASCII characters]
3 = PLC FB name for warning information	[max. 9 ASCII characters]
4 = PLC FB name for start prerequisites	[max. 9 ASCII characters]

Example MCD1 Read all device data of the module configuration

Assumption: the following devices of the MTCX device group are defined:

- Device address 01 (MTC200-P)
- Device address 03 (MT-CNC)

FI command	03_BR_MCD1			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	01	PVSetup_1	PVWarn_1	PVStart_1
2	03	PVSetup_3	PVWarn_3	PVStart_3

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the Indramat documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P.

Module data of the module configuration: MCM

Designation **MCM** **M**odul **C**onfiguration: **M**odul Information

Explanation All module data of a specific device is read out of the „Moduldef.ini“ file, which is located in the „[Drive]:\MT-CNC\CONFIG“ directory and which contains all module configuration data. The module data is located in the [DeviceAddrX\ModulY] sections; „X“ stands for the addressed device and „Y“ for the configured module numbers.

FI command Read module data out of the module configuration with regard to a device of the MTAX device group.

BR_MCM1 **(Single Read)**

BC_MCM1 **(Cyclic Read)**

BB_MCM1 **(Break Cyclic Read)**

Structure of the reply The following table shows the general structure of the reply of the FI command MCM1. The number of rows depends on the number of configured modules of a device. Each row consists of four columns for the module number, the module name and PLC FB names for general module faults and module messages.

Row 1	Column 1	...	Column 4

Value range of the columns

1 = Module number	[0...99]
2 = Module name	[max. 28 ASCII characters]
3 = PLC FB name for gen. module faults	[max. 9 ASCII characters]
4 = PLC FB name for messages from module	[max. 9 ASCII characters]

Example MCM1

Read the module data of device 03 from the module configuration:

Assumption: the following modules are defined:

- Module number 5
- Module number 7

FI command	**03_BR_MCM1**			
Reply				
Row	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	5	Module 5 – Milling	PVError_5	PVMsg_5
2	7	Module 7 - Drilling	PVError_7	PVMsg_7

References

For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the Indramat documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P

Process data of the module configuration: MCP

Designation

MCP **M**odul **C**onfiguration: **P**rocess Information

Explanation

All process data of a specific module is read out of the „Moduldef.ini'„ file. This file is located in the „[Drive]:\MT-CNC\CONFIG“ directory and contains all module configuration data. The process data is located in the [DeviceAddrX\ModulY\Process] sections; „X“ stands for the addressed device and „Y“ for the selected module number.

BR_MCP1_(1) **(Single Read)**

BC_MCP1_(1) **(Cyclic Read)**

BB_MCP1_(1) **(Break Cyclic Read)**

(1) = Module number [0...99]

Structure of the reply

The reply of the FI command MCP1 consists of one to n=32 rows with one column for the number of the CNC process or of external mechanisms.

Value range of the columns

(1) = Mechanism number [0...31]

Example MCP1 Read the CNC process number of module 5 of device 03 of the module configuration.

Assumption: the following CNC processes are defined:

- CNC process number 1
- CNC process number 4

FI command	03_BR_MCP1_5
Reply	
Row	Column 1
1	1
2	4

References For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the Indramat documentation:

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P

SFC data of the module configuration: MCS

Designation **MCS** **M**odul **C**onfiguration: **S**FC Information

Explanation All SFC data of a specific module is read out of the „Moduldef.ini” file. This file is located in the „[Drive]:\MT-CNC\CONFIG“ directory and contains all module configuration data. The SFC data is contained in the [DeviceAddrX\ModulY\Sfc] sections; „X“ stands for the addressed device and „Y“ for the selected module number.

FI command Read the SFC data with regard to the module of a device out of the module configuration of the MTAX device group.

BR_MCS_(1) **(Single Read)**

BC_MCS_(1) **(Cyclic Read)**

BB_MCS_(1) **(Break Cyclic Read)**

(1) = Module number [0...99]

Structure of the reply The number of rows depends on the number of configured Indrastep sequencers of a module. Each row has a column for the names of the Indrastep sequencers.

Value range of the columns

1 = Name of the Indrastep sequencer [Format W.X.Y.Z].

Format W.X.Y.Z	Value range
W	max. 9 ASCII characters
X	max. 9 ASCII characters ! OPTIONAL !
Y	max. 9 ASCII characters ! OPTIONAL !
Z	max. 9 ASCII characters ! OPTIONAL !

Example MCS1

Read the names of the Indrastep sequencers of module 5 of device 03 of the module configuration.

Assumption: the following Indrastep sequencers are defined:

- ISFB_1
- FB_US.ISFB_3
- FB_US.ISFB_3.SW1.ABBA
- FB_US.ISFB_3.SW1.ABBA

FI command	03_BR_MCS1_5
Reply	
Row	Column 1
1	ISFB_1
2	FB_US.ISFB_3
3	FB_US.ISFB_3.SW1
4	FB_US.ISFB_3.SW1.ABBA

References

For further information about module configuration and the structure of the „Moduldef.ini“ file, refer to the INDRAMAT documentation::

„Diagnostic and message system for HMI systems ProVi“,
Chapter 3.2 Configuring Moduldef.ini,
DOK-MTC200-DIAG*PROVI*-ANW1-DE-P

Read and write PLC variables: PVS

Designation **PVS** **P**LC-**V**ariable **S**ingle

Explanation The following types of PLC variables can be read or written:

- BOOL, BYTE, SINT, USINT, CHAR, WORD, INT, UINT, STRING, DWORD, DINT, UDINT, TIME, REAL and imported structures and arrays.

FI command Read PLC variables.

CR_PVS_(1)	**(Single Read)**
CC_PVS_(1)	**(Cyclic Read)**
CB_PVS_(1)	**(Break Cyclic Read)**

(1) = Designator of the PLC variable

Note: Addressing a variable that has not been declared leads to an error message. The data length must not exceed 240 bytes (see also chapter entitled „Programming, Guidelines“).

Example PVS Read the value of the PLC variable with the designator „IB_EXT24“ at device address 00 in the CNC process 0.

Assumption:

- The PLC variable with the designator „IB_EXT24“ is of the „BOOL“ type.

FI command	**00_CR_PVS_IB_EXT24**
Reply	
Row	**Column 1**
1	1

FI command Write a PLC variable.

CW_PVS_(1) **(Single Write)**

(1) = Designator of the PLC variable

Value to be written PLC variable [Format according to the type in the PLC program]

Note: Only defined PLC variables can be written. Addressing a variable that has not been declared leads to an error message. The data length must not exceed 240 bytes (see also chapter entitled „Programming, Guidelines“). The value to be written is communicated as an ASCII value in the „acValue“ parameter in the „DataTransfer“ routine.

Structure of the reply One row is output with one column for acknowledgement of the FI command sent. Meanings:

(P_ACK) = **P**ositive **ACK**nowledge Variable has been written.

Example PVS Write the value 1 into the PLC variable with the designator „IB_EXT24“ at device address 00.

Assumption:

- The PLC variable with the designator „IB_EXT24“ is of the „BOOL“ type.

FI command	**00_CW_PVS_IB_EXT24** **Value to be written: 1**
Reply	
Row	**Column 1**
1	(P_ACK)

7 Error codes

7.1 General error result line

If the „DataTransfer" routine returns an error code, it is not the requested data that is returned with the „ReadGroupItem" routine, but a general error result line. This error result line contains further information about the possible cause of the error.

Note: Refer to the chapter entitled „Programming" for details of how the routines function and ofr tips and tricks on how to work with the function interface.

The following table shows the general structure of the error result line. One row consisting of 5 columns for the error lcass, the error code, additional information, the error text and detailed text is output.

Row 1	Column 1	...	Column 5

Meanings of the columns

1 = Error class

2 = NACK code or return error code (depending on error class)

3 = Additional information [hexadecimal LONG value]

4 = Error text [ASCII characters]

5 = Detailed text [ASCII characters]

The „INDIF000.h" or the „INDIF000.BAS" file contans the following error classes:

Error class	Meaning
FI_ERROR_CLASS_NACK	NACK messages of the controller
FI_ERROR_CLASS_FCT	Error codes of the log functions

7.2 Error codes from 200 to 999

Code	Designation	Meaning
201	BOF_INVALID_MTCNC_NUMBER	Invalid device address
202	BOF_NO_INST_PATH	No installation path found (Indramat.INI).
203	BOF_NO_MAP_FILE_FOUND	MAP file „PLCMAP.SPS“ not founc.
204	BOF_NO_MAP_FILE_NAME	No entry found for required device address in the „PLCMAP.DAT“ file.
205	BOF_FILE_ERROR	File editing error.
206	BOF_VERSION_ERROR	More than MAXLINES in the „VERSION.DAT“ file. Remedy: delete „VERSION.DAT“ file
207	BOF_MUTEX_ERROR	Error generating a MUTEX object
208	BOF_FILE_MAPPING_ERROR	Error generating a file mapping.
209	BOF_MEMORY_ERROR	Error allocating a memory
210	BOF_DATA_MAP_ERROR	Error access the DATA-MAP.
211	BOF_MUTEX_TIMEOUT	Acces error to the MAP file
212	BOF_DATA_LENGTH_ERROR	Data buffer is too small.
213	BOF_FILE_NOT_FOUND	File not found.
214	BOF_SYS_MAP_ERROR	Error accessing the SYSTEM-MAP.
215	BOF_MAP_ELEMENT_ERROR	No valid MAP structure element
216	BOF_INVALID_CHANNEL_ERROR	Invalid number of LOG channel
217	BOF_TIMEOUT_ERROR	Specified timeout has elapsed
218	BOF_SHMEM_ALREADY_EXIST	SHARED-MEM already exists.
219	BOF_PROCESS_NOT_EXIST	Adressed process does not exist
220	BOF_FILE_EOF	End of file reached
221	BOF_EVENT_ERROR	Error generating an event object.
222	BOF_PROCESS_ALREADY_EXIST	Process to start already running
223	BOF_COMM_ADDRESS_ERROR	No valid communicaiton address
224	BOF_DEVICE_TYP_ERROR	No valid device type.
225	BOF_DEVICE_ERROR	No valid device address defined
226	BOF_DEVICE_NAME_ERROR	Invalid device designation
227	BOF_DEVICE_STATUS_ERROR	No valid device status.
228	BOF_DEVICE_PLC_ERROR	NO VALID PLC Information.
229	BOF_TASK_ID_ERROR	invalid or wrong task ID
230	BOF_TASK_ADM_ERROR	Task management error.
231	BOF_TASK_TRIGGER_ERROR	Task trigger event error
232	BOF_EVENT_NOT_FOUND	EVent object does not yet exist
233	BOF_TASK_NAME_ERROR	Task name is too long.
234	BOF_SYS_STACK_INDEX_ERROR	Invalid SYS-MSG-STACK-INDEX
235	BOF_SYS_STACK_FULL_ERROR	SYS-MSG-STACK is full
236	BOF_SYS_STACK_MSG_ERROR	SYS-MSG message not known in the SYS-MSG-STACK

Code	Designation	Meaning
237	BOF_SYS_STACK_ACCEPT_ERROR	SYS-MSG message could not be transferred to the SYS-MSG-STACK in the specified time
238	BOF_SYS_MSG_SET_ERROR	The SYS_MSG channel could not be accessed in the specified time (sending a SYS message).
239	BOF_SYS_MSG_GET_ERROR	The SYS_MSG channel could not be accessed in the specified time (fetching a SYS message
240	BOF_DATA_TIME_ERROR	A data element in the shared memory area was not released in the specified time.
241	BOF_DATA_ACCESS_ERROR	A data element in the shared memory area is access-locked.
242	BOF_FCT_PAR_ERROR	An invalid parameter value was passed on in the function.
243	BOF_SYS_STACK_QUIT_ERROR	SYS-MSG acknowledgement was no released in the specified time.
244	BOF_NO_SYS_MSG_RDY	No SYS-MSG message available.
245	BOF_FORMAT_ERROR	Format error
246	BOF_SYS_MSG_LIST_ERROR	Error in the SYS-MSG list for manual acknowledgement.
247	BOF_NO_IFDLL_MODE_ERROR	Mode missing in the „IfDllMode=„ entry of the „INDRAMAT.INI“ file.
248	BOF_LOG_GRP_COUNT_ERROR	Invalid group number for the LOG channel.
249	BOF_NO_SYS_MSG_CONF_ERROR	No SYS-MSG acknowledgement message.
250	BOF_NO_PATH_ENV_ERROR	No path environment variable available
251	BOF_LOGIN_IF_EVENT_ERROR	LOGINIF event oculd not be created
252	BOF_LOGIN_SYS_MSG_ERROR	LOGINIF could not be realised inthe specified time owing to exsiting SYS-MSGs.
253	BOF_LOGIN_EVENT_TIME_ERROR	Could not send a SYS-MSG message because the login event was occupied too long.
254	BOF_DLL_MUTEX_TIMEOUT_ERROR	Access to DLL locking mutex was not possible in the specified time
255	BOF_DLL_ALREADY_INSTALLED	DLL is already in the process
256	BOF_DLL_LOAD_ERROR	DLL could not be loaded in with LoadLibrary
257	BOF_DLL_LIST_FULL_ERROR	DLL management structure is full.
258	BOF_DLL_LIST_DELETE_ERROR	DLL to delete is not in the management structure.
259	BOF_DOS_NT_SYS_MSG_ERROR	Invalid SYS-MSG message number in the case of DOS → NT job processing
260	BOF_DOS_NT_JOB_STR_ERROR	Invalid DOS → NT command string.
261	BOF_SYS_MSG_RANGE_ERROR	SYS-MSG message number outside valid number range.
262	BOF_DOS_NT_JOB_INFO_ERROR	DOS → NT command info is too long
263	BOF_DOS_NT_SYS_MSG_Q_ERROR	In the DOS → NT command SYSC_xxx an odd SYS-MSG message number (acknowledgement) has been passed on
264	BOF_DOS_NT_FKT_NOT_FOUND_ERROR	DOS → NT sent for which there is no processing function in the „BOFINTFC.DAT“ file.
265	BOF_DOS_NT_DLL_NAME_NOT_FOUND_ERROR	No DLL name for DOS → NT commands in the „BOFINTFC.DAT“ file.

Code	Designation	Meaning
266	BOF_DOS_NT_DLL_NOT_FOUND_ERROR	DLL not found for the DOS → NT commands.
267	BOF_DOS_NT_FKT_NOT_IN_DLL_ERROR	DOS → NT processing function not found in the specified DLL.
268	BOF_DOS_NT_BOF_DAT_NOT_FOUND_ERROR	The „BOFINTFC.DAT“ file could not be found.
269	BOF_TASK_NAME_NOT_FOUND_ERROR	Task name is not found in task management.
270	BOF_TASK_ID_NOT_FOUND_ERROR	No task available for the Task-Ident.
271	BOF_NT_CODE_ERROR	WIN-32 API error has occurred.
272	BOF_DOS_NT_PROCESS_PRIORITY_ERROR	Ivalid Process-Priority-Class.
273	BOF_DOS_TASK_NAME_ERROR	Error generating the DOS-BOF task name.
274	BOF_PARENT_WIN_NAME_LEN_ERROR	Name of parent window is too long
275	BOF_TERMINATE_EVENT_NAME_LEN_ERROR	Name of termination event is too long
276	BOF_PARENT_WIN_NOT_EXIST_ERROR	Loged-on tasks has no parent window
277	BOF_DLL_NOT_EXIST_ERROR	Sought DLL does not exist
278	BOF_DLL_FUNCTION_NOT_FOUND_ERROR	Sought function does not exist in the specified DLL.
279	BOF_PROCESS_NOT_LOGIN_ERROR	An FI command has been called up although the application has not logged on. The „LogInIf“ routine has not yet been executed
280	BOF_DEVICE_HANDLE_ERROR	Could not generate a device handle
281	BOF_DEVICE_ASSIGN_ERROR	No or invalid „DeviceAssign“ entry in the „IND_DEV.INI“ file.
282	BOF_MEMORY_CLASS_ERROR	No valid memory type in the case of the DOS → NT RDNT/WRNT command.
283	BOF_MEMORY_DOS_NT_DATA_LEN_ERROR	Data length in the case of the RDNT/WRNT command too long.
284	BOF_SHMEM_INDEX_ERROR	No valid SHARED-MEMORY-INDEX.
285	BOF_NO_PORT_ADDR_ERROR	No port address in the communicaiton address.
286	BOF_NO_PORT_VALUE_ERROR	No port value in the communicaiton address.
287	BOF_VRT_MANAGER_MODE_ERROR	Invalid VRT-Manager-Mode
288	BOF_VRT_START_MODE_ERROR	No or invalid MTVNC mode entry in the „IND_DEV.INI“ file.
289	BOF_PAR_MIN_ERROR	No PARMIN value in the „BOFINTFC.DAT“ file for the FI command.
290	BOF_PAR_MAX_NUMBER_ERROR	Too many transfer parameters in the FI command.
291	BOF_PAR_MAX_ERROR	No PARMAX value in the „BOFINTFC.DAT“ file for FI command.
292	BOF_PAR_NUMBER_ERROR	Number of FI parameters does not agree with the data in the DAT files (e.g.: MTCX.DAT, BOFINTFC.DAT, etc.).
293	BOF_PAR_DESCRIPT_ERROR	No parameter description available.
294	BOF_PAR_TYPE_ERROR	Inalid FI parameter type.
295	BOF_PAR_TYPE_NOT_FOUND_ERROR	No FI parameter type description available
296	BOF_PAR_DATA_ERROR	Invalid FI parameter data item, i.e. FI data item does not exist in the defined FI data type
297	BOF_PAR_TYPE_DESCRIPT_ERROR	Invalid FI parameter type description
298	BOF_PAR_INDEX_ERROR	FI parameter index is too large
299	BOF_PAR_NO_CYCLIC_ERROR	Either no or invalid CYCLIC entry in the „BOFINTFC.DAT“ file.

Code	Designation	Meaning
300	BOF_PAR_NO_CYCLIC_FI_COMMAND_ERROR	Cyclic requesting of the FI command not enabled.
301	BOF_PAR_NO_BINAER_ERROR	Either no binary entry or invalid binary entry in the „BOFINTFC.DAT“ file.
302	BOF_PAR_NO_BINAER_FI_COMMAND_ERROR	Binary operation of the FI command not enabled.
303	BOF_NT_DOS_CHANNEL_ACCESS_ERROR	Could not access the NT → DOS job channel in the specified time
304	BOF_NT_DOS_COMMAND_LENGTH_ERROR	NT → DOS command string too long.
305	BOF_NT_DOS_COMMAND_INFO_LENGTH_ERROR	NT → DOS command info string too long.
306	BOF_NT_DOS_TIMEOUT_ERROR	Could not execute NT → DOS job in the specified time.
307	BOF_NT_DOS_FKT_NOT_FOUND_ERROR	An NT → DOS command was sent that is not declared in the „BOFINTFC.DAT“ file.
308	BOF_NT_DOS_DLL_NAME_NOT_FOUND_ERROR	For the NT→DOS command sent, no DLL is declared in the „BOFINTFC.DAT“ file.
309	BOF_NT_DOS_DLL_NOT_FOUND_ERROR	DLL not found for the NT → DOS commands
310	BOF_NT_DOS_FKT_NOT_IN_DLL_ERROR	NT → DOS processing function not found in the specified DLL.
311	BOF_NT_DOS_JOB_STR_ERROR	Invalid NT → DOS command string
312	BOF_NT_DOS_JOB_INFO_ERROR	NT → DOS command info string is too long
313	BOF_NT_DOS_DPR_TIMEOUT_ERROR	Could not access the NT-DOS-DPR memory in the specified time.
314	BOF_NT_DOS_NO_COMMAND_ERROR	No NT → DOS command string available
315	BOF_NT_DOS_BOF_INDEX_ERROR	Invalid DOS-BOF-INDEX when sending an NT → DOS command.
316	BOF_PAR_INVALID_VALUE_ERROR	Transfer parameter of the function has an invalid value.
317	BOF_DOS_BOF_EXE_PATH_ERROR	DOS-BOF EXE must not have a path.
318	BOF_LOG_IN_LOG_OUT_TIMEOUT_ERROR	Could not log in/out in the specified time
319	BOF_DEVICE_TYP_GROUP_ERROR	Selected device address does not exist in the device group
320	BOF_INVALID_PROCESS_NUMBER_ERROR	Invalid CNC process numbere
321	BOF_PROCESS_NAME_LENGTH_ERROR	Proces name is too long or invalid
322	BOF_PARAM_IDENT_REQUEST_ERROR	Invalid data returned by interface when requesting parameters.
323	BOF_SWITCH_DEVICE_ERROR	Attempt made to access a virtual MTC that is assigned to a real MTC
324	BOF_DEVICE_TYPE_REQUEST_ERROR	Invalid data returned when requesting the device type ID.
325	BOF_DEVICE_SPS_IDENT_ERROR	Interface returned invalid data when requesting the long ID of the PLC SPS MAP file.
326	BOF_INVALID_AXIS_NUMBER_ERROR	Invalid axis number received [1...32].
327	BOF_NO_GBOVERSION_ERROR	No or invalid „GBOVERSION=„ entry in the „INDRAMAT.INI“ file.
328	BOF_NO_ACHSREF_TABLE_ERROR	Axis reference table error
329	BOF_DEVICE_GROUP_ERROR	Invalid device group for the job.

Code	Designation	Meaning
330	BOF_PROCESS_NOT_DEFINED	Process is not defined in the current parameters
331	BOF_INVALID_DEVICE_GROUP_VALUE_ERROR	Invalid device group number
332	BOF_INVALID_DEVICE_ID_STR_ERROR	Invalid device ident. string
333	BOF_INVALID_DEVICE_GROUP_STR_ERROR	Invalid device group string
334	BOF_FI_JOB_CLASS_ALREADY_RUN_ERROR	FI-JOB already running.
335	BOF_FI_JOB_REQUEST_ERROR	No further FI-JOBS possible.
336	BOF_FI_JOB_ID_ERROR	No t a valid FI-JOB-Ident.
337	BOF_FI_JOB_NO_ID_FOUND_ERROR	No FI-JOB-Ident found in the management structure.
338	BOF_FI_JOB_PROGRESS_TYPE_ERROR	Invalid FI-JOB progress query.
339	BOF_FI_JOB_EXECUTE_FKT_NOT_FOUND_ ERROR	Processing function for the FI-JOB not found in the specified DLL.
340	BOF_FI_JOB_ERROR_STRING_TO_LONG	FI-JOB-ERROR-STRING is too long.
341	BOF_FI_JOB_TIMEOUT_ERROR	FI-JOB could not be processed in the specified time.
342	BOF_FI_ERROR_STRING_TO_LONG	String for the general FI ERROR RESPONSE TELEGRAM (general error result line) is too long.
343	BOF_DOS_MANAGERPROG_NOT_READY_ERROR	DOS-NT manager program not running.
344	BOF_NT_DOS_ORDER_TO_LONG	Compsed NT→DOS job descriptoin is too long
345	BOF_FILE_CLASS_OBJECT_INSTALL_ ERROR	File class object for accesses to the BOF files could not be created
346	BOF_FILE_DIAGOFF_NOT_FOUND_ERROR	BOF file „DIAGOFF.XXX“ does not exist
347	BOF_FILE_DIAGOFF_OPEN_ERROR	Error opening the „DIAGOFF.XXX“ file.
348	BOF_SH_MEM_DIAGOFF_NOT_FOUND_ERROR	No SHARED-MEMORY available for DIAGOFFxxx.
349	BOF_FILE_DIAGTAB_NOT_FOUND_ERROR	BOF file „DIAGTAB.XXX“ does not exist
350	BOF_FILE_READ_WITH_FS_CLASS_ERROR	Read error with FS CLASSES
351	BOF_FILE_DIAGTEXT_NOT_FOUND_ERROR	Diagnostic text file „STERRxx.YYY“ does not exist.
352	BOF_FILE_STERR_FILE_CLOSE_ERROR	Diagnostic text file „STERRxx.YYY“ could not be closed.
353	BOF_FILE_STERR_FILE_OPEN_ERROR	Diagnostic text file „STERRxx.YYY“ could not be opened.
354	BOF_FILE_STERR_FILE_POSITION_ERROR	File positioning in the diagnostic text file „STERRxx.YYY“ not possible
355	BOF_FILE_STERR_FILE_READ_ERROR	Read function to the diagnostic text file „STERRxx.YYY“ not possible
356	BOF_FILE_STERR_FILE_NOT_FOUND_ERROR	Diagnostic text file „STERRxx.YYY“ does not exist.
357	BOF_FILE_DIAGTAB_POSITION_ERROR	File positioning „DIAGTAB.xxx“ not possible
358	BOF_FILE_STERR_FILE_TIMEOUT_ERROR	TIMEOUT waiting for MUTEX release for access to the STERR files.
359	BOF_TASK_THREAD_TRIGGER_INFO_TO_LONG	Passed on additional info for TASK-THREAD-TRIGGERING is too long.

Code	Designation	Meaning
360	BOF_TASK_THREAD_TRIGGER_TIMEOUT_ERROR	MUTEX release TIMEOUT for access to TASK-THREAD-TRIGGERING.
361	BOF_FILE_SPRACHE_FILE_OPEN_ERROR	Could not open the „SPRACHE.DAT“ file
362	BOF_COMMAND_RESULT_DATA_TYPE_ERROR	For an FI command (BR_...), a result data type has been requested which is not permisible (e.g. 00_BR_AMM1/2).
363	BOF_FILE_TEXT_FILE_NOT_FOUND_ERROR	Corresponding TEXTxx.YY file does not exist
364	BOF_FILE_TIND_FILE_NOT_FOUND_ERROR	Corresponding TINDxx.YY file does not exist
365	BOF_FILE_TIND_FILE_OPEN_ERROR	Could not open TINDxx.YY
366	BOF_TEXT_NUMBER_TO_LARGE_ERROR	Text number to read from BOF text file is too large.
367	BOF_FILE_TEXT_FILE_OPEN_ERROR	Could not open TEXTxx.YY.
368	BOF_FILE_TEXT_FILE_POSITION_ERROR	File positioning not possible in the text file „TEXTxx.YY“.
369	BOF_FILE_TEXT_FILE_READ_ERROR	Read function to the test file „TEXTxx.YY“ not possible.
370	BOF_DIAGNOSTIC_NUMBER_TO_LARGE_ERROR	Message number for CNC/PLC message system is too large.
371	BOF_FILE_SYSERI_NOT_FOUND_ERROR	BOF file „SYSERI.XXX“ does not exist.
372	BOF_FILE_SYSERI_OPEN_ERROR	Error opening te „SYSERI.XXX“ file.
373	BOF_FILE_SYSERI_POSITION_ERROR	File positioning not possible in SYSERI.xxx
374	BOF_SH_MEM_SYSERI_NOT_FOUND_ERROR	NO SHARED-MEMORY available for SYSERIxxx
375	BOF_FILE_SYSANW_NOT_FOUND_ERROR	Diagnostic text file SYSANW.YY does not exist
376	BOF_FILE_SYSANW_FILE_CLOSE_ERROR	Diagnostic text file SYSANW.YY could not be closed.
377	BOF_FILE_SYSANW_FILE_OPEN_ERROR	Diagnostic text file SYSANW.YY could not be opened.
378	BOF_FILE_SYSANW_POSITION_ERROR	File positioning in SYSANW.YY not possible.
379	BOF_FILE_SYSANW_READ_ERROR	Read function to the diagnostic text file „SY-SANW.YY“ not posible.
380	BOF_FILE_SYSANW_FILE_TIMEOUT_ERROR	TIMEOUT waiting for MUTEX release for access ot the SYSANW.YY files.
381	BOF_FILE_TXERR_FILE_NOT_FOUND_ERROR	Corresponding TXERR.YY-FILE does not exist.
382	BOF_FILE_TXERI_FILE_NOT_FOUND_ERROR	Corresponding TXERI.YY file does not exist.
384	BOF_FILE_TXERR_FILE_OPEN_ERROR	Could not open TXERR.YY
385	BOF_FILE_TXERR_FILE_POSITION_ERROR	File positioning in the text file „TXERR.YY“ not possible.
386	BOF_FILE_TXERR_FILE_READ_ERROR	Read function to the text file „TXERR.YY“ not posible.
387	BOF_COMMAND_NOT_AVAILABLE_DLL_MODE	The requested FI command is not available for the „IfDllMode=„ set in the „INDRAMAT.INI“ file.
383	BOF_FILE_TXERI_FILE_OPEN_ERROR	Could not open TXERI.YY
388	BOF_NO_PARAMETER_SET_IN_CONTROL	No valid parameter set in the controller
389	BOF_ANDRON_COMMANDLINE_ERROR	No valid command line for the ANDRON DRIVER.

Code	Designation	Meaning
390	BOF_FAR_DEVICE_STATUS_ERROR	No or invalid FARDEVICE entry.
391	BOF_DEVICE_PATH_ERROR	No or invalid device path entry.
392	BOF_DEVICE_PROTOCOL_ERROR	No or invalid device log entry.
393	BOF_DEVICE_IP_ERROR	No or invalid DEVICEIP entry.
394	BOF_DOS_NT_TASK_CHANNEL_ TIMEOUT_ERROR	Could not access the DOS→NT job channel in the specified time.
395	BOF_PROCESS_NAME_ERROR	Syntax error detected in process name.
396	BOF_NETINTFC_MANAGER_MODE_ERROR	Invalid NETINTFC-MANAGER-MODE.
397	BOF_NET_MANAGER_STATUS_ERROR	Invalid NET-MANAGER-STATUS entered in the „IND_DEV.INI“ file.
398	BOF_TERMINATE_EVENT_NOT_FOUND_ERROR	No termination event available for the logged on task.
399	BOF_PARENT_WIN_ALREADY_EXIST_ERROR	PARENT-WINDOW-NAME already exists in TASK MANAGEMENT
400	BOF_NO_IFVERSION_ERROR	No „IfVersion=„ entry in the „Indramat.INI“ file.
401	BOF_NO_IFVERSION_ERROR	No IFVERSION entry in INDRAMAT.INI
402	BOF_NO_ANDRON_INST_PATH	No ANDRON installation path found
403	BOF_SYSANW_FILTER_FILE_CREATE_ERROR	Filter file SYSSTW.XX for SYSANW.XX (SHORT MESSAGES only!) could not be created
404	BOF_FILTER_FILE_DIRECTORY_CREATE_ERROR	The temporary subdirectory TEMPDATA for the data files of the miniature control units could not be created.
405	BOF_DELETE_FILE_ERROR	Data file (miniature control units) cannot be deleted
406	BOF_TXERR_FILTER_FILE_CREATE_ERROR	Filter file TXEST.XX for TXERR.XX (SHORT MESSAGES only!) could not be created
407	BOF_STERR_FILTER_FILE_CREATE_ERROR	Filter file STESTyy.XX for STERRyy.XX (SHORT MESSAGES only!) could not be created
408	BOF_TXERR_FILTER_FILE_ NOT_FOUND_ERROR	Filter file TXEST.XX für TXERR.XX (SHORT MESSAGES only!) does not exist in temporary TEMPDATA subdirectory
409	BOF_TXERR_FILTER_FILE_OPEN_ERROR	Filter file TXEST.XX for TXERR.XX (SHORT MESSAGES only!) could not be opened in the temporary TEMPDATA subdirectory
410	BOF_TXEST_INDEX_FILE_CREATE_ERROR	INDEX file for TXEST.XX (SHORT MESSAGES only!) could not be created
411	BOF_BUFFER_LENGTH_ERROR	The PROCESSING BUFFER is to small for the data to process
412	BOF_MSG_NUMBER_0_NOT_EXIST_ERROR	There is NO messag enumber 0 in the message file
413	BOF_MSG_NUMBER_TO_BIG_ERROR	Mesage number in the mssage file too large
414	BOF_WRITE_FILE_ERROR	File could not be written
415	BOF_SYSANW_FILTER_FILE_ NOT_FOUND_ERROR	Filter file SYSSTW.XX for SYSANW.XX (SHORT MESSAGES only!) does not exist in temporary TEMPDATA subdirectory
416	BOF_SYSSTW_INDEX_FILE_CREATE_ERROR	Index file for SYSSTW.XX (SHORT MESSAGES only!) could not be created
417	BOF_SYSANW_FILTER_FILE_OPEN_ERROR	Filter file SYSSTW.XX for SYSANW.XX (SHORT MESSAGES only!) could not be opened in the temporary TEMPDATA subdirectory

Code	Designation	Meaning
418	BOF_STERR_FILTER_FILE_ NOT_FOUND_ERROR	Filter file STESTyy.XX for STERRyy.XX (SHORT MESSAGES only!) does not exist in temporary TEMPDATA subdirectory.
419	BOF_STESTYY_INDEX_FILE_CREATE_ERROR	The index file for STESTyy.XX (SHORT MESSAGES only!) could not be created
420	BOF_STERR_FILTER_FILE_OPEN_ERROR	Filter file STESTYY.XX for STERRYY.XX (SHORT MESSAGES only!) could not be opened in the temporary TEMPDATA subdirectory.
421	BOF_WRONG_TELEGRAMM_CODE_ERROR	An INVALID TELEGRAM CODE has been returned by the controller.
422	BOF_TXEST_INDEX_FILE_NOT_FOUND_ERROR	Index file TXESI.XX could not be found.
423	BOF_TXEST_INDEX_FILE_OPEN_ERROR	Index file TXESI.XX could not be opened.
424	BOF_TXEST_INDEX_FILE_READ_ERROR	Index file TXESI.XX could not be read.
425	BOF_SYSSTW_INDEX_FILE_ NOT_FOUND_ERROR	Index file SYSSIW.XX could not be found.
426	BOF_SYSSTW_INDEX_FILE_OPEN_ERROR	Index file SYSSIW.XX could not be opened.
427	BOF_SYSSTW_INDEX_FILE_READ_ERROR	Index file SYSSIW.XX could not be read.
428	BOF_STESTXX_INDEX_FILE_ NOT_FOUND_ERROR	Index file STESIYY.XX could not be found.
429	BOF_STESTXX_INDEX_FILE_OPEN_ERROR	Index file STESIYY.XX could not be opened.
430	BOF_STESTXX_INDEX_FILE_READ_ERROR	Index file STESIYY.XX could not be read.
431	BOF_DEVICE_TYPE_VALUE_TO_LARGE	DEVICE-TYPE-NUMBER is too large.
432	BOF_NOT_ENOUGH_MEMORY_IN_CONTROL	The required memory is not avvalable in the selected slot number.
433	BOF_TXEST_KENNUNG_FILE_CREATE_ERROR	The ID FILE for TXEST.XX (SHORT MESSAGES only!) could not be created (TXESK.XX).
434	BOF_TXEST_KENNUNG_FILE_OPEN_ERROR	The ID FILE for TXEST.XX (SHORT MESSAGES only!) could not be opened (TXESK.XX).
435	BOF_TXEST_KENNUNG_FILE_READ_ERROR	The ID FILE for TXEST.XX (SHORT MESSAGES only!) konnte nicht gelesen werden (TXESK.XX).
436	BOF_SYSSTW_KENNUNG_FILE_ CREATE_ERROR	The ID FILE for SYSSTW.XX (SHORT MESSAGES only!) could not be created (SYSSKW.XX).
437	BOF_SYSSTW_KENNUNG_FILE_OPEN_ERROR	The ID FILE for SYSSTW.XX (SHORT MESSAGES only!) could not be opened (SYSSKW.XX).
438	BOF_SYSSTW_KENNUNG_FILE_READ_ERROR	The ID FILE for SYSSTW.XX (SHORT MESSAGES only!) could not be read (SYSSKW.XX)
439	BOF_STESK_KENNUNG_FILE_CREATE_ERROR	The ID FILE for STESTxx.YY (SHORT MESSAGES only!) could not be created (STESKxx.YY)

Code	Designation	Meaning
440	BOF_STESK_KENNUNG_FILE_OPEN_ERROR	The ID FILE for STESTxx.YY (SHORT MESSAGES only!) could not be opened (STESKxx.YY)
441	BOF_STESK_KENNUNG_FILE_READ_ERROR	The ID FILE for STESTxx.YY (SHORT MESSAGES only!) could not be read (STESKxx.YY).
442	BOF_COMPONENT_TYPE_STR_TO_LARGE	The component string in IND_DEV.INI is too long
443	BOF_INVALID_COMPONENT_NUMBER_ERROR	Invalid component number
444	BOF_DEVICE_COMPONENT_TYPE_REQUEST_ERROR	INVALID DATA returned from the inteface in the case of the REQUESTING of the DEVICE-COMPONENT-TYPES.
445	BOF_DEVICE_DAT_FILE_NOT_FOUND_ERROR	Corresponding DEVICE-DAT file does not exist for the BOF configuration.
446	BOF_MAIN_MENU_ITEM_ERROR	Invalid BOF MAIN_MENU_ITEM NUMBER
447	BOF_MAIN_DEF_FILE_CONTENT_ERROR	BOF configuration file \MT_TEXTE\MAIN_DEF.INI does not contain sought DEVICE-TYPE entry
448	BOF_DEVICE_INI_FILE_NOT_FOUND_ERROR	Corresponding DEVICE-INI file does not exist for the BOF configuration
449	BOF_DEVICE_INI_FILE_SYNTAX_ERROR	Format error in the DEVICE-INI file for the BOF configuration

7.3 Error codes from 1000 to 1999

Code	Designation	Meaning
1001	BOF_FAULT_FCT	Invalid function code passed on (e.g. „CW“ for a read function).
1002	BOF_DATA_FAULT	Data invalid.
1003	BOF_FAULT_PIPE_NR	Invalid pipe number
1004	BOF_NO_CREATED_PIPE	Pipe not created.
1005	BOF_PIPE_NOT_RUN	Pipe not running.
1006	BOF_NO_DATA_CREATED	Data not yet generated
1007	BOF_PIPE_NOT_BREAK	Pipe not running.
1008	BOF_NO_VALUE	No value string
1009	BOF_BUFFER_SIZE_TO_SMALL	Buffer too small.
1010	BOF_NO_INDEX_DATA	No index data
1011	BOF_FAULT_INDEX_NR	No index number
1012	BOF_DATA_NO_FOUND	Data not found.
1013	BOF_FUNC_LOCK	Function is blocked, repeat access
1014	BOF_NEGATIVE_ACKNOWLEDGE	Negative Acknowledge for the executed FI command
1015	BOF_PARAMETER_INVALID	Invalid parameter entry
1016	BOF_FUNCTION_INVALID	Invalid FI command
1017	BOF_DEVICE_TIMEOUT	Timeout of CNC task
1018	BOF_INDEX_DATA_ERROR	Index data of result buffer damaged
1019	BOF_UNKNOWN_TOOL_STORE	Unknown memory type != magazine, spindle, gripper.
1020	BOF_MAX_COUNT_ERROR_FOR_TOOL_DATA	Number of memories violated
1021	BOF_NO_TOOLMANAGMENT	Tool management does not exist.
1022	BOF_NO_TOOLMANAGMENT_FOR_PROCESS	Tool management does not exist for a process.
1025	BOF_RESULT_BUF_TYPE_ERROR	Result return type is invalid or is not supported.
1030	BOF_NC_PACKET_IS_PRESENT	NC package already in the controller
1031	BOF_NC_PARTPROGRAM_IS_NOT_PRESENT	Part program does not exist
1032	BOF_NC_PROGRAM_DIRECTORY_IS_EMPTY	Part or machining directory is empty
1033	BOF_NC_PROGRAM_COMPILER_ERROR	Program has set error flag
1034	BOF_NC_DAT_FILE_NO_PRESENT	NC-DAT file does not exist or cannot be opened
1035	BOF_NC_PACKET_DIR_NOT_PRESENT	Package directory does not exist
1036	BOF_NC_PACKET_DIR_READ_ERROR	Package directory cannot be loaded.
1037	BOF_NC_PARTPROGRAM_DIR_NOT_PRESENT	Program directory does not exist
1038	BOF_NC_PARTPROGRAM_DIR_READ_ERROR	Program directory cannot be loaded
1039	BOF_PIPE_CYCLE_LIST_EMPTY	Pipe request list is empty
1040	BOF_PIPE_RUN	Pipe already running.
1041	BOF_ITEM_DATA_INVALID	Part result is invalid.

7.4 Error codes from 2000 to 2999

Code	Meaning
2001	No channel free.
2002	Channel already open.
2003	Channel cannot be closed.
2004	Channel not open.
2005	Reinitialisation error.
2006	Channel cannot be opened.
2007	Version incompatible with the „LOGINTFC.EXE“ file.
2008	Channel flags are blocked.
2010	Receive-Request Timeout.
2011	No request active.
2012	Invalid event in Receive.
2013	Status Request still active.
2014	Cyclic Request still active.
2015	No Cyclic Request active.
2016	Single Request still active.
2017	Transfer format of the „GetSysMsg“ routine contains errors
2018	System message (SysMsg) cannot be sent
2019	DMA-Request is still active.
2020	Invalid code of FI command.
2021	Ivalid result type.
2022	Result too long for receive buffer
2023	Invalid FI command in the case of a group request
2024	Empty result buffer.
2025	Request too long for request buffer
2026	Invalid input format

Code	Meaning
2052	Communication process (COM task) not responding.
2053	„LOGINTFC.EXE“ file not found.
2059	Error message from the LOG process

Code	Meaning
2150	„LOGINTFC.DAT“ file cannot be opened.
2154	FileVersion Mismatch.
2155	„LOGINTFC.DAT“ file is too large.
2156	Internal configuraiton error.
2160	Invalid command string.
2161	Telegram code not implemented.
2162	Parameter outside of limit
2163	Invalid parameter syntax.
2164	Unknown PLC variable.
2165	Not enough parameters passed on
2166	PLC map file cannot be opened.
2167	PLC varaible type not implemented
2168	PLC varaible reference error.
2169	Data item not editable.
2170	Checksum error.
2171	Undefined telegram code.
2172	Processing specification missing.
2173	Too many data items for reply telegram
2174	Unknown additional diagnostic data
2175	Unknown unit.
2176	PLC variable is more than 240 bytes

Code	Meaning
2201	„Date-time“ input string is not in format: „DD.MM.YY hh:mm:ss“.
2202	Excessive SIS telegram user data length

Code	Meaning
2304	Specified file not found

7.5 Error codes from 4000 to 4999

Code	Error text	Error designation and meaning
4000		Error detected when checking the composition of the request of the command BR_NPA1..... and ff. (see also FI command: NPA1_/?) The following error messages in the error window provide further information about the occurring error.
	ERROR : invalid ParNo/value	Invalid parameter number passed on
	ERROR : invalid ParNo/value	Error detected when checking passing on of parameters. Possible cause: invalid parameter designation or an input sequence error. The first parameter number must be less than the second one. (see also FI command: NPA1_/?)
	invalid Parametervalue or No.:[<ParNr>]	Error detected when checking the command. Either a versoin number has been chosen outside the validitiy range or a parameter designation is invalid.
	[Nr.] missing Startparameter	Command not communicated completely
	Illegal Startparameter value [falsche Übergabe]	Invalid designation detected in the parameter number
	ERROR : different Parametertypes	Requesting different parameter types in one request command is not possible
	ERROR : Second ParNo before First ParNo	The parameter request must take place from the lower to the higher significance number. (see also FI command: NPA1_/?)
	ERROR : Invalid startparameter - ProcNo out of Range	A violation of the definition range has been detected when requesting one or several process parameters. A request is only possible in the range of the CNC process number [0...6].
	ERROR : Invalid startparameter - AxesNo out of Range	A violation of the definition range has been detected when requesting one or several axis parameters. A request is only possible in the range of axis numbers from 1 to 20 or 32.
4001	ERROR : invalid function	The FI command contains an invalid parameter input.
4002	NO_PARAMETER_DATA_FOUND	The requested parameter(s) do(es) not exist. Either parameters have been requested that were not defined beforehand or the affiliated parameters have been removed. Check all entries and check whether the affiliated data exists in the user interface menu item F5 (Parameters).
4003	Verz_No_Out_of_Range	A range violation has been detected when checking the command passed on. Check your entered directory number.
4004	BR_NPA_No_Data_File_exists	The attempt to read data from a file could not be executed. Check your input again in relation to the definition range of possible processes or axes. Otherwise, attempt to view the data via the user interface menu item F5 (Parameters). The data possibly does not exist or the installation contains an error. In this case, you should contact after-sales service.
4005	BR_NPA_No_INI_File_exists	Parameter data from an initialisation file could not be read. Possible causes: • The file does not exist. There is an installation error or the file has been inadvertently deleted. ⇒ Run an update installation • The file contains errors. It has been inadvertently edited or illegally copied, rendering the file identifier invalid. ⇒ Run an update installation or contact after-sales service. • The file has been damaged either by a system crash or by a defect on the storage medium. ⇒ Contact after-sales service.

Code	Error text	Error designation and meaning
4006	Device Adress out of Range	A system outside the definition range has been selected in the command.
4007	Buffererror detected =[error code]	Internal error. The defined data area for provision of the result is not large neough: • Request smaller data quantities. Use group requests. • Increase avaialble memory for the data area of a user-created application. ⇒ Contact after-sales service.
4012	Create_DLL_Error detected!	Could not initialise the result buffer. Contact after-sales service.
4013	Function will not running by DLL-VersionMode:[DLL-Version]	An attempt has been made to run a command that is not yet available in the existing DLL version.
4014	Corrupted Parameter Intentification = [Parameterindent.]	Initialisation of the necessary data memory is not possible owing to an invalid parameter identifier. Check whether a valid parameter set exists for all devices. If necessary, again transfer the parameter(s) to the controller. If the error should persist or if transfer of the parameters is not possible contact after-sales service.
4015	wrong Version installed	This error message appears in the starting phase of the user interface whenever initialisation of the memory is not possible owing to the existing version. Error code 4109 was returned up to and including Version 18. The corrected error code 4015 appears as Version 19.
4017	**OK** (none Parameterset in CNC) - finished function FillParamDataInCncDataMap	This message only appears in the starting phase with the setting „/U0“ of the start parameters (in the case of TSRPG25I.EXE) if an emty parameter entry has been passed on. This means that htere is not yet a parameter set in the controller. An error is not returned.
4100	Couldn‘t open ParameterIndexFile:[File Error=xxxx]	Error detected when attempting to open a parameter directory file. The following possibilities might trigger this error: • Versions do not agree ⇒ Conversion of parameters necessary. (see Parameter conversion) • The parameter directory file has been inadvertently deleted. • The drive is defective.
4102	ParameterIndexFile has wrong structure	Error detected when reading out the parameter directory data which indicates that data is not in the correct format. Check this by running Parameter conversion. You must contact after-sales service if the error should persist.
4103	to much Indizies found - File has wrong structure ?:	Error detected when creating the directory data. More irectories detected than permitted by the definition range. This probably consists of parameter directory data belonging to an earlier version. Run a Parameter conversion. You must contact after-sales service if the error should persist.
4104	invalid Parametervalue detected	Range violation detected during initialisation (start-up of the interface). Contact after-sales service.
4105	Can' t create Parameterindexbuffer: [filename]	No data could be provided in the memory. Close other applicatoins to enable sufficient memory for the data.
4108	Don't found the Parameter: [Parameter number]	The message appears only in the starting phase with the „/U0“ setting (in the case of TSRPG25I.EXE). The error code is returned whenever an attempt has been made to request an undefined parameter.
4109	Didn't get BOF-Version - BOF installed? [Error code]	Attempt to determined interface version failed. Contact after-sales service.

Code	Error text	Error designation and meaning
4110	Couldn't load Parameters in shared Memory - Error= [ErrorCode]	Initialisation not possible while starting the interface. Contact after-sales service.
4111	invalid Parameter value Cxx.053 [Cxx.053 <Wert>]	Initialisation not possible while starting the interface. Invalid axis designation detected in the current parameter set of a device. Set the system concerned to offline and correct the parameter set. After correction, set the system back to online and then transfer the corrected parameter set to the controller. If the problem persists, contact after-sales service.
4200	invalid Start Parameter	The message appears only in the starting phase with the „/U0" setting (in the case of TSRPG25I.EXE). The error code is returned whenever an attempt has been made to request an a parameter outside the definition range. If the problem persists, contact after-sales service
4201	invalid Parametertype	Parameter query with an undefined parameter type. Check your input or request.
4202	Buffersize not enough	The result of the parameter request cannot be passed on owing to the provided transfer area. In the case of user-created applications, increase the size of the transfer area. Otherwise, contact after-sales service.
4203	Error detect by ReadPar_Value - can't read Data [Error number or directory number]	The requested parameter could not be generated or found. Check your request again or contact after-sales service.
4204	didn't found Verz Entry	The error message was not output. The error code is always returned when the **parameter number** has not been found after a query for a specific parameter directory entry.
4205	Function will not running by InterFace-Version: [Version]	On command request it was found that execution is not possible in this version. Contact after-sales service.

Code	Designation	Meaning
4220	invalid Save Order by Save function please test the ParType by Save_Begin;	The „Write parameters“ function has been repeatedly started before the previously started execution was ended.
4221	invalid IndexNo by Save[ParameterNummer	Parameter number outside definition range.
4222	co_str_ConWData_Buffer_Size_to_small [defined size 2000]	The message appears only in the starting phase with the „/U0“ setting (in the case of TSRPG25I.EXE). The error code is returned if the memory area defined in the program is not adequately dimensioned. In this case, you must contact after-sales service.
4223	WriteError by Config-SCR-File = [Error number]	Error detected when writing the configuration parameters. Function aborted. Contact after-sales service.
4224	SaveError – Couldn't rename DAT->old [File name]	The message appears only in the starting phase with the „/U0“ setting (in the case of TSRPG25I.EXE). Attempt to rename original file failed. Check the properties of the affiliated parameter file and the free memory of your storage medium.
4225	SaveError – Couldn't rename tmp ->dat => copy old to DAT[File name]	The message appears only in the starting phase with the „/U0“ setting (in the case of TSRPG25I.EXE). Attempt to recopy the newly created original file failed. Check the properties of the affiliated parameter file and the free memory of your storage medium
4226	Missing File = [File name]	The previously created file could not be found or opened. Check the free memory on your storage medium.
4227	Create Instance failed - can't save Data	Internal error has occurred. Contact after-sales service.
4228	Can't create File = [File name <Zusatzinfo>]	The specified file could not be created in accordance with the additional info. Check the free memory on your storage medium.and the access properties of the affected directory. Otherwise, contact after-sales service.
4229	Can't open File = [File name]	The specified file could not be found. Check your entry. An invalid directory number may have been specified.
4230	ConWData_Error_by_WPar_Begin	This error code is returned after the occurrence of an error on transfer of an error code to the function interface.
4231	ConWData_Error_by_WPar_End	This error code is returned after the occurrence of an error on transfer of an error code to the function interface.
4232	Error detect by WritePar_Value - can't save Data:[File name bzw. Parameterwert]	The error message is only indicated internally in the 'debug mode'. The value of a parameter could not be transferred to the specified file. Contact after-sales service.
4233	Attention - Return value of Process-Definition undefined	The error message is only indicated internally in the „debug mode“. Error detected on generation of the process definitions. Within the scope of parameter editing, check the process definitions. You may have to contact after-sales service.

Code	Designation	Meaning
4234	Can't actualize VerzLine [Parameter directory line]	The specified parameter directory line could not be updated.
4235	Can't actualize Date or Length in Verzline	The date or length could not be updated when updating the parameter directory line. Contact after-sales service.
4236	CreateFiErrorResult_DLL failed	The error message is only indicated internally in the „debug mode". Could not transfer the error message to the FI. Contact after-sales service.
4237	Can't write by undefined Parameternumber [Parameter number]	Attempt made to write an undefined parameter for this parameter type. Check your input. Pay attention to parameter availability of the various axis types, for example.
4238	Cxx.083 : more as defined Elements for Cxx.083 found:	Attempt made to transfer a larger number of compensatoin values than permitted by the max. range of a compensation list. Check your input. A maximum of 1000 values can be placed in the list.
4239	Installation Error - Missing File: [File name]	Specified file not found. Run an update installation. If the error should persist, contact after-sales service.
4240	Invalid Parametervalue =[Parameter line]	Range violation detected in the specified parameter line. Check your input again.

Parameter conversion

With an update installation of the Indramaet user interface, a parameter conversion from version „xx" to the next version „yy" is run automatically. A parameter conversion is run by calling up the DOS program „COPAxxyy.EXE" in the standard installation directory „C:\MT-CNC\". The two placeholder parameters „xx" and „yy" represent the from-to version identifiers.

Note: In the event of an error, you can start the conversion program „COPAxxyy.EXE" with the „/?" start parameter for further information.

7.6 Error codes from 5000 to 5999

Code	Designation	Meaning
5001	NO_REQUEST_ACTIVE	No request active
5002	INVALID_COMMANDSTRING	Invalid command string
5007	INVALID_LOGICALDEVICE	Invalid logical device (FAR-Device)
5401	NET_EFALSE	unspecific error
5402	NET_EINVPARAM	invalid parameter passed to function
5403	NET_ETIMEOUT	transfer timeout, remote PC not ready, or network connection down
5404	NET_ESND_ERROR	send failed; error sending to a remote PC
5405	NET_ENOMEM	memory shortage; in Remote access of the interface
5406	NET_EINVCONN	invalid connection to a remote PC
5407	NET_ESERVDIS	service disabled
5408	NET_EABORT	Connection to remote partner aborted
5409	NET_EINVHOOK	invalid parameter hook id; Sys Message Handling
5410	NET_EINVDEVICE	invalid device number

7.7 Error codes from 6000 to 6999

Code	Designation	Meaning
6001	BOF_C_TYP_FAULT	Transferred data type not OK
6002	BOF_C_LEN_FAULT	Transferred data length not OK
6003	BOF_C_DEV_FAULT	Transferred system number not OK
6004	BOF_C_PAKNR_FAULT	Transferred package number not OK
6005	BOF_C_PROZ_FAULT	Transferred process number not OK
6006	BOF_C_PROG_FAULT	Transferred program number not OK
6007	BOF_C_FILE_NOT_DEL	File cannot be deleted
6008	BOF_C_NO_NCPROG_CREATED	NO NC program in part prgram directory
6009	BOF_C_NCPROG_CREATED	NC program available (if check =1)
6010	BOF_C_DESCR_FAULT	Designator, e.g. data length not OK
6011	BOF_C_FILE_WRITE_CLOSE_ERROR	Error writing or closing a file
6012	BOF_C_PACK_EXIST	NC package already exists
6013	BOF_C_INVALID_MTCNC_NUMBER	Invalid system number
6014	BOF_C_FILE_NOT_FOUND	File not found
6015	BOF_C_PAR4_FAULT	Parameter 4 not OK
6016	BOF_C_NO_NC_SEEK_SET	NC program cannot be positioned at N0000
6017	BOF_C_NCPROG_NOT_READ	File cannot be opened
6018	BOF_C_PART_PROGR_DIRECTORY_ERROR	Part program directory could not be read.
6019	BOF_C_PACKET_DIRECTORY_ERROR	Package program directory could not be read.
6020	BOF_C_PAR5_FAULT	Parameter 5 not OK
6021	BOF_C_PAR6_FAULT	Parameter 6 not OK
6022	BOF_C_COMP_ERROR	Error in test after commands to compile
6023	BOF_C_CURS_FILE_ERROR	Handling error with with the NCCPxx.DAT file.
6024	BOF_C_TOOL_SETUP_LIST_NOT_READ	Setup list contains errors
6025	BOF_C_TOO_MUCH_TOOLS_IN_LIST	More tools in the setup list than in parameters

7.8 Error codes from 7000 to 7999

All error codes - with the exception of error code 7000, which indicates a syntax error in the compiled NC program - generally require contact with Rexroth Indramat for further clarification of the cause of the error. Either there is a software error or files belonging to the BOF/GBO user interface have been deleted or damaged.

Note: As with all error codes, further information about an occurring error can be requested via the „**Fehler! Verweisquelle konnte nicht gefunden werden.**„. The error information provides theuser with plain language information about the cause of an error.

Code	Meaning and diagnostic notes as well as remedy notes
7000	Syntax error in the NC program. „**Fehler! Verweisquelle konnte nicht gefunden werden.**„ contains further information.
7002	File with incorrect information. „**Fehler! Verweisquelle konnte nicht gefunden werden.**„ contains the file name and the line.
7005	File not found. „**Fehler! Verweisquelle konnte nicht gefunden werden.**„ contains the file name.
7006	File cannot be created. „**Fehler! Verweisquelle konnte nicht gefunden werden.**„ contains the file name.
7008	File cannot be read. „**Fehler! Verweisquelle konnte nicht gefunden werden.**„ contains the file name.
7009	Error in the functoin interface link. No connection with the device (controller) can be established via the function interface.
7015	Too many axes defined. More than 9 axes used in the CNC process.
7016	Invalid number of parameters. The number of parameters in the „NCPRG.CFG“ file has been exceeded.
7017	Invalid axis designation. The axis designation in the axis parameters „CXX.001“ or „CXX.075“ is invalid.
7018	Axis designation is invalid. The axis designation in the axis parameter „CXX.053“ is invalid.
7019	Maximum axis speed is invalid. The value of the axis parameter „CXX.016“ is invalid.
7020	Maximum axis acceleration is invalid. The value of the axis parameter „CXX.018“ is invalid.
7021	Shortest execution time of an NC block = [2.5...30ms]. The numeric value of the „METB“ in the NC options of the BOF/GBO is outside the permitted range.
7022	Shortest execution time of an NC block invalid. The numeric value of the „METB“ in the NC options of the BOF/GBO is invalid.
7023	Only 4 or 5 decimal places allowed. The process parameter „BXX.002“ is invalid.
7024	Invalid numeric value. The numeric value of the „VFBT“ or „BBTC“ parameters in the NC options of the BOF/GBO is invalid.
7025	Only 0 (mm) or 1 (inch) allowed! The process parameter „BXX.001“ is invalid.

Code	Meaning and diagnostic notes as well as remedy notes
7026	Numeric value outside the permitted range. The axis parameter „CXX.006“ is less than 0.1.
7027	Internal block number invalid. The block numbers in the NC program file are in a wrong order.
7028	Block number in the file invalid. „**Fehler! Verweisquelle konnte nicht gefunden werden.**„ contains the name of the file in which the block numbers are not correct.
7070	Numeric value outside the permitted range (1..10). The numeric value of the „BBTC“ parameter in the NC options of the BOF/GBO is outside the permitted range.
7077	Numeric value outside the permitted range (1..25). The numeric value of the „VFBT“ parameter in the NC options of the BOF/GBO is outside the permitted range .
7083	Parameter invalid. The „**Fehler! Verweisquelle konnte nicht gefunden werden.**„ contains the invalid controller parameter.

7.9 Error codes from 8000 to 8999

Code	Designation	Meaning
8000	OUTOFMEMORY	Heap memory used up
8001	PARAMETER_FAILURE	Error in parameter transfer (reply telegram)
8002	INVALIDARG	Invalid request string
8003	REQUEST_NOT_FILLED	Internal execution error
8004	GET_ATTRIBUT_FAILED	Invalid attribute in the reply telegram
8005	WALK	Internal execution error
8006	EXTRACT_COMMON_INFO_FAILED	Error in parameter transfer (reply telegram)
8007	WRONG_DATA_SIZE	Undefined data length in the reply
8008	ELEMENT_UNEXPECTED	Unexpected coding in BW_SPA1
8009	SERCOS_LONG_TO_ASCII	Error in result conversion
8010	VERSION_MISMATCH	Command did not yet exist for set IfDllMode.
8011	ERROR_BYTE_INFO	Error reading out the error byte information
8012	CANT_OPEN_MODULDEF_INI	The „Moduldef.ini“ file cannot be opened.
8013	WRONG_PROFILE_FILENAME	Invalid profile file name
8014	WRONG_SECTION_INFORMATION	Invalid section information in the profile
8015	ERROR_IN_LAST_LINE	Error in the last profile line
8016	Reserved	Reserved
8017	Reserved	Reserved
8018	SECTION_NOT_FOUND	Section not found; (e.g. invalid device or module parameter).
8019	LANGUAGE_NOT_FOUND	Language not supported
8020	Reserved	Reserved
8021	MODUL_NOT_FOUND	Module not found (e.g. missing module name key word).
8022	DEVICE_ADDR_GENERAL_NOT_FOUND	No device entry found.
8023	FB_NOT_FOUND	No functoin block found (e.g. key word errors or messages missing).
8024	DEVICE_ADR_FALSE	Device address not in the valid range
8025	MODULE_NO_FALSE	Module number not in the valid range (0-99)
8026	KEY_WORD_FALSE	Key word is invalid (e.g. no ModulY in the section name [DeviceAddrX\ModulY])
8027	MODULE_ASSIGN_PROCESS	No module can be found for the specified process.
8028	PROCESS_NO_FALSE	Process number not in the valid range (0-31).
8031	RESULT_TYPE_INVALID	Invalid result type
8032	E_COM_SIS_TEL_TOO_LONG	Transferred telegram length exceeds maximum SIS telegram length.
8033	E_COM_SIS_TEL_POS	Addressed telegram position outside of the SIS telegram range.
8034	E_COM_SIS_TEL_NO_LEN	SIS telegram does not have the length „0“
8035	E_COM_OPERATING_SYSTEM_ NOT_SUPPORTED	Operating system not supported.
8036	SERCOS_ASCII_TO_LONG	Error converting the value to write.

7.10 10000 error codes

Code	Meaning
10001	The WIN-HMI component is not installed.
10101	Invalid version of function interface.
10102	The „CreateGroup“ routine has failed.
10103	Error in command string.
10104	Unknown variable requested.
10105	Status determination error.
10107	The „HMI_Data.DLL“ has not been found.
10110	WIN-HMI not started in the same process.

7.11 100000 error codes

Code	Designation	Meaning
100101	FS_NO_TEXT_FILE_ACCESS	Opening record file in text mode.
100102	FS_REC_SIZE_TO_SMALL	Invalid record size
100103	FS_REC_FILE_BOUND_ERROR	Invalid file position
100104	FS_NO_CREATE_OBJECT	Could not generate an interface object.
100105	FS_ERROR_SIM5	No „iMTc“ identifier
100106	FS_ERROR_FILETYP	Different file type
100107	FS_ERROR_FILEVERSION	Current file version greater than file version
100108	FS_ERROR_FILELENGTH	Current file length != identifier length
100109	FS_ERROR_FILEDATE	File date != identifier date
100110	FS_ERROR_FILETIME	File time != identifier time
100111	FS_ERROR_FILENAME	File name != identifier name
100112	FS_ERROR_CHECKSUM	Checksum is not correct
100113	FS_ERROR_FILE_NOT_EXIST	File does not exist
100114	FS_ERROR_FILE_MIN_LENGTH	File with identifier must be at least 65 bytes long.
100115	FS_ERROR_T04	No „iT04“ identifier
100116	FS_ERROR_FILE_NOT_OPEN	File cannot be opened
100117	FS_ERROR_NO_SIGN	File has no identifier
100118	FS_ERROR_MMIVERSION	Interface version is less than file version.

7.12 110000 error codes

Code	Designation	Meaning
110001	BOF_MAP_VERSION_FUNC_ERROR	Invalid Dll mode is set
110002	BOF_MAP_FILE_VERSION_ERROR	Invalid file version number
110003	BOF_MAP_LANGKENNUNG_VERSION_ERROR	When long identifier version is invalid
110004	BOF_MAP_LANGKENNUNG_INVALID_ERROR	When long identifier is invalid
110005	BOF_MAP_LANGKENNUNG_PARAM_ERROR	Missing parameter for split long identifier
110006	BOF_MAP_COMMON_ERROR	Error not clearly defined
110007	BOF_MAP_FILE_NOT_OPEN	File could not be opened.
110008	BOF_MAP_FILE_IS_OPEN	File is already open.
110009	BOF_MAP_PLAUSIBLE_TEST_ERROR	Plausibility check of map file long identifier negative.
110010	BOF_MAP_KENNUNGS_ERROR	Long identifier comparison negative.
110011	BOF_MAP_TO_MANY_IMPORT_TAB_ENTRIES	Too many import table entries (>65535).
110012	BOF_MAP_INVALID_DATA	Map file contains invalid data.
110013	BOF_MAP_PARAMETER_INVALID	Missing parameter for a function.
110014	BOF_MAP_INVALID_DEVICE_NO	Transferred device number does not agree with the system number in MAP.
110015	BOF_MAP_INVALID_STATUS	Access status is invalid
110016	BOF_MAP_ACCESS_ERROR	Access to a MAP without MPA having been loaded, invalid DeviceNo
110017	BOF_MAP_NO_LOAD_ERROR	MAP file is not loaded internally
110018	BOF_MAP_NO_LOAD_MAPFILE_ERROR52	MAP file is not loaded with error 52
110019	BOF_MAP_NO_LOAD_MAPFILE_ERROR55	MAP file is not loaded with error 55
110020	BOF_MAP_MAPFILE_INVALID_VERSION_ ERROR56	Map file has invalid version 56
110021	BOF_MAP_VARIABLE_NO_FOUND_ERROR	13...46 → Variable not found.
110022	BOF_MAP_LANGKENNUNG_DIFFERENT_TO_ MAP12	MAP file long identifier differs from PLC long identifier
110023	BOF_MAP_INVALID_ARRAY_INDEX50	Invalid array indexing
110024	BOF_MAP_INVALID_STRING_INDEX51	Invalid string indexing
110050	BOF_MAP_COMMON_FILETOOL_ERROR	Basic number cErrorGruppe_filetool
110100	BOF_MAP_COMMON_MAP_BAS_C_ERROR	Basic number ErrorGruppe_map_bas_c
110150	BOF_MAP_COMMON_LKENN_ERROR	Basic number cErrorGruppe_lkenn
110200	BOF_MAP_COMMON_GROUP_ERROR	Basic number general error
110263 110296	BOF_MAP_VARIABLE_NO_FOUND_ERROR_ BASE+13 .... BOF_MAP_VARIABLE_NO_FOUND_ERROR_ BASE+46	PLC variable does not exist so the error can be clearly identifier; error number added to the BASE.

8 Frequently asked questions: FAQ

8.1 Function interface FAQs

This chapter summarises questions and answers (FAQ = **F**requently **A**sked **Q**uestions) about the Indramat function interface from customer feedback.

Question 1 A message box appears when my application starts. Is the message box sent by the function interface?

Note: As message boxes are entered as „applications“ in the Windows NT Task Manager, it is fairly easy to find out who is sending the message box.

Answer Open the Windows NT Task Manager, e.g. with the keystroke combination: Ctrl + Shift + Esc

In the „Applications“ tab, mark the message box entry and right click it.

Note: The keystroke combination Shift +F10 for the right mouse click does not function here!

In the context menu that is opened for the marked object, select the „Switch to process“ command.

If one of the following processes is now displayed:

- LOGINTFC.exe
- BOFINTFC.exe
- COMINTFC.exe

this is a basic process of the function interface.

Question 2 Can group requests also be sent by means of the „DataTransfer“ routine?

Answer No, the „DataTransfer“ routine only serves to send single writing or reading requests. Group requests are sent via the routines for cyclic reading via pipes.

Question 3 Why does login by my application in the function interface take so long?

Answer During the initialisation phase of the function interface, a number of security checks are run (see chapter entitled „Function interface structure with configuration data „).

8.2 Windows NT FAQs

This summarises questions and answers about Windows NT from customer feedback.

Question 1 How do I log in automatically with my name and password (AutoLogin)?

Answer You must make the following entries in the windows NT registry with the registry editor „REGEDT32“ under the key

HKEY_LOCAL_MACHINE\ Software\ Microsoft\ Windows NT\ Current Version\ Winlogon

Value	Type	Contents	Info
AutoAdminLogon	REG_SZ	1	Autologin On/Off
DefaultUserName	REG_SZ	user_name	User’s login name
DefaultPassword	REG_SZ	Password	User’s password (a password must exist)
DefaultDomainName	REG_SZ	DomainName	Login via a different computer

Note: A login box no longer appears. If you wish to log in via a different name, you must keep the Shift pressed during start-up. You will then be prompted for your name and password again.

Note: If no password is entered in the Registry, AutoLogin functions once only and Windows then sets „AutoAdminLogon“ back to „0“. Therefore, it is imperative to specify the password. Remember that this means that the password is visible to all and sundry in the Registry!

9 Glossary

ANSI **A**merican **N**ational **S**tandards **I**nstitute; developed the ANSI emulation (see also: ANSI code).

ANSI code Code that was standardised by ANSI and which makes it possible to generate images, animations and texts as well as tones from a PC loudspeaker out of strings of ANSI control frequencies. A method mainly used in mailboxes to design a user interface. In most cases, the ASCII code is also referred to as the ANSI code. The characters are generated in a document by pressing the <AltGr> key, in combination with input of the relevant code.

ASCII **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange, widely spread code, used particularly on home and personal computers to represent digits, letters and special characters; consists of a 7-bit code with a character set of 128 characters or of an 8-bit code with a character set of 256 characters including upper and lower case letters. The free eighth and ninth bits in the byte format (Byte) serves as the parity bit (check bit).

Operating data item The operating data item is understood to be data block element 7 of a parameter. It contains the value of the parameter.

Client A client is understood to be a computer system or a process that requests the services of another computer system or process. It also designates a workstation computer that is able to use the services and resources (e.g. printers, scanners or plotters) of a server or of other clients. In doing so, in most cases it has clearly less access privileges than the server.

DDE **D**ynamic **D**ata **E**xchange is a standard defined by Microsoft for cross-program data exchange under MS-Windows, Version 3.0 or greater. DDE enables a combination of files or parts thereof between two applications that support the DDE standard. At the same time, a distinction is made between a source application (server) and a target application (client) and the target application accepts a complete or partial copy of the server file. If the data in the source file is modified, this information about the link is communicated to the target application, where it is dynamically updated. DDE communication can, however, also be used simply to exchange commands and instructions between two programs.

DLL **D**ynamic-**L**ink **L**ibrary is a library that is linked to a program during the run time. DLLs are special files for Windows out of which function, dialog boxes or symbols, for example, can be loaded by applications. They facilitate programming and save hard disk capacity, e.g. if identical functions are needed by several user programs. A dynamic link library offers several advantages: it only needs to be loaded whenever required and does not use up any memory until needed.

Remote device This term depends on the point of view (see also Local Device). From the point of view of a user program, the device is **not** connected to the PC on which the user program is running, but on a PC in the PC network.

Device A device is understood to consist of an item of control hardware, a drive or an I/O device.

Device address The device address corresponds to the system address within the BOF/GBO. For example, device 00 corresponds to system 0. Pay attention to the fact that the BOF/GBO always requires a device 00.

Device type The device type specifies the Indramat device involved, e.g. MTC200-P, MTVNC, ISP200-R, etc.

FarDevice The configuration of the PC network not only includes the list of the PCs, but also the list of the FarDevices. The FarDevice addresses were introduced for linear and unique addressing of the devices in the PC network. Thus, each device that is available in the PC network is assigned an additional address (FarDevice address). A device that is available in the PC network has a FarDevice address and is referred to as a FarDevice. The list of the FarDevices is defined on the basis of the devices previously created on each PC.

Local device This term depends on the point of view (see also Remote Device). From the point of view of a user program, the device is connected to the PC where the user program is running.

MPI (**MPI** = **M**ulti-**P**rotocol **I**nterface). The Indramat MPI realises a standardised user interface for the Profibus FMS, MMS-Ethernet (MAP), TCP/IP and FIPWAY communication interfaces on PC modules under the Windows NT 4.0 operating system. The MPI provides the interfaces for the realisation of client and server programs. The MPI communication driver links the MPI to the function interface, thus realising the link to all protocols supported by the MPI to the Indramat devices that are configurable via the function interface.

PC network The PC network is a combination of several PCs at the level of the function interface. The PC network embraces the PCs that are used for the control (mainly visualisation, operation and programming) of a machine.

Server A server is understood to be a computer that contains applications and documents to which connected computers (clients) have access. The term also refers to a program that offers specific services which can be accessed with the aid of programs that have been specially adapted to the server.

Thread Threads are objects within processes that executes program instructions. They permit simultaneous actions within a process and make it possible to simultaneously run different parts of the program on different processors.

10 Index

_

_default.pif file 4-7

<

<Alt>+<Tab> keystroke combination 4-2

A

AAC Active **AC**celeration 6-21
AAD Active **A**ngle **D**imension 6-22
AAS Actual **A**xis **S**peed 6-23
ABI Active NC-**B**lock **I**nformation 6-24
ACS Active **C**utting **S**peed 6-25
Add module 3-3
ADN Active **D**-Correction **N**umber 6-26
AEM Active **E**vent **M**onitoring 6-27
AEN Active **E**dge-**N**umber 6-28
AFO Active **F**eedrate **O**verride 6-29
AFR Actual **F**eed**R**ate 6-30
AGF Active **G**-**F**unction 6-31
AMF Active **M**-**F**unction 6-32
AMM Active **M**echanism **M**essages 6-32, 6-149
ANSI 5-3
API Active Machine-**P**arameter **I**ndex 6-42
APM Active Part-**P**rogram **M**essage 6-45
APN Active Part-**P**rogram Message-**N**umber 6-46
APO Actual Axis **PO**sition 6-46, 6-173
APP Active **P**art-**P**rogram number 6-48
ARO Actual **R**apid **O**verride 6-49
ASCII 5-3
ASE Actual **S**ERCANS **E**rror 6-144
ASF Active **S**pindle **F**or Process 6-50
ASG Actual **S**pindle **G**ear 6-51
ASM Active **S**ystem **M**essages 6-52, 6-157, 6-174
ASN Actual **S**equence **N**umber 6-59
ASO Actual **S**pindle **O**verride 6-59
ASS Actual **S**pindle **S**peed 6-60
ATN Active **T**ool-**N**umber 6-61
AutoLogin 8-2
AZB Active **Z**ero-Offset **B**ank 6-62

B

B = **B**reak Cyclic Read 5-2
Basic processes 2-1
Binary 5-3
BOF process 2-1

C

C = **C**yclic Read 5-2
Calling up the Task Manager 4-2
CCP Cell **C**onfiguration **P**arameter 6-1
CMA CMos Ram **A**SCII Parameter 6-183
CMI CMos Ram **F**loatingpoint Parameter 6-181, 6-182
CMI CMos Ram **I**nteger Parameter 6-180
Common-MAP 2-7
Communication process 2-1
Complete reinitialisation 3-31
CPO Command **PO**sition 6-63
CRT Control **R**ese**T** **6-65**, 6-163, 6-184

D

DAC Device **A**xis **C**onfiguration Parameter 6-67
Data-MAP x 2-7
DataTransfer 3-10, 3-11, **5-2**
DCP Device **C**onfiguration **P**arameter 6-70, 6-185
DCR D-**C**orrection **R**ecord 6-73
Default installation 4-13
Device address 5-1
DFJ Delete Function-Interface Jobs 6-8
Diagnostic information 3-35
Diagnostics 3-35
DIS Data **I**dentification **S**tring 6-76, 6-164, 6-188
DLL 1-1
DTG Distance **T**o **G**o 6-77, 6-188
DTY Device **TY**pe 6-77, 6-164, 6-188
Dual port RAM 1-1, 2-2

E

EPO End**PO**int 6-78
Example 1 (cyclic reading) 5-3
Example 2 (cyclic reading) 5-3
Example 3 (cyclic reading) 5-4

F

FCP Far Device Configuration Parameter 6-13
FDC Far **D**evice **C**onfiguration 6-16
FPC Far **P**C **C**onfiguration Parameter 6-18
Function code 5-2

G

GPC Global **P**rocess **C**onfiguration 6-80
GPC Global **P**rocess **P**arameter 6-84

I

Ident number 6-126, 6-140
IFJ Information about **F**unction-Interface **J**obs 6-9
INDIF000.C 3-2
INDIF000.H 3-2, 3-4
INDIFX00.H 3-2, 3-4
Indramat system configurator 4-19
INDRAMAT user interface WIN200 2-4
INDRAMAT.C 3-2
INDRAMAT.H 3-2, 3-4
Installation from CD-ROM 4-9
Installation from diskettes 4-9
Interface designator 5-1
IPP Insert NC-**P**rogram **P**ackage 6-105

K

Kernel 2-2, 4-23
Kernel mode 4-23
Kernel mode driver 4-23
Kernel-mode 2-2, 4-8

L

LNG Activated LaNGuage 6-12
Logic channel 2-1
Logic process 2-1
LogInIf 3-6
LogOutIf 3-8

M

MAP Modul **A**ssign of **P**rocess 6-87, 6-165, 6-189
Maximal FeedRate 6-94
MCD Modul **C**onfiguration
Device Information 6-166, 6-190
MCD Module **C**onfiguration
Device Information 6-88
MCM Modul **C**onfiguration
Modul Information 6-89, 6-167, 6-191
MCP Modul **C**onfiguration
Process Information 6-90, 6-168, 6-192
MCS Modul **C**onfiguration
Process Information 6-91, 6-169, 6-193
Menu item
Project 3-3
MFO Maximal **F**eedrate **O**verride 6-92
MFR Maximal **F**eedrate **O**verride 6-94
MPI (Multi-Protocol Interface) 2-3
MRO Maximal **R**apid **O**verride 6-95
MS command prompt 4-8
MSO Maximal **S**pindle **O**verride 6-96
MSS Maximal **S**pindle **S**peed 6-97
MTCNC00I.SYS 2-2, 4-23
MTD Machine **T**able **D**ata 6-98

N

NEV NC-**EV**ent 6-99
NMM NC-**M**e**M**ory 6-101
Non-preemptive multitasking 4-1
NPA NC-**PA**rameter 6-102
NPC NC-**P**ackage **C**ompiling 6-109
NPD NC-**P**ackage **D**ownload 6-110
NPI NC-**P**ackage D**I**rectory 6-104
NPS NC- **P**rogram **S**election 6-112
NTN Next **T**ool-**N**umber 6-113
NVS NC-**V**ariable **S**ingle 6-114

O

Objective 1-1
OPD Optimal **P**osition **D**istance 6-116

P

Parameterised device 2-2
PFR Programmed **F**eed**R**ate 6-118
physical communication address 1-1
PPD Part-**P**rogram **D**irectory 6-106
PPN Part-**P**rogram **N**C 6-107, 6-108
PPS Part **P**rogram **S**equence 6-119
Preemptive multitasking 4-1
PRO version 4-14
Problem 3-30
Programming languages 3-1
Properties of Date/Time 4-4
PSS Programmed **S**pindle **S**peed 6-120
PVS PLC-**V**ariable **S**ingle 6-121, 6-170, 6-195

R

R = Single **R**ead 5-2
Read and write command 5-2
Read command **R 5-2**
ReadGroupItem 5-2
Remedy 3-30

Requirements 1-1
Right mouse click 3-37
RS232-C 2-2
RS485 2-2
RS485 interface 2-2

S

Selector 5-3
Separator 5-1
serial interface 1-1
Serial number 4-12
Shared memory 2-2
Shared memory area 1-1
Simulation pair 4-19
SLA Servo**LA**g 6-123
Software licence agreement 4-12
SPA1 Sercos **Pa**rameter 6-125, 6-139
SPH Sercos **Ph**ase 6-128, 6-143
SPP Selected **P**art-**P**rogram number 6-129
Start bar 3-37
SWD-CONTRL-FUN-04VRS-MS-C1,44 3-2, 4-9
SWD-CONTRL-FUN-04VRS-MS-CD600 4-9
SWS-FUN*PC-PRO-04VRS-MS 3-2, **4-14**, 4-24
SYS-MSGs 3-1
System configuration 4-19
System configurator 2-7, **4-19**
System messages 3-1
System-MAP 2-7

T

Tabs 3-37
Task Manager 4-2
TQE TorQuE 6-130
Type designation 4-14

U

Unicode 5-3
User interface process 3-35

V

Validity 1-1
Virtual device driver 2-3
Visual Basic 3-3
Visual C/C++ 3-2

W

W = **W**rite 5-2
Windows NT kernel-mode driver 4-23
Windows NT kernel-mode driver (MTCNC00I.SYS) 2-2
Windows NT Task Manager 4-2, 8-1
WinHMI 2-4
WinMTC 2-4
Write command **W 5-2**

Z

ZOD Zero **O**ffset **D**ata 6-132

11 Kundenbetreuungsstellen – Sales & Service Facilities

Deutschland – Germany

vom Ausland: (0) nach Landeskennziffer weglassen!!
from abroad: don't dial (0) after country code!

Vertriebsgebiet Mitte ☒ SALES Germany Centre ☒ Service INDRAMAT GmbH Bgm.-Dr.-Nebel-Str. 2 D - 97816 Lohr am Main Telefon: +49 (0)9352/40-0 Telefax: +49 (0)9352/40-4885	Vertriebsgebiet Ost ☒ SALES Germany East ☒ Service INDRAMAT GmbH Beckerstraße 31 D - 09120 Chemnitz Telefon: +49 (0)371/35 55-0 Telefax: +49 (0)371/35 55-333	Vertriebsgebiet West ☒ SALES Germany West ☒ Service INDRAMAT GmbH Harkortstraße 25 D - 40849 Ratingen Telefon: +49 (0)2102/43 18-0 Telefax: +49 (0)2102/41 315	Vertriebsgebiet Nord ☒ SALES Germany North ☐ Service INDRAMAT GmbH Kieler Straße 212 D - 22525 Hamburg Telefon: +49 (0)40/85 31 57-0 Telefax: +49 (0)40/85 31 57-15
Vertriebsgebiet Süd ☒ SALES Germany South ☐ Service INDRAMAT GmbH Ridlerstraße 75 D-80339 München Telefon: +49 (0)89/540138-30 Telefax: +49 (0)89/540138-10 indramat.mue@t-online.de	Gebiet Südwest ☒ SALES Germany South-West ☒ Service INDRAMAT GmbH Böblinger Straße 25 D-71229 Leonberg Telefon: +49 (0)7152/9 72-6 Telefax: +49 (0)7152/9 72-727	Vertriebsgebiet Mitte ☒ SALES Germany Centre ☐ Service Mannesmann Rexroth AG Geschäftsbereich INDRAMAT Lillistraße 14-18 D – 63067 Offenbach Telefon: +49 (0) 62/82 00 90-0 Telefax: +49 (0) 62/82 00 90-80	Vertriebsgebiet Nord ☒ SALES Germany North ☐ Service Mannesmann Rexroth AG Vertriebsniederlassung Region Nord Geschäftsbereich INDRAMAT Schützenstraße 20 D – 30853 Langenhagen Telefon: +49 (0) 511/72 66 57-0 Telefax: +49 (0) 511/72 66 57-93
			INDRAMAT Service-Hotline INDRAMAT GmbH Telefon: +49 (0)172/660 04 06 oder/or Telefon: +49 (0)171/333 88 26 Hotline ERSATZTEILE: - nur an Werktagen von 15 -18 Uhr - +49 (0) 93 52/40 42 22

Kundenbetreuungsstellen in Deutschland - Service agencies in Germany

Europa – Europe

vom Ausland: (0) nach Landeskennziffer weglassen, 0 nach Landeskennziffer mitwählen!
from abroad: don't dial (0) after country code, dial 0 after country code!

Austria ☒ SALES ☐ Service Mannesmann Rexroth Ges.m.b.H. Geschäftsbereich INDRAMAT Hägelingasse 3 A - 1140 Wien Telefon: +43 (0)1/9852540-400 Telefax: +43 (0)1/9852540-93	Austria ☒ SALES ☐ Service Mannesmann Rexroth G.m.b.H. Geschäftsbereich INDRAMAT Industriepark 18 A - 4061 Pasching Telefon: +43 (0)7221/605-0 Telefax: +43 (0)7221/605-21	Belgium ☒ SALES ☒ Service Mannesmann Rexroth N.V.-S.A. Geschäftsbereich INDRAMAT Industrielaan 8 B-1740 Ternat Telefon: +32 (0)2/5823180 Telefax: +32 (0)2/5824310	Denmark ☒ SALES ☒ Service BEC AS Zinkvej 6 DK-8900 Randers Telefon: +45 (0)87/11 90 60 Telefax: +45 (0)87/11 90 61
England ☒ SALES ☒ Service Mannesmann Rexroth Ltd. INDRAMAT Division Broadway Lane, South Cerney GB - Cirencester, Glos GL7 5UH Telefon: +44 (0)1285/863000 Telefax: +44 (0)1285/863030	Finland ☒ SALES ☐ Service Rexroth Mecman OY INDRAMAT division SF-017 40 Vantaa Telefon: +358 (0)9/84 91 11 Telefax: +358 (0)9/84 91 13 60	France ☒ SALES ☒ Service Mannesmann Rexroth S.A. Division INDRAMAT Parc des Barbanniers 4, Place du Village F-92632 Gennevilliers Cedex Telefon: +33 (0)141 47 54 30 Telefax: +33 (0)147 94 69 41 Hotline: +33 (0)6 08 33 43 28	France ☒ SALES ☐ Service Mannesmann Rexroth S.A. Division INDRAMAT 270, Avenue de Lardenne F - 31100 Toulouse Telefon: +33 (0)5 61 49 95 19 Telefax: +33 (0)5 61 31 00 41
France ☒ SALES ☐ Service Mannesmann Rexroth S.A. Division INDRAMAT 91, Bd. Irène Joliot-Curie F - 69634 Vénissieux – Cedex Telefon: +33 (0)4 78 78 53 65 Telefax: +33 (0)4 78 78 52 53	Italy ☒ SALES ☒ Service Mannesmann Rexroth S.p.A. Divisione INDRAMAT Via G. Di Vittoria, 1 I - 20063 Cernusco S/N.MI Telefon: +39 02/92 36 52 70 Telefax: +39 02/92 36 55 12	Italy ☒ SALES ☒ Service Mannesmann Rexroth S.p.A. Divisione INDRAMAT Via Borgomanero, 11 I - 10145 Torino Telefon: +39 011/7 50 38 11 Telefax: +39 011/7 71 01 90	Italy ☒ SALES ☐ Service Mannesmann Rexroth S.p.A. Divisione INDRAMAT Via del Progresso, 16 (Zona Ind.) I - 35020 Padova Telefon: +39 049/8 70 13 70 Telefax: +39 049/8 70 13 77
Italy ☐ SALES ☒ Service Mannesmann Rexroth S.p.A. Divisione INDRAMAT Via de Nicola, 12 I - 80053 Castellamare di Stabia NA Telefon: +39 081/8 72 30 37 Telefax: +39 081/8 72 30 18	Italy ☒ SALES ☐ Service Mannesmann Rexroth S.p.A. Divisione INDRAMAT Viale Oriani, 38/A I - 40137 Bologna Telefon: +39 051/34 14 14 Telefax: +39 051/34 14 22	Netherlands ☒ SALES ☐ Service Hydraudyne Hydrauliek B.V. Kruisbroeksestraat 1 (P.O. Box 32) NL - 5281 RV Boxtel Telefon: +31 (0)411/65 19 51 Telefax: +31 (0)411/65 14 83 e-mail: indramat@hydraudyne.nl	Netherlands ☐ SALES ☒ Service Hydrocare B.V. Kruisbroeksestraat 1 (P.O. Box 32) NL - 5281 RV Boxtel Telefon: +31 (0)411/65 19 51 Telefax: +31 (0)411/67 78 14
Norway ☒ SALES ☐ Service Rexroth Mecman AS INDRAMAT Division Berghagan 1 or Box 3007 N–1405 Ski-Langhus N-1402 Ski Telefon: +47 (0)64 86 41 00 Telefax: +47 (0)64 86 90 62	Poland ☒ SALES ☐ Service Mannesmann Rexroth Sp.zo.o. Biuro Poznan ul. Dabrowskiego 81/85 PL – 60-529 Poznan Telefon: +48 061/847 67 99 Telefax: +48 061/847 64 02	Russia ☐ SALES ☒ Service Tschudnenko E.B. Arsenia 22 RUS - 153000 Ivanovo Rußland Telefon: +7 093/223 96 33 oder/or +7 093/223 95 48 Telefax: +7 093/223 46 01	Spain ☒ SALES ☒ Service Mannesmann Rexroth S.A. Divisiòn INDRAMAT Centro Industrial Santiga Obradors s/n E-08130 Santa Perpetua de Mogoda Barcelona Telefon: +34 937 47 94 00 Telefax: +34 937 47 94 01
Spain ☒ SALES ☒ Service Goimendi S.A. División Indramat Jolastokieta (Herrera) Apartado 11 37 E - 20017 San Sebastian Telefon: +34 9 43/40 01 63 Telefax: +34 9 43/39 17 99	Sweden ☒ SALES ☒ Service Rexroth Mecman Svenska AB INDRAMAT Division Varuvägen 7 S - 125 81 Stockholm Telefon: +46 (0)8/727 92 00 Telefax: +46 (0)8/647 32 77	Slowenia ☒ SALES ☒ Service INDRAMAT elektromotorji d.o.o. Otoki 21 SLO - 64 228 Zelezniki Telefon: +386 64/61 73 32 Telefax: +386 64/64 71 50	Turkey ☒ SALES ☒ Service Mannesmann Rexroth Hidropar A..S. Fevzi Cakmak Cad No. 3 TR - 34630 Sefaköy Istanbul Telefon: +90 212/541 60 70 Telefax: +90 212/599 34 07
Switzerland ☒ SALES -East- ☒ Service Mannesmann Rexroth Schweiz AG Geschäftsbereich INDRAMAT Gewerbestraße 3 CH-8500 Frauenfeld Telefon: +41 (0)52/720 21 00 Telefax: +41 (0)52/720 21 11	Switzerland ☒ SALES -West- ☐ Service Mannesmann Rexroth Suisse SA Département INDRAMAT Rue du village 1 CH-1020 Renens Telefon: +41 (0)21/632 84 20 Telefax: +41 (0)21/632 84 21		

Europäische Kundenbetreuungsstellen (ohne Deutschland)
European Service agencies (without Germany)

Außerhalb Europa - outside Europe

vom Ausland: (0) nach Landeskennziffer weglassen!
from abroad: don't dial (0) after country code!

Argentina ☒ SALES ☐ Service Mannesmann Rexroth S.A.I.C. Division INDRAMAT Acassusso 48 41/7 RA - 1605 Munro (Buenos Aires) Telefon: +54 (0)11/4756 01 40 Telefax: +54 (0)11/4762 6862 e-mail:mannesmann@impsat1.com.ar	Argentina ☒ SALES ☒ Service NAKASE Servicio Tecnico CNC Calle 49, No. 5764/66 RA - 1653 Villa Balester Prov. - Buenos Aires Telefon: +54 (0) 11/4768 36 43 Telefax: +54 (0) 11/4768 24 13 e-mail: nakase@usa.net nakase@infovia.com.ar	Australia ☒ SALES ☒ Service AIMS - Australian Industrial Machinery Services Pty. Ltd. Unit 3/45 Horne ST Campbellfield , VIC 3061 AUS - Melbourne Telefon: +61 (0)3/93 59 02 28 Telefax: +61 (0)3/93 59 02 86	Australia ☒ SALES ☐ Service Mannesmann Rexroth Pty. Ltd. No. 7, Endeavour Way Braeside Victoria, 31 95 AUS – Melbourne Telefon: +61 (0)3/95 80 39 33 Telefax: +61 (0)3/95 80 17 33 Email: mel@rexroth.com.au
Brazil ☒ SALES ☒ Service Mannesmann Rexroth Automação Ltda. Divisão INDRAMAT Rua Georg Rexroth, 609 Vila Padre Anchieta BR - 09951-270 Diadema-SP [Caixa Postal 377] [BR-09901-970 Diadema-SP] Telefon: +55 (0)11/745 90 60 +55 (0)11/745 90 70 Telefax: +55 (0)11/745 90 50	Brazil ☐ SALES ☒ Service Mannesmann Rexroth Automação Ltda. Divisão INDRAMAT Rua Umberto Pinheiro Vieira, 100 Distrito Industrial BR - 09220-390 Joinville - SC [Caixa Postal 1273] Tel./Fax: +55 (0)47/473 55 833 Mobil: +55 (0)47 974 6645 e-mail: prochnow@zaz.com.br	Canada ☒ SALES ☒ Service Basic Technologies Corporation Burlington Division 3426 Mainway Drive Burlington, Ontario Canada L7M 1A8 Telefon: +1 905/335 55 11 Telefax: +1 905/335-41 84	China ☐ SALES ☒ Service Mannesmann Rexroth (China) Ldt. Shanghai Parts & Service Center 199 Wu Cao Road, Hua Cao Minhang District PRC - Shanghai 201 103 Telefon: +86 21/62 20 00 58 Telefax: +86 21/62 20 00 68
China ☒ SALES ☐ Service Mannesmann Rexroth (China) Ldt. 15/F China World Trade Center 1, Jianguomenwai Avenue PRC - Beijing 100004 Telefon: +86 10/65 05 03 80 Telefax: +86 10/65 05 03 79	China ☒ SALES ☐ Service Mannesmann Rexroth (China) Ldt. A-5F., 123 Lian Shan Street Sha He Kou District PRC - Dalian 116 023 Telefon: +86 411/46 78 930 Telefax: +86 411/46 78 932	Hongkong ☒ SALES ☒ Service Rexroth (China) Ldt. 1/F., 19 Cheung Shun Street Cheung Sha Wan, Kowloon, Hongkong Telefon: +852 22 62 51 00 Telefax: +852 27 44 02 78	India ☒ SALES ☒ Service Mannesmann Rexroth (India) Ltd. INDRAMAT Division Plot. 96, Phase III Peenya Industrial Area IND - Bangalore - 560058 Telefon: +91 (0)80/8 39 73 74 Telefax: +91 (0)80/8 39 43 45
India ☒ SALES ☒ Service Mannesmann Rexroth (India) Ltd. INDRAMAT Division Plot. A-58, TTC Industrial Area Thane Turbhe Midc Road Mahape Village IND - Navi Mumbai - 400 701 Telefon: +91 (0)22/7 61 46 22 Telefax: +91 (0)22/7 68 15 31	Indonesia ☒ SALES ☐ Service PT. Rexroth Wijayakusuma Jl. Raya Bekasi Km 21 Pulogadung RI - Jakarta Timur 13920 Telefon: +62 21/4 61 04 87 +62 21/4 61 04 88 Telefax: +62 21/4 60 01 52	Japan ☐ SALES ☒ Service Rexroth Automation Co., Ltd. Service Center Japan Yutakagaoka 1810, Meito-ku, NAGOYA 465-0035, Japan Telefon: +81 (0)52/777 88 41 +81 (0)52/777 88 53 +81 (0)52/777 88 79 Telefax: +81 (0)52/777 89 01	Japan ☒ SALES ☒ Service Rexroth Automation Co., Ltd. INDRAMAT Division 1F, I.R. Building Nakamachidai 4-26-44, Tsuzuki-ku YOKOHAMA 224-0041, Japan Telefon: +81 (0)45/942 72 10 Telefax: +81 (0)45/942 03 41
Mexico ☒ SALES ☐ Service Rexroth Mexico S.A. de C.V. Calle Neptuno 72 Unidad Ind. Vallejo MEX - 07700 Mexico, D.F. Telefon: +52 5 754 17 11 +52 5 754 36 84 +52 5 754 12 60 Telefax: +52 5 754 50 73 +52 5 752 59 43	Korea ☒ SALES ☒ Service Mannesmann Rexroth-Seki Co Ltd. 1500-12 Da-Dae-Dong ROK - Saha-Ku, Pusan, 604-050 Telefon: +82 (0)51/2 60 06 18 Telefax: +82 (0)51/2 60 06 19	Korea ☒ SALES ☒ Service Seo Chang Corporation Ltd. Room 903, Jeail Building 44-35 Yeouido-Dong Yeoungdeungpo-Ku C.P.O.Box 97 56 ROK - Seoul Telefon: +82 (0)2/7 80 82 08 +82 (0)2/7 80 82 09 Telefax: +82 (0)2/7 84 54 08	South Africa ☒ SALES ☒ Service TECTRA Automation (Pty) Ltd. 28 Banfield Road,Industria North RSA - Maraisburg 1700 Telefon: +27 (0)11/673 20 80 Telefax: +27 (0)11/673 72 69
Taiwan ☒ SALES ☐ Service Rexroth Uchida Co., Ltd. No.1, Tsu Chiang Street Tu Cheng Ind. Estate Taipei Hsien, Taiwan, R.O.C. Telefon: +886 2/2 68 13 47 Telefax: +886 2/2 68 53 88			

Kundenbetreuungsstellen außerhalb Europa - Service agencies outside Europe

Außerhalb Europa / USA - outside Europe / USA

USA ☒ SALES ☒ Service Mannesmann Rexroth Corporation INDRAMAT Division 5150 Prairie Stone Parkway USA -Hoffman Estates, IL 60192-3707 Telefon: +1 847/6 45 36 00 Telefax: +1 847/6 45 62 01	USA ☒ SALES ☒ Service Mannesmann Rexroth Corporation INDRAMAT Division Central Region Technical Center USA - Auburn Hills, MI 48326 Telefon: +1 248/3 93 33 30 Telefax: +1 248/3 93 29 06	USA ☒ SALES ☒ Service Mannesmann Rexroth Corporation INDRAMAT Division Southeastern Technical Center 3625 Swiftwater Park Drive USA - Suwanee Georgia 30174 Telefon: +1 770/9 32 32 00 +1 770/9 32 19 03	USA ☒ SALES ☒ Service Mannesmann Rexroth Corporation INDRAMAT Division Northeastern Technical Center 99 Rainbow Road USA - East Granby, Connecticut 06026 Telefon: +1 860/8 44 83 77 +1 860/8 44 85 95
USA ☒ SALES ☒ Service Mannesmann Rexroth Corporation INDRAMAT Division Charlotte Regional Sales Office 14001 South Lakes Drive USA - Charlotte, North Carolina 28273 Telefon: +1 704/5 83 97 62 +1 704/5 83 14 86			

Kundenbetreuungsstellen außerhalb Europa / USA
Service agencies outside Europe / USA

Notes

282792

Printed in Germany

Rexroth Indramat